K^{*0} Resonance Production and Elliptic Flow at STAR

A thesis Submitted
in Partial Fulfilment of the Requirements
for the Degree of

MASTER OF SCIENCE

by

Arabinda Behera

to the

School of Physical Sciences

National Institute of Science Education and Research

Bhubaneswar

Date - 26.04.2015

DECLARATION

I hereby declare that I am the sole author of this thesis in partial fulfillment of the requirements for a postgraduate degree from National Institute of Science Education and Research (NISER). I authorize NISER to lend this thesis to other institutions or individuals for the purpose of scholarly research.

Signature of the Student

Date:

The thesis work reported in the thesis entitled K^{*0} **Resonance Production and Elliptic Flow at STAR** was carried out under my supervision, in the school of Physical Sciences at NISER, Bhubaneswar, India.

Signature of the thesis supervisor

School:

Date:

ACKNOWLEDGEMENTS

First and foremost I would like to express my deepest gratitude to my supervisor Prof. Bedangadas Mohanty for giving me the opportunity to work in the exciting field of "Relativistic Heavy Ion Collisions". He has been a constant source of inspiration to me for pursuing experimental high energy physics. His guidance, fundamental approach to research, insights in experimental work and rigorous discussions have been of great help for me to have a clear understanding of the project. Without him this project would not have been possible.

I would like to thank Dr. Sandeep Chatterjee of VECC, Kolkata for helping me in developing a theoretical model for rescattering and regeneration. He guided me in writing the Mathematica code to carry out numerical calculations using the developed model and also provided me with important experimental inputs necessary in the code like cross-section values for different interactions. I thank Dr. Subhash Singha, who just completed his PHD at NISER, for guiding me to learn AMPT model and coding ROOT macros for generating events using AMPT model and analysing the data. I thank the members of Prof. Bedangadas Mohanty's group - Dr. Ranbir Singh, Dr. Ajay Kumar Dash, Rihan Haque and Vipul Bairathi for allowing me to run my ROOT macros on their desktop and their valuable suggestions and discussions which made me more confident about my work.

I am also thankful to my classmates - Jyotishman Sahoo and Abhishek Mathur for discussions and help in theoretical derivations.

ABSTRACT

This thesis presents the study of K^{*0} resonance production and elliptic flow with the aim of probing the characteristics of the medium created in ultra-relativistic nucleus-nucleus collisions. K^{*0} has average lifetime $\sim$4 fm/c which is comparable to the hadronic cascade time of the fireball ($\sim$10 fm/c). So, K^{*0} can decay within the hadronic phase and the daughter particles can interact with the medium particles which will shape the final K^{*0} yield. These interactions are called in-medium effects and categorised as Rescattering and Regeneration. First the production of resonances in heavy ion collisions and their properties like mass, width and lifetime obtained from the invariant mass distribution of the daughter particles is studied. The shape of the invariant mass distribution following a Breit-Wigner function is derived using Quantum Scattering and Phase shift. After getting a detailed idea about resonances and their properties, a simulation of collision of heavy nuclei (Au-Au) at $\sqrt{s_{NN}} = 200GeV$ is done using AMPT model. The simulated data is analysed to obtain the invariant mass distribution which gives the yield, mass and width of K^{*0} produced in the simulation. The reconstructed K^{*0} yield is observed to decrease with the increase in hadronic cascade time. To explain this behaviour a theoretical model was developed which demonstrated the effect of in-medium effects like Rescattering and Regeneration on the K^{*0} yield.

The second part involves experimental data analysis of U-U collision data of STAR at center of mass energy of 193 GeV ($\sqrt{s_{NN}} = 193GeV$) for 9.98 million events for K^{*0} study. Various experimental variables like transverse momentum, pseudorapidity, vertex position etc. and their distributions are obtained from the data. The invariant mass distribution of K^{*0} is obtained which gives the actual yield, mass and width of K^{*0} produced in the experiment. Finally the "Elliptic flow" (v_2)of K^{*0} is studied, which is a very important property of the resonance and gives the experi-

mental measurement of the momentum anisotropies produced in the collective flow of the particle. The event planes are constructed and corrected and used to obtain the v_2 of K^{*0}. The plot of v_2 as a function of p_T for minimum-bias is obtained. The non-zero value of v_2 confirms the presence of anisotropy in collective flow of K^{*0}. The elliptic flow of K^{*0} is then compared with that of Φ which is also a meson resonance like K^{*0} with mass close to K^{*0}. The plot shows that v_2 of both the particles are quite similar as expected which confirms the efficiency of the analysis. To show the centrality dependance, v_2 vs p_T is plotted for different centralities. Comparision of the plots shows that v_2 increases with increase in centrality. This confirms the presence of different degree of anisotropies for different centralities.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The field of the ultra-relativistic heavy-ion collisions connects two big branches of physics namely the nuclear physics and the elementary particle physics. The unifying aspect is an attempt to analyze the properties of dense hadronic matter in terms of elementary interactions. The fundamental theory of strong interactions is Quantum Chromodynamics (QCD). QCD predicts that at high energy density, hadronic matter will turn into a plasma of deconfined quarks and gluons (QGP). The search for such a phase transition is the main motivation for the continuous experimental and theoretical efforts in the field. One of the goals of the experimental program is to recreate (on a microscopic scale) the physical conditions that are thought to have existed in the early universe. Lattice QCD calculations predict a phase transition from hadronic matter to quark gluon plasma (QGP) at high temperatures and/or high densities. Matter under such extreme conditions can be studied in the laboratory by colliding heavy nuclei at very high energies. The Relativistic Heavy Ion Collider (RHIC) at Brookhaven National Laboratory provides collisions of heavy nuclei and protons at center of mass energies up to $\sqrt{s_{NN}} = 200$ GeV.

Heavy ion collisions produce a lot of particles with different lifetimes. Resonances are particles with shorter lifetime which decay into their daughter particles. They can only be detected by the invariant mass reconstruction of their daughter particles.

The typical lifetime of a resonance is a few fm/c, which is comparable to the expected lifetime of the hot and dense matter produced in heavy-ion collisions. Inside hadron gas, which is a hot and dense system after Chemical freeze-out, resonances are in close proximity with other strongly interacting hadrons. The interactions of the daughter particles of the resonance with the medium gives rise to in-medium effects related to the high density and/or high temperature of the medium which can modify various resonance properties, such as mass, width, and even the mass line shape. Thus, measurements of various resonance properties can provide detailed information about the interaction dynamics in relativistic heavy-ion collisions. K^{*0} is one such resonance with average lifetime $\sim$4 fm/c which is comparable to the hadronic cascade time (time periond between Chemical and Kinetic freeze-out) of the fireball ($\sim$10 fm/c). So, K^{*0} can decay within the hadronic phase and the daughter particles can interact with the medium particles which will shape the final K^{*0} yield.

In this report we present the study of K^{*0} production via its hadronic decay channel ($K^{*0} \rightarrow K^{+}\pi^{-}$ and $\overline{K^{*0}} \rightarrow K^{-}\pi^{+}$). The study is divided in two parts. In the first part we will learn in detail about resonances, their production in heavy ion collision experiments and their properties like mass, width and lifetime from the invariant mass distribution of the daughter particles. Then we will use AMPT model to simulate the collision experiment of two gold (Au) nuclei at center of mass energy of 200 GeV ($\sqrt{s_{NN}} = 200GeV$) with a fixed hadronic cascade time (τ_{HC}). The simulated data will be analysed to get the invariant mass distribution of K^{*0} and fitted with Breit-Wigner function to obtain different parameters like the mass, yield and width of the resonance. By varying the hadronic cascade time (τ_{HC}), the effect of τ_{HC} on the K^{*0} yield will be seen. Finally a theoretical model will be developed to explain this effect by studying the interaction of the daughter particles of K^{*0} - Kaons and

Pions with the hadron gas medium giving rise to two important phenomena called Rescattering and Regeneration. The effect of these phenomena on the K^{*0} yield will be shown by simulating the model to obtain the K^{*0} yield as a function of hadronic cascade time τ_{HC}.

The second part involves experimental data analysis of U-U collision data of STAR at center of mass energy of 193 GeV ($\sqrt{s_{NN}} = 193GeV$). The analysis is done to get the invariant mass distribution of K^{*0} from experiment and fitted with Breit-Wigner function to obtain different parameters like the mass, yield and width of the resonance to verify the results obtained from simulation. The results obtained are compared with the PDG values to check the efficiency of the analysis. Finally we study "Elliptic flow" of K^{*0}, which is a very important property of the resonance. Elliptic flow is the experimental measurement of the momentum anisotropies produced in the collective flow of the particle. The elliptic flow of K^{*0} is compared with that of Φ which is also a meson resonance like K^{*0} with mass close to K^{*0}. The elliptic flow is plotted for different centralities to show its dependence on centrality and initial position anisotropy.

Chapter 2

Resonances in Heavy Ion Collisions

2.1 Relativistic Heavy Ion Collisions

In a heavy ion collision experiment two heavy nuclei collide at relativistic velocities. Due to very high velocity they are Lorentz contracted along the direction of motion and appears like a pan cake. Fig. 2.1 describes schematically different stages in a heavy ion collision. The energy carried by the incoming hadrons are deposited within a small region in space in a short duration of time. The energy density at the collision center is sufficiently high and it can form a strongly interacting matter consisting of deconfined quarks and gluons, called quark gluon plasma (QGP). Model calculations indicate that beyond a critical energy density of $\sim 1GeV/fm^3$ (or temperature ~ 200 MeV) matter can exist in a QGP phase. Initially the QGP may not be in thermal equilibrium, but subsequent expansion may bring it to a local thermal equilibrium at some proper time τ_0 . The plasma then may behave like a hydrodynamic fluid.

The space-time (z-t) evolution of heavy ion collision is shown in Fig. 2.2. As the system expands both temperature and density decreases. Eventually, at a particular temperature called Critical Temperature (T_c), quark confinement occurs leading to particle production. Hadrons are produced from free quarks and gluons. On further expansion and cooling two important stages occur called - Chemical freeze-out and

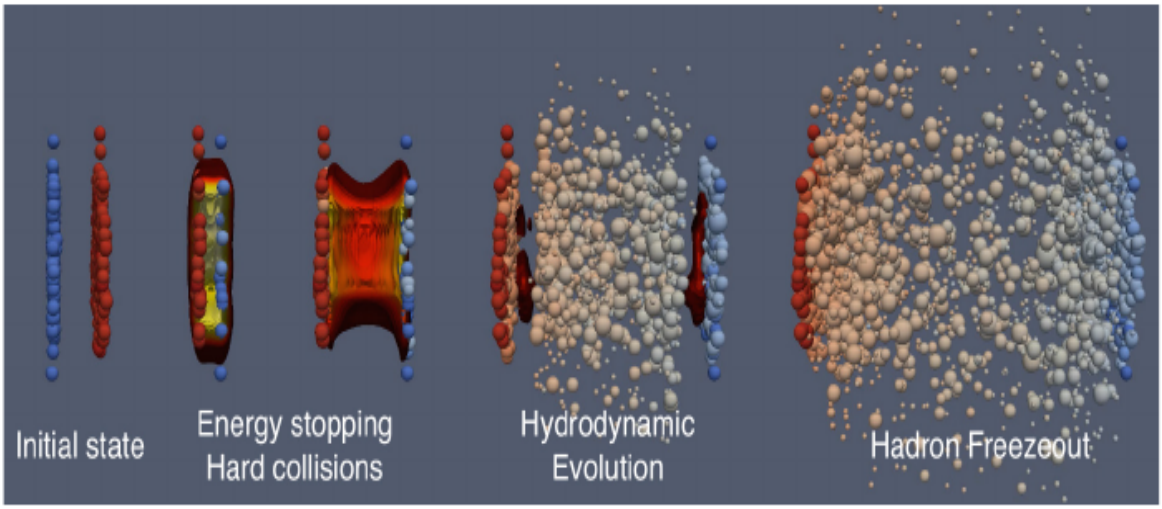

Figure 2.1: Different stages in heavy ion collision process

kinetic-freeze-out.

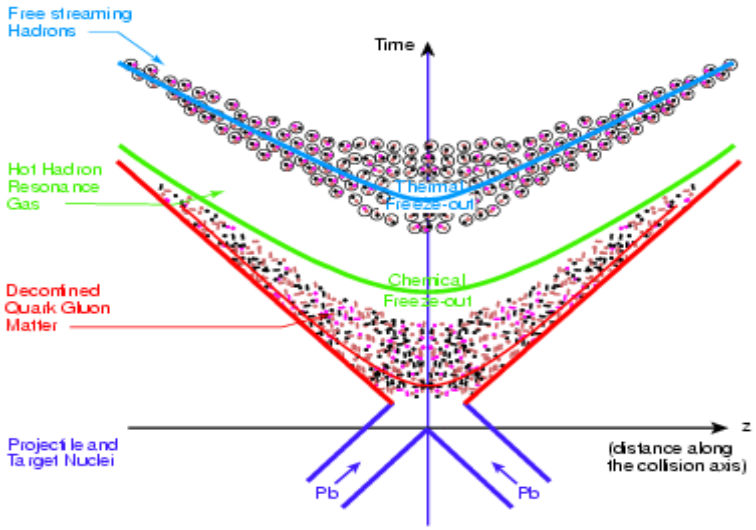

Figure 2.2: Space-time evolution of a system formed by relativistic heavy ion collision

Chemical freeze-out occurs when the temperature of the system falls below a freeze out temperature called chemical freeze out temperature (T_{ch}). Its called chemical freeze-out because at this point the chemical composition of the produced particles get fixed. All inelastic collisions stop as the mean free path of the particles becomes

comparable with the inelastic cross-section. Particle production stops and particle ratios become fixed. At this point the system is in thermal and chemical equilibrium. The system is now a non-interacting ideal gas composed of hadrons called "Hadron gas". After the chemical freeze out the constituents can interact among themselves via elastic scattering only which may further change the shape of their transverse momentum spectra. On further expansion and cooling of the system, at a particular temperature called kinetic freeze out temperature (T_{fo}), kinetic or thermal freeze-out takes place. At this stage, the mean free path of the hadrons exceeds the dynamical size of the system, and the elastic interaction among the hadrons ceases. The momentum distribution of the particles become fixed. After that the hadrons stream freely to the detector and get detected. The whole system from QGP to free streaming particles is called fireball. And the phase between Chemical and Kinetic Freeze-out is called Hadron gas.

2.2 Resonances

A lot of particles are produced in the heavy ion collision. Most particles have lifetimes $\tau \geq 10^{-12}$ sec, which do not decay before hitting the detectors like protons. These are called stable particles and are observed directly in the detectors. But there are many particles with shorter lifetimes which decay into daughter particles before hitting the detectors. These unstable particles are called "Resonances" which decay in a very short time through strong interactions. The resonances can not be detected directly in an experiment. The only way they detected is by reconstructing the invariant mass of their decay products. Resonance are produced by the interaction of their daughter particles form the intermediate "Resonance" which then again decays back to the daughter particles.

A resonance is observed as a peak located around a certain energy found in the plot of differential cross sections as a function of energy in scattering experiments. These peaks are associated with subatomic particles (baryons and mesons like K^{*0}, Λ, ρ etc) and their excitations. The width of the resonance (Γ) is related to the lifetime of the particle (τ) by the relation : $\Gamma = \hbar/\tau$. The experimental measurement of short-lived hadron resonances can potentially be very useful in clarifying some of the least understood aspects of heavy ion collisions.

2.2.1 Invariant Mass

Suppose 'R' is a resonance particle with mass 'M', decays into its two daughter particles 'A' and 'B'. In the laboratory frame let the energy and momentum of 'R' before its decay be E_R and $\vec{p}_R$ respectively. After its decay into daughter particles 'A and 'B', let the energy and momentum of 'A' be E_A and $\vec{p}_A$, and energy and momentum of 'B' be E_B and $\vec{p}_B$ respectively. The four momentum of the particles are given as follows :

$$P_R = (E_R, \vec{p}_R) \tag{2.1}$$

$$P_A = (E_A, \vec{p}_A) \tag{2.2}$$

$$P_B = (E_B, \vec{p}_B) \tag{2.3}$$

The decay is shown in Fig. 2.3.

Now, conservation of four momentum gives :

$$\begin{aligned} P_R &= P_A + P_B \\ \Rightarrow P_R^2 &= (P_A + P_B)^2 \\ \Rightarrow M^2 &= (E_A + E_B)^2 - (\vec{p}_A + \vec{p}_B)^2 \\ \Rightarrow M &= \sqrt{(E_A + E_B)^2 - (\vec{p}_A + \vec{p}_B)^2} \end{aligned} \tag{2.4}$$

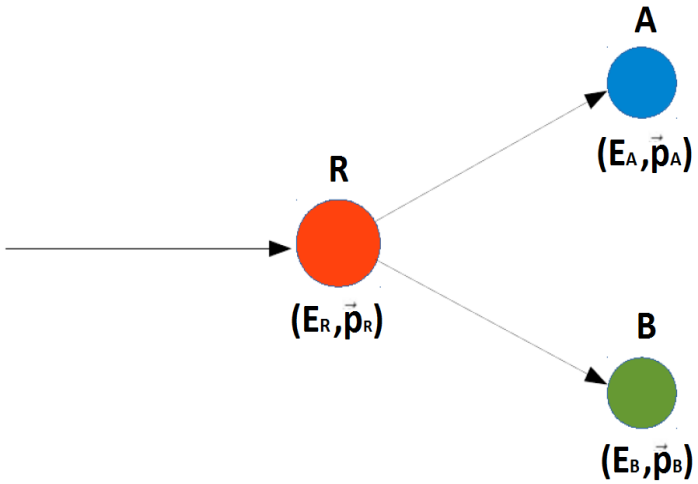

Figure 2.3: Two body decay [8] picture of a resonance R into A and B

Invariant mass [8] of the resonance is given by Eq. (2.4). The name invariant signifies that it is frame independent which means its value remains the same in different coordinate frames (for example laboratory frame or centre of mass frame).

2.3 Motivation

Resonance	Mass (in MeV/c^2)	Major Decay Channel	Branching Ratio ($\mathcal{B}$)	τ (in fm/c)
ρ^0	770	$\pi^+\pi^-$	1	1.1
Δ	1232	$p\pi$	1	1.6
f^0	980	$\pi^+\pi^-$	2/3	2.6
K^{*0}	892	$K\pi$	2/3	4
Σ^*	1385	$\Delta\pi$	0.88	5.5
Λ^*	1520	pK	0.45	12.6
Φ	1020	K^+K^-	0.49	45

Table 2.1: Resonances produced in heavy ion collision and their properties

Some of the important resonances measured in high energy experiments and their properties like their mass, decay channels, branching ratio and lifetime are given in

the Table 2.1. The fireball lifetime in heavy ion collisions is of the order of 45 fm/c. As shown in the table the resonances have lifetime within the range 1.1 to 45 fm/c, which covers the typical lifetime of the fireball produced in heavy ion collisions. The short lived resonances are very useful tool in high energy collisions to study the dynamics and properties of a strongly interacting medium. By studying the properties of the above resonance we can get a complete picture of the details of the dynamic processes going inside the fireball.

We are interested in one of these resonance which is K^{*0}. It is chosen because of its short average lifetime $\sim$4 fm/c which is comparable to the time interval between the two frezze-outs, called the lifetime of the hadron gas or the hadronic cascade ($\sim$10 fm/c). It means K^{*0} can decay within the hadron gas and the daughter particles - K and π can interact with the medium particles of the hadron gas and give rise to in-medium effects like Rescattering and Regeneration which will be discussed in detail in later sections of the report. Rescattering process decreases the resonance yield and regeneration process increases the resonance yield. Model calculations suggest that such re-scattering is most probable for low momentum resonances. High momentum resonances are more likely escape the medium and hence may not be affected by re-scattering. These two competing processes, the re-scattering and re-generation, will decide the final yield of the resonance. The rescattering and regeneration effect can be understood better through resonance to non-resonance ratio (such as K^{*0}/K^{-}) in AA and pp collisions. A naive expectation of re-scattering and re-generation effect on K^{*0}/K^{-} ratio is qualitatively shown in Fig. 2.4. If there is only re-scattering effect present in the hadronic phase of heavy ion collision, then this ratio should be smaller than pp collision and follow the blue line as a function of collision centrality. Here, collision centrality is a measure of how central the collision between the two nuclei

was and determines the overlapping region of the two nuclei in the collision. Thus the suppression in central collision should be more than peripheral collision. For re-generation only case the ratio should be above the pp measurement and follow the red line. The experimental measurement of collision centrality dependence K^{*0}/K^{-} ratio in heavy ion with respect to measurement in pp collisions is sensitive to the nature of the hadronic phase. Thus this ratio can be used to extract the lifetime of the hadron gas phase in heavy ion collision.

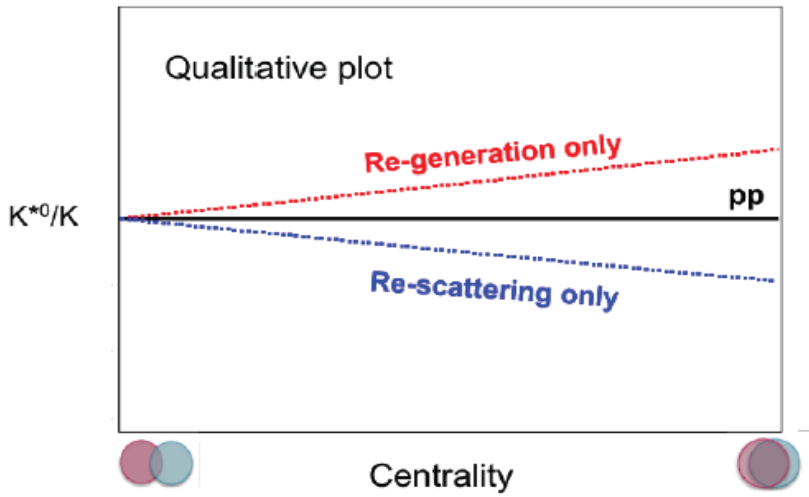

Figure 2.4: Qualitative plot showing the effect of re-scattering and regeneration only processes on K^{*0}/K^{-} ratio in heavy ion collision with respect to pp collision

Our aim in this project is to understand the systematics of the in-medium effects of rescattering and regeneration on the yield of K^{*0} resonance produced in the heavy ion collisions. We first will study the resonance production, their cross-section and their properties like their invariant mass distribution. Then we will try to see if there is any effect of rescattering and regeneration in K^{*0} yield using AMPT simulations. Finally we will try to develop a theoretical model to explain these effects and perform simulations using the model to get K^{*0} yield as function of hadronic cascade time.

2.4 Cross-section of Resonances

The cross section of a resonance particle follows a Breit-Wigner distribution. The derivation of this distribution [6] is presented in this section. By solving the time independent Schrodinger equation, stationary states are obtained as solutions represented as ψ_r.

$$H\psi = E\psi \tag{2.5}$$

Here, H is the time independent Hamiltonian and E is the energy eigenvalue. The stationary solutions are represented by $\psi(r)$. The solution of the time dependent Schrodinger equation consists of a time independent solution and a time dependent part given by :

$$\Psi(r,t) = \psi(r) \cdot e^{-iEt/\hbar} \tag{2.6}$$

Probability of finding a particle with this wave function is :

$$P = |\Psi(r,t)|^2 \tag{2.7}$$

Resonances decay with time, so similar to the radioactive decay process, the probability associated with the wave function should decay with time as follows :

$$P(t) = |\Psi(r,t)|^2(t) = |\psi(r)|^2 \cdot e^{-t/\tau} \tag{2.8}$$

where, $\tau = 1/W$ is the mean lifetime of the state with decay rate W. So, now the total time dependent wave function becomes :

$$\Psi(r,t) = \psi(r) \cdot e^{-iEt/\hbar} \cdot e^{-t/2\tau} \tag{2.9}$$

Now from the minimum energy-time uncertainty principle :

$$\Delta E \cdot \Delta t \simeq \hbar \;\Rightarrow\; \Delta E = \frac{\hbar}{\tau} \tag{2.10}$$

For strong decays τ is immeasurably small so another quantity decay width Γ is considered which is the natural spread in energy of the decaying state given by :

$$\Gamma = \frac{\hbar}{\tau} = \hbar W = 2\pi |M|^2 \int \rho_f d\Omega \tag{2.11}$$

where, ρ_f is the density of states and $d\Omega$ is the differential solid angle. Now the wave function for a non-stationary decaying state of central angular frequency $\omega_R = E_R/\hbar$, where E_R is the resonance energy, and the lifetime $\tau = \hbar/\Gamma$ can be written as :

$$\psi(t) = \psi(r) e^{-i\omega_R t} e^{-t/2\tau} = \psi(r) e^{-t(iE_R + \Gamma/2)} \tag{2.12}$$

Broad states with finite width and lifetimes, which can be formed by the collision of the particles into which they decay are called resonances. The energy dependence of the cross-section for creating the resonant state from its constituents is obtained from the Fourier transform of the time pulse. Taking the Fourier transform of the above equation :

$$g(\omega) = \int_0^\infty \psi(t) e^{i\omega t} \tag{2.13}$$

with $\omega = E/\hbar = E$ in natural units. The amplitude as a function of E is then :

$$\chi(E) = \int_0^\infty \psi(t) e^{i\omega t} = \psi(r) \int_0^\infty e^{-t[i(E_R - E) + \Gamma/2]} = \frac{\psi(0)}{(E - E_R + i\Gamma/2)} \tag{2.14}$$

The cross-section $\sigma(E)$ which gives the probability of two particles a and b forming the resonant state c will be proportional to $\chi^*(E)\chi(e)$:

$$\sigma(E) = \sigma_{max} \frac{\Gamma^2/4}{(E - E_R)^2 + \Gamma^2/4} \tag{2.15}$$

Its not the complete picture. To obtain the exact form of $\sigma(E)$, the exact form of σ_{max} has to be determined. For this we have to look at the quantum picture of the scattering theory [5].

2.4.1 Quantum Scattering Theory

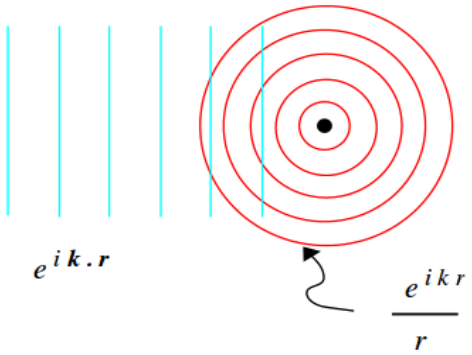

Figure 2.5: Quantum scattering from a potential

Assume an incident beam of particles traveling along z-direction encounters a potential and is scattered by the potential. The incoming beam can be represented by a plane wave Ae^{ikz}, corresponding to momentum $p = \hbar k$. Quantum scattering from the potential[5] results in outgoing spherical waves as shown in Fig. 2.5. So, the total wave function :

$$\psi(r,\theta) \approx A\left[e^{ikz} + f(\theta)\frac{e^{ikr}}{r}\right] \tag{2.16}$$

Here, $k \equiv \frac{\sqrt{2mE}}{\hbar}$ is the wave vector and f(θ) is the scattering amplitude. $f(\theta)$ gives the probability of scattering in a given direction θ. It seems very similar to differential cross-section which is defined as :

$$D(\theta) = \frac{d\sigma}{d\Omega} \tag{2.17}$$

A relation between differential cross-section and scattering amplitude can be derived as follows.

Probability of incident particle with velocity 'v' going through cross-sectional area 'dσ' in time 'dt' :

$$dP_{in} = |\psi_{in}|^2 dV = |A|^2 (vdt)d\sigma \tag{2.18}$$

Probability of observing scattered particle in solid angle $d\Omega$:

$$dP_{sc} = |\psi_{sc}|^2 dV = \frac{|A|^2|f|^2}{r^2}(vdt)r^2 d\Omega \tag{2.19}$$

These two probabilities should be equal.

$$dP_{in} = dP_{sc} \tag{2.20}$$

$$\Rightarrow D(\theta) = \frac{d\sigma}{d\Omega} = |f(\theta)|^2 \tag{2.21}$$

The scattering amplitude f(θ) is obtained by solving Quantum scattering problem using Partial Wave Analysis method[5].

2.4.2 Partial Wave Analysis

When the potential is central, i.e., spherically symmetric V ($\vec{r}$) = V (r), angular momentum is conserved due to Noethers theorem. Therefore, we can expand the wave function in the eigenstates of the angular momentum. Obtained waves with definite angular momenta are called partial waves. The main idea of the partial wave analysis method [5] is that it assumes the incoming wave as a superposition of different partial waves. Each partial wave is scattered individually from the scattering potential.So, the outgoing wave is also a superposition of partial spherical waves. We can solve the scattering problem for each partial wave separately, and then in the end put them together to obtain the full scattering amplitude.

Schrodinger's equation for a spherically symmetrical potential V(r) :

$$\frac{-\hbar}{2m}\nabla^2\psi + V(r)\psi = E\psi \tag{2.22}$$

admits the separable solutions :

$$\psi(r,\theta,\phi) = R(r)Y_l^m(\theta,\phi) \tag{2.23}$$

where, R(r) is the radial solution and $Y_l^m(\theta, \phi)$ is the angular solution called Spherical Harmonics. Taking $u(r) = rR(r)$, we get the radial equation :

$$\frac{-\hbar}{2m}\frac{d^2u}{dr^2} + \left[V(r) + \frac{-\hbar}{2m}\frac{l(l+1)}{r^2}\right]u = Eu \tag{2.24}$$

where, V(r) is the scattering potential and 'l' is the angular momentum quantum number. Before solving this radial equation, we divide the space in 3 zones because for finite scattering potential the value of potential falls with radial distance (r).

(i) Scattering region - Its the region close to the scattering centre where the potential is non-zero.

(ii) Intermediate region - In this region the potential can be ignored but not the centrifugal term.

(iii) Radiation zone - for very large value of r, potential is essentially zero.

At intermediate region, $V \equiv 0$, but centrifugal term is not 0. So, the radial equation becomes :

$$\frac{d^2u}{dr^2} + \frac{l(l+1)}{r^2}u = -k^2u \tag{2.25}$$

Its solution is represented in terms of spherical Bessel functions :

$$u(r) = A_r j_l(kr) + B_r n_l(kr) \tag{2.26}$$

However, neither j_l(∼sine function) nor n_l(∼cosine function) represents an outgoing (or incoming) wave. What we need are the linear combinations analogous to e^{ikr} and e^{-ikr} which are known as spherical Hankel functions.

$$h_l^{(1)}(x) \equiv= j_l(x) + in_l(x) \; ; \; h_l^{(2)}(x) \equiv= j_l(x) - in_l(x) \tag{2.27}$$

At large r, the Hankel function of first kind, $h_l^{(1)}(kr)$ goes like e^{ikr}/r, whereas the Hankel function of second kind, $h_l^{(2)}(kr)$ goes like e^{-ikr}/r. For outgoing waves, then,

we need spherical Hankel functions of the first kind :

$$R(r) \sim h_l^{(1)}(kr) \tag{2.28}$$

So, the wave function outside scattering region where $V(r) = 0$ is given by :

$$\psi(r,\theta,\phi) = A\left[e^{ikz} + \sum_{l,m} C_{l,m} h_l^{(1)}(kr) Y_l^m(\theta,\phi)\right] \tag{2.29}$$

Here, the first term is the incident plane wave and the second term is the outgoing spherical wave. Since, we assumed V(r) is spherically symmetric, so there is no 'ϕ'-dependence of ψ. This implies only m=0 terms survive.

$$Y_l^0(\theta,\phi) = \sqrt{\frac{2l+1}{4\pi}} P_l(cos\theta) \tag{2.30}$$

Now, the wave function becomes :

$$\psi(r,\theta) = A\left[e^{ikz} + \sum_{l=0}^{\infty} C_{l,0} \sqrt{\frac{2l+1}{4\pi}} h_l^{(1)}(kr) P_l(cos\theta)\right] \tag{2.31}$$

From solution of spherical harmonics $Y_l^m(\theta,\phi)$ of hydrogen atom the coefficient $C_{l,0}$ is obtained as:

$$C_{l,0} = k i^{l+1} \sqrt{4\pi(2l+1)} a_l \tag{2.32}$$

For very large r, the Hankel function goes like $(-i)^{l+1} e^{ikr}/kr$. So, Eq-(2.31) becomes :

$$\psi(r,\theta) = A\left[e^{ikz} + \sum_{l=0}^{\infty} (2l+1) a_l P_l(cos\theta) \frac{e^{ikr}}{r}\right] \tag{2.33}$$

Comparing this equation with Eq-(2.16) we get the scattering amplitude as :

$$f(\theta) = \sum_{l=0}^{\infty} (2l+1) a_l P_l(cos\theta) \tag{2.34}$$

It tells how to compute scattering amplitude, $f(\theta)$ in terms of partial wave amplitudes a_l. The differential cross-section is :

$$D(\theta) = |f(\theta)|^2 = \sum_l \sum_{l'} (2l+1)(2l'+1) a_l^* a_{l'} P_l(cos\theta) P_{l'}(cos\theta) \tag{2.35}$$

The total cross-section is the integral of differential cross-section.

$$\sigma = \int D(\theta) d\Omega = \int \sum_{l} \sum_{l'} (2l+1)(2l'+1) a_l^* a_{l'} P_l(cos\theta) P_{l'}(cos\theta) sin\theta d\theta d\phi \quad (2.36)$$

Using the orthogonality of the Legendre polynomials we get :

$$\sigma = 4\pi \sum_{l=0}^{\infty} (2l+1)|a_l|^2 \quad (2.37)$$

This equation is called Optical theorem. For finding σ, the partial wave amplitudes (a_l) have to be determined. This is accomplished by solving the Schrodinger equation in the interior region (where V(r) is non-zero) and matching with the exterior solution using proper boundary conditions. The problem here is - spherical coordinates are used for the scattered wave, but Cartesian coordinates are used for the incident wave. We need to rewrite the wave function in a more consistent way. This is done by using Rayleigh's formula which expands a plane wave in terms of spherical waves.

$$e^{ikz} = \sum_{l=0}^{\infty} i^l (2l+1) j_l(kr) P_l(cos\theta) \quad (2.38)$$

Using this the wave function in the exterior region can be expressed entirely in terms of spherical coordinates, r and θ.

$$\psi(r,\theta) = A \sum_{l=0}^{\infty} (2l+1) \left[j_l(kr) + ika_l h_l(kr) \right] P_l(cos\theta) \quad (2.39)$$

2.4.3 Phase Shift

For getting σ, we need to compute a_l first. Amplitudes are complex quantities, so finding them is a bit difficult. This problem can be made simpler by expressing σ in terms of phase shift (δ) which is a real quantity.

First lets consider the problem of one-dimensional scattering from a localized potential V(x) at x=0, so a wave incident from the left,

$$\psi_i(x) = Ae^{ikx} \quad (x < -a) \quad (2.40)$$

is entirely reflected.

$$\psi_r(x) = Be^{-ikx} \quad (x < -a) \tag{2.41}$$

By conservation of probability, the amplitude of incident and reflected wave must be same but may not have the same phase. If there is no potential present then, B=-A because the total wave function must vanish at x=0.

$$\psi_0(x) = A\left(e^{ikx} - e^{-ikx}\right) \quad (V(x) = 0) \tag{2.42}$$

If the potential is not zero, the wave function takes the form :

$$\psi(x) = A\left(e^{ikx} - e^{i(2\delta - kx)}\right) \quad (V(x) \neq 0) \tag{2.43}$$

Now, the whole scattering problem is reduced to the calculation of the phase shift. This is done by solving the Schrodinger equation in the region $(-a < x < 0)$, and imposing the appropriate boundary conditions. The use of makes the physical interpretation very simple. The conservation of probability limits the effect of potential to only shift the phase of the reflected wave.

Now, in our 3-dimensional scattering problem, the incident plane wave (Ae^{ikz}) has no angular momentum component in the z-direction ($m \neq 0$). But it includes all values of the total angular momentum (l=0,1,2,3,4...) as by Rayleigh's formula the incident plane wave can be expressed as a linear combination of partial spherical waves. Because in a spherically symmetric potential angular momentum is conserved, each partial wave (labelled by a particular l) scatters independently, with no change in amplitude but only in phase. The 'l'th incident partial wave in V(r)=0, is (from Eq.(2.38))

$$\psi_0^l = Ai^l(2l+1)j_l(kr)P_l(cos\theta) \tag{2.44}$$

Here,

$$j_l(x) = \frac{1}{2}\left[h_l^{(1)}(x) + h_l^{(2)}(x)\right] \approx \frac{1}{2x}\left[(-i)^{l+1}e^{ix} + i^{l+1}e^{-ix}\right] \quad , \quad for\ x \gg 1 \tag{2.45}$$

So, for large r :

$$\psi_0^l \approx A\frac{2l+1}{2ikr}\left[e^{ikr} - (-1)^{l+1}e^{-ikr}\right]P_l(cos\theta) \quad , \quad for\ V(r) = 0 \tag{2.46}$$

Here, the first term is the outgoing wave and the second term is the incoming wave. Now, when the potential V(r) is turned on, the incoming wave is unchanged and the outgoing wave picks up a total phase shift of $2\delta_l$ (δ_l for way in and δ_l for way out).

$$\psi^l = A\frac{2l+1}{2ikr}\left[e^{ikr+2\delta_l} - (-1)^{l+1}e^{-ikr}\right]P_l(cos\theta) \quad , \quad for\ V(r) = 0 \tag{2.47}$$

In the last section the whole theory was expressed in terms of partial wave amplitudes a_l. Here we have formulated everything in terms of the phase shift δ_l. The connection between a_l and δ_l can be found by comparing asymptotic (large r) equations for both the cases. In partial wave analysis formalism, we got :

$$\psi^l = A\left[e^{ikz} + ki^{l+1}(2l+1)a_l h_l'(kr)\right]P_l(cos\theta) \tag{2.48}$$

Expressing e^{ikz} in terms of incoming and outgoing waves and using the asymptotic form of $h_l'(kr)$, we get :

$$\psi^l = A\left[\frac{2l+1}{2ikr}(e^{ikr} - (-1)^l e^{-ikr}) + \frac{2l+1}{r}e^{ikr}a_l\right]P_l(cos\theta) \tag{2.49}$$

Comparing the total wave function in phase shift formalism Eq. (2.47) with the total wave function in partial wave analysis formalism Eq. (2.49), we get relation between a_l and δ_l :

$$a_l = \frac{e^{i\delta_l}}{k}sin(\delta_l) \tag{2.50}$$

2.4.4 Breit-Wigner Distribution

Previously we obtained scattering amplitude $f(\theta)$ in terms of partial wave amplitudes a_l. Expressing $f(\theta)$ in terms of phase shift δ_l using Eq. (2.50) :

$$f(\theta) = \sum_l (2l+1) a_l P_l(cos\theta) \tag{2.51}$$

$$\Rightarrow f(\theta) = \frac{1}{k} \sum_l (2l+1) e^{i\delta_l} sin(\delta_l) P_l(cos\theta) \tag{2.52}$$

The total cross-section (Eq. (2.37)) is given as :

$$\sigma = \frac{4\pi}{k^2} \sum_l^{\infty} (2l+1) sin^2(\delta_l) \tag{2.53}$$

$$or,\ \sigma = \frac{4\pi}{k^2} \sum_l^{\infty} (2l+1) \frac{1}{1+cot^2(\delta_l)} \tag{2.54}$$

Resonance occurs at maximum of σ or minimum of $cot^2(\delta_l)$, [7]. At $\delta_l = \pi/2$, $cot^2(\delta_l) = 0$, its minimum value. Cross-section (σ) is a function of energy. At the minimum σ point resonance occurs and the corresponding energy is called the resonance energy E_R. Taylor series expansion of $cot(\delta_l)(E)$ around $\pi/2$ or E_R is given by :

$$cot\delta_l(E) = cot\delta_l(E_R) + (E - E_R) \left(\frac{\partial cot\delta_l}{\partial E} \right)_{E=E_R} + \frac{1}{2}(E - E_R)^2 \left(\frac{\partial^2 cot\delta_l}{\partial E^2} \right)_{E=E_R} + ... \tag{2.55}$$

Here, $\left(\frac{\partial cot\delta_l}{\partial E}\right)_{E=E_R}$ = -$\left(\frac{\partial \delta_l}{\partial E}\right)_{E=E_R}$ and $cot\delta_l(E_R) = 0$. Neglecting the higher order terms we get :

$$cot\delta_l = -(E - E_R) \left(\frac{\partial \delta_l}{\partial E} \right)_{E=E_R} = -\frac{E - E_R}{A} \tag{2.56}$$

The cross-section for the 'l'th partial [7] wave becomes :

$$\sigma_l = \frac{4\pi}{k^2}(2l+1) \frac{1}{1+cot^2\delta_l} = \frac{4\pi}{k^2}(2l+1) \frac{A^2}{(E-E_R)^2 + A^2} \tag{2.57}$$

Previously we derived the cross-section distribution of resonance in Eq.(2.15). Comparing it with the above equation we get :

$$A = \frac{\Gamma}{2}, \qquad \sigma_{max} = \frac{4\pi}{k^2}(2l+1) \tag{2.58}$$

Hence, the final expression for the cross-section of a resonance as a function of energy is:

$$\sigma(E) = \frac{4\pi}{k^2}(2l+1)\frac{\Gamma^2/4}{(E-E_R)^2+\Gamma^2/4} \tag{2.59}$$

This is called Breit-Wigner distribution of Resonance cross-section. In case of decay of a resonance into its daughter particles, the invariant mass distribution of the resonance obtained from its daughter particles follows this Breit-Wigner distribution. At E = E_R, $\sigma(E_R) = \sigma_{max}$. At E = $E_R \pm \frac{\Gamma}{2}$, the cross-section becomes half ($\sigma(E_R \pm \Gamma/2) = \frac{\sigma_{max}}{2}$). Γ is the Full Width at Half Maximum (FWHM) of the Breit-Wigner distribution. This can be verified by plotting σ/σ_{max}, and finding the width at $\frac{\sigma}{\sigma_{max}} = 0.5$. The shape of the Breit-Wigner resonance curve is shown in Fig. 2.6.

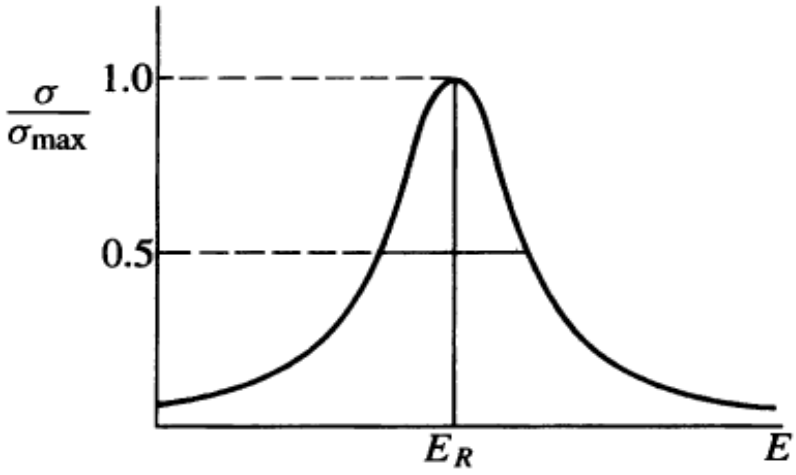

Figure 2.6: Normalised cross-section plot as function of energy following Breit-Wigner distribution

Chapter 3

K^{*0} Resonance Analysis Using AMPT Simulation

This chapter presents the invariant mass analysis of K^{*0} using AMPT simulation. First a brief description of AMPT is given. Then the complete analysis using AMPT simulation is discussed with results and plots. will be developed to explain this effect by studying the interaction of the daughter particles of K^{*0} - Kaons and Pions with the hadron gas medium giving rise to two important phenomena called Rescattering and Regeneration. The effect of these phenomena on the K^{*0} yield will be shown by simulating the model to obtain the K^{*0} yield as a function of hadronic cascade time τ_{HC}.

3.1 AMPT

To understand the extensive experimental results of the relativistic heavy ion collision experiments, many theoretical models have been introduced. They are broadly divided into three categories :

(i) Thermal Models - are based on the assumptions of global and thermal equilibrium.

(ii) Hydrodynamic Models - are based only on the assumption of local thermal

equilibrium,

(iii) Transport Models - treat nonequilibrium dynamics explicitly.

Thermal models have been very successful in accounting for the yield of various particles and their ratios, while the hydrodynamic models are particularly useful for understanding the collective behavior of low transverse momentum particles such as the elliptic flow. Since transport models treat chemical and thermal freeze-out dynamically, they are also natural and powerful tools for studying the Hanbury-Brown-Twiss interferometry of hadrons.

The dense matter created in heavy ion collisions at RHIC may not achieve full thermal or chemical equilibrium as a result of its finite volume and energy. To address such nonequilibrium many-body dynamics, a multiphase transport (AMPT) model [1], has been developed that includes both initial partonic and final hadronic interactions and the transition between these two phases of matter.

3.1.1 Structure of AMPT

The AMPT model consists of four main components: the initial conditions, partonic interactions, conversion from the partonic to the hadronic matter, and hadronic interactions. For each component different models are used as follows :

(i) Initial Conditions - The initial conditions, which include the spatial and momentum distributions of minijet partons and soft string excitations, are obtained from the Heavy Ion Jet Interaction Generator (HIJING) model.

(ii) Partonic Interactions - Scatterings among partons are modeled by Zhangs Parton Cascade (ZPC), which at present includes only two-body scatterings with cross sections obtained from the pQCD with screening masses.

(iii) Partonic to Hadronic conversion - The hadronization process is based on either the Lund string fragmentation model or on a quark coalescence model.

(iv) Hadronic interactions - Scatterings among the resulting hadrons are described by A Relativistic Transport (ART) model.

3.1.2 Types of AMPT - Default and String melting

Based on the two models used for the third stage that is partonic matter to hadronic matter conversion stage, two variants of the AMPT model are classified.

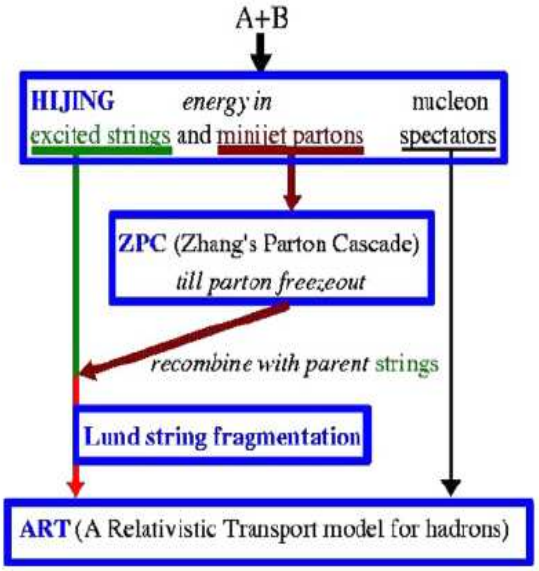

Figure 3.1: Default AMPT model

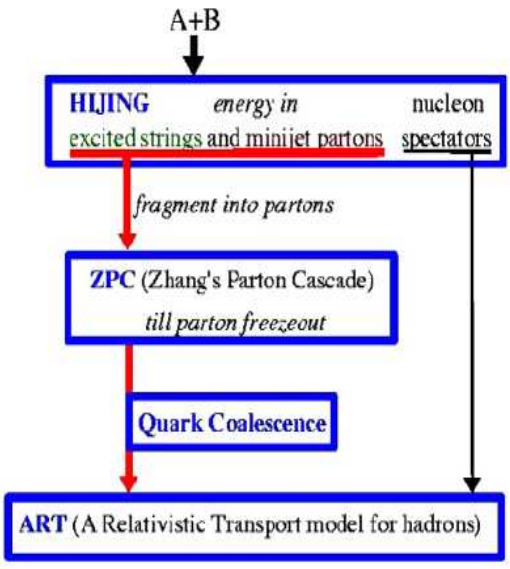

Figure 3.2: String Melting AMPT model

(a) Default AMPT Model -

In the Default AMPT model Partons are recombined with their parent strings when they stop interacting, and the resulting strings are converted to hadrons using the Lund string fragmentation model.

(b) String Melting AMPT Model (AMPT-SM) -

In the AMPT model with string melting, a quark coalescence model is used to combine partons into hadrons.

Since the default AMPT model reproduces the yield and transverse momentum spectra much better than AMPT-SM, we have chosen the AMPT default for our study. Final results are obtained after the termination of hadronic cascade. The termination time of hadronic cascade (τ_{HC}) can be tuned by an input parameter called NTMAX, which is the number of time steps (each step corresponds to 0.2 fm/c) of the hadronic interactions. The default value of NTMAX is 150 which corresponds to a value of τ_{HC} = 30 fm/c, while NTMAX = 3, corresponding to τ_{HC} = 0.6 fm/c effectively turns off the hadronic cascade. Fig. 3.1 and Fig. 3.2 show the structure of Default AMPT and AMPT with string melting model.

3.2 K^{*0} Resonance Analysis

3.2.1 K^{*0} Resonance

The production of resonances in heavy-ion collisions is expected to be sensitive to the properties of strongly interacting matter produced in these collisions. The resonance production may be affected by the onset of deconfined phase of quarks and gluons called the quark gluon plasma (QGP). The Large Hadron Collider (LHC) at CERN can provide collisions of heavy nuclei at center of mass energies up to 5.5 TeV per nucleon, where such a QGP can be formed. The resonance like $K^{*0}(892)$ meson is of particular interest because it has a small lifetime ($\sim$ 4 fm/c) compared to hadronic cascade time ($\sim$ 10 fm/c at LHC). So the characteristic properties of K^{*0} such as its mass, invariant mass distribution width and yield could be modified relative to systems where there is no QGP (like small systems of minimum bias pp collisions). Due to short lifetime, decay particles of resonances may undergo in-medium effects like rescattering and K^{*0} contains a strange quark, it may also provide some information regarding the strangeness enhancement for the system.

Properties

Isospin, I = $\frac{1}{2}$.

Angular Momentum, J = 1. So, its a vector meson.

Parity, P = -1.

Mass = 895.81 ± 0.19 MeV.

Decay Width (Γ) = 47.4 ± 0.6 MeV.

Decay Modes

The different decay channels and their branching ratios $\mathcal{B}$ of the K^* resonance as given in Particle Data Group (PDG) [2] are tabulated in Table 3.1.

Table 3.1: K^* resonance decay channels with their $\mathcal{B}$ as given in PDG

Decay Mode	Branching Ratio($\mathcal{B}$)
$K^{*0} \longrightarrow K\pi$	$\sim$1
$K^{*0} \longrightarrow K^0\gamma$	$2.46 \pm 0.21 \times 10^{-3}$
$K^{*0} \longrightarrow K^{\pm}\gamma$	$9.9 \pm 0.9 \times 10^{-4}$
$K^{*0} \longrightarrow K\pi\pi$	$< 7 \times 10^{-4}$

3.2.2 AMPT Analysis and Plots

Simulated events of Au-Au collision were generated using Default AMPT model for centre of mass energy 200 GeV ($\sqrt{s_{NN}} = 200GeV$). The hadronic cascade time was set to 0.6 fm/c by fixing the value of the parameter NTMAX = 3. The data was generated for 50000 events. The output of the AMPT were text files containing the following information of all the produced particles - particle identification, mass, position coordinates, momentum coordinates. We generated a root file containing a tree with the information from all the text files arranged in the branches of a tree. The root file was then used for further analysis. The complete codes in ROOT Macros are given in Macro-1 and Macro-2 of APPENDIX-I.

Unlike Sign Invariant Mass Distribution

Out of all the produced particles, the Kaons ($K^{\pm}$) and Pions ($\pi^{\pm}$) were selected from the root file using their particle identification numbers and their properties like position coordinates and momentum coordinates were extracted. The PID as given in Particle Data Group (PDG) :

$$K^+ = 321, \;\; K^- = -321, \;\; \pi^+ = 211, \;\; \pi^- = -211 \tag{3.1}$$

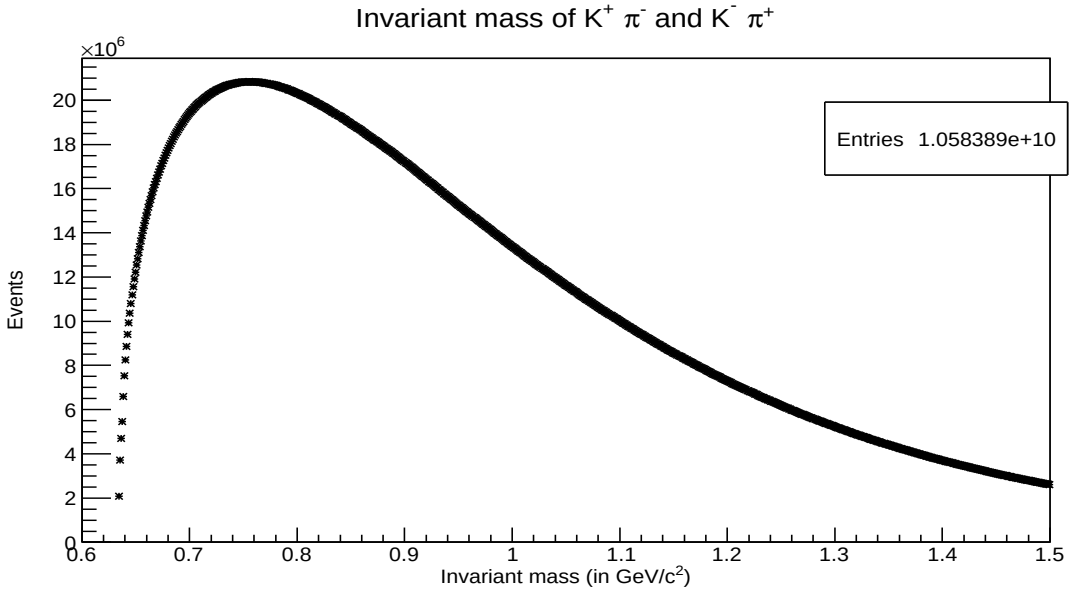

Figure 3.3: K^{*0} invariant mass distribution obtained from $K^+\pi^-$ and $K^-\pi^+$ (Unlike sign distribution)

Energy of these Kaons and Pions are found from their mass and momentum using the following formula :-

$$E = \sqrt{p_x^2 + p_y^2 + p_z^2 + m^2} \tag{3.2}$$

The two decay channels for K^{*0} resonance are as follows :-

$$K^{*0} \longrightarrow K^+\pi^- \ \ and \ \ \overline{K^{*0}} \longrightarrow K^-\pi^+ \tag{3.3}$$

The invariant mass of the K^{*0} is calculated from the above two decay channels using the energies and momenta of the Kaons and Pions as follows :

$$M_{K^{*0}} = \sqrt{[E_{K^+} + E_{\pi^-}]^2 - [(p_{x_{K^+}} + p_{x_{\pi^-}})^2 + (p_{y_{K^+}} + p_{y_{\pi^-}})^2 + (p_{z_{K^+}} + p_{z_{\pi^-}})^2]} \tag{3.4}$$

$$M_{\overline{K^{*0}}} = \sqrt{[E_{K^-} + E_{\pi^+}]^2 - [(p_{x_{K^-}} + p_{x_{\pi^+}})^2 + (p_{y_{K^-}} + p_{y_{\pi^+}})^2 + (p_{z_{K^-}} + p_{z_{\pi^+}})^2]} \tag{3.5}$$

Histogram of invariant mass of K^{*0} is constructed. As in these two cases, sign of Kaons and Pions are opposite, so this is called Unlike sign distribution. The plot of the unlike sign invariant mass distribution is shown in Fig. 3.3.

Here, we dont get a signal peaking at K^{*0} mass. So, we expect that there is some background which we need to remove to get a clear signal.

Like Sign Background

Invariant mass of $K^+\pi^+$ and $K^-\pi^-$ are calculated using the energies and momentum of Kaons and Pions as follows :-

$$M = \sqrt{[E_{K^+} + E_{\pi^+}]^2 - [(p_{x_{K^+}} + p_{x_{\pi^+}})^2 + (p_{y_{K^+}} + p_{y_{\pi^+}})^2 + (p_{z_{K^+}} + p_{z_{\pi^+}})^2]} \quad (3.6)$$

$$M = \sqrt{[E_{K^-} + E_{\pi^-}]^2 - [(p_{x_{K^-}} + p_{x_{\pi^-}})^2 + (p_{y_{K^-}} + p_{y_{\pi^-}})^2 + (p_{z_{K^-}} + p_{z_{\pi^-}})^2]} \quad (3.7)$$

Histogram of invariant mass is constructed. As in these two cases, sign of Kaons and Pions are same, so this is called Like sign distribution. The plot of the like sign invariant mass distribution is shown in Fig. 3.4. The complete codes for unlike and like distribution are given in Macro-3 of APPENDIX-I.

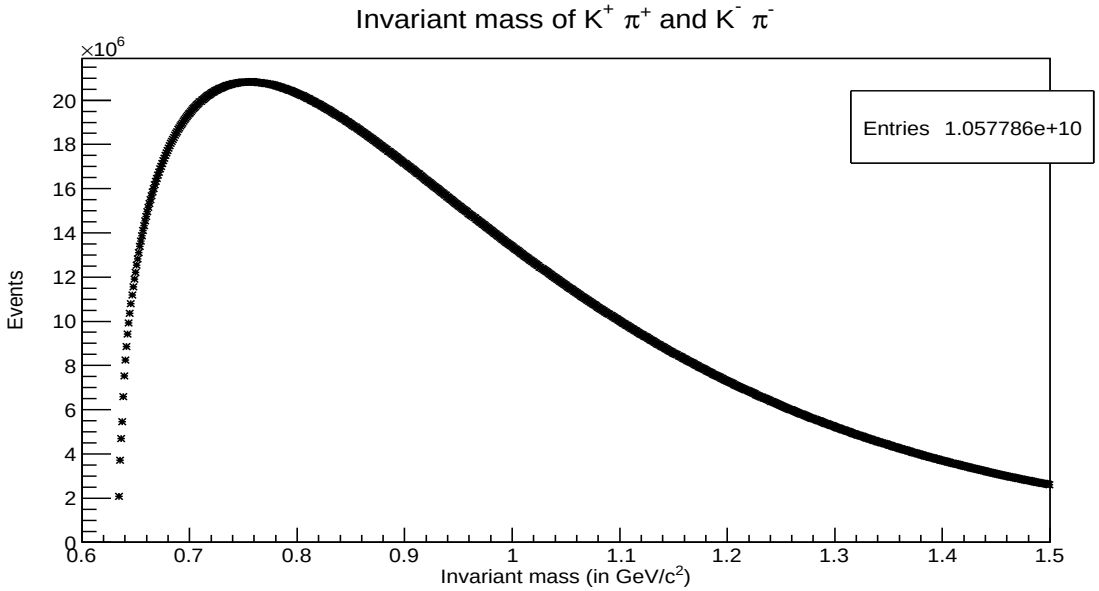

Figure 3.4: Invariant mass distribution obtained from $K^+\pi^+$ and $K^-\pi^-$ (Like sign distribution)

K^{*0} Signal Distribution

As observed in the Unlike sign invariant mass distribution of K and π, we do not get a K^{*0} peak. The invariant mass reconstruction of all the $K^+\pi^-$ and $K^-\pi^+$ pairs do not give the correct result and is dominated by background events. Along with the daughter $K\pi$ pairs, it also involves the background combinations of $K\pi$ pairs not originating from K^{*0} decay. To observe the signal we need to subtract the statistical or combinatorial background from the same event $K\pi$ pairs distribution. The combinatorial background is estimated using like sign method. In this method the background is the Like sign invariant mass distribution. It is subtracted from the Unlike sign invariant mass distribution to get the signal distribution as shown in the Fig. 3.5.

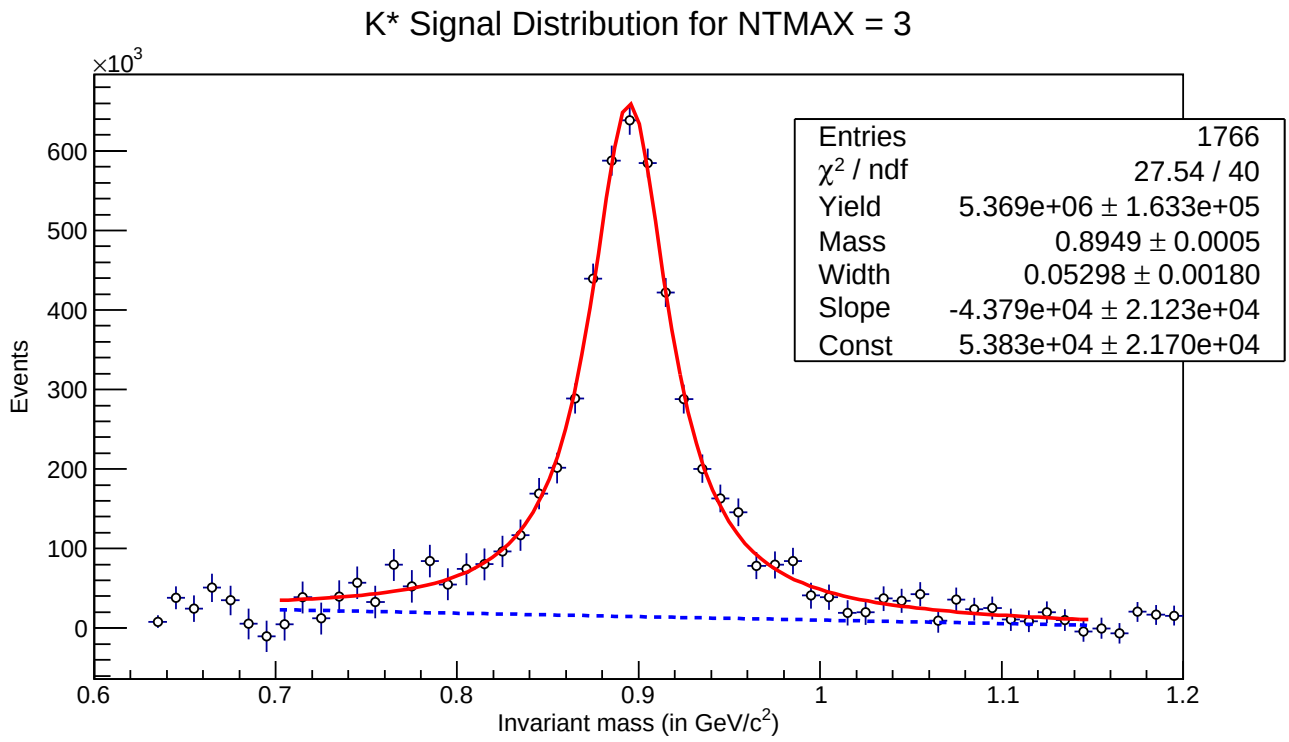

Figure 3.5: K^{*0} Signal distribution fitted with Breit-Wigner + linear background function for $\tau_{HC} = 0.6$ fm/c

Table 3.2: Fit parameters of AMPT analysis of K^{*0} invariant mass

Parameter	Value
Yield	$5.368 \times 10^6 \pm 163261$
Mass	0.894861 ± 0.000545235 MeV/c^2
Width	0.0529776 ± 0.00179965 MeV/c^2
Linear Bkg Slope	-43791.2 ± 21232.2 $(MeV/c^2)^{-1}$
Linear Bkg Intercept	53826.6 ± 21703

The signal is fitted with a combination of Breit-Wigner function and linear background to get the Mass, Width and Yield of K^{*0}. The fit parameters are tabulated in Table 3.2. The complete code is given in Macro-4 of APPENDIX-I.

Dependence of Yield on Hadronic cascade time

The whole analysis is repeated for NTMAX = 5, 10, 20, 30 and 40 values corresponding to hadronic cascade time, τ_{HC} = 1, 2, 4, 6 and 8 fm/c as each time step corresponds to 0.2 fm/c. The K^{*0} signal distribution is obtained for each case and fitted with a combination Breit-Wigner function and linear background function to obtain the corresponding K^{*0} yield, mass and width. A comparison plot of the signal invariant mass distribution is shown in Fig. 3.6.

The termination time of the hadronic cascade time is varied from 1 to 8 fm/c (shown by different coloured lines). The number of events in each configuration are kept same i.e. 50000 in order to make a proper comparison. It is clearly observed that the invariant mass signal decreases with increase in hadronic cascade time. An increase in hadronic cascade means there is more time for K^{*0} and its daughter particles to interact with the medium which leads to increase in hadronic re-scattering. It is expected because of the change in momentum of the daughters of K^{*0} - π and K, due to rescattering. In the next section we will try to develop a theoretical model

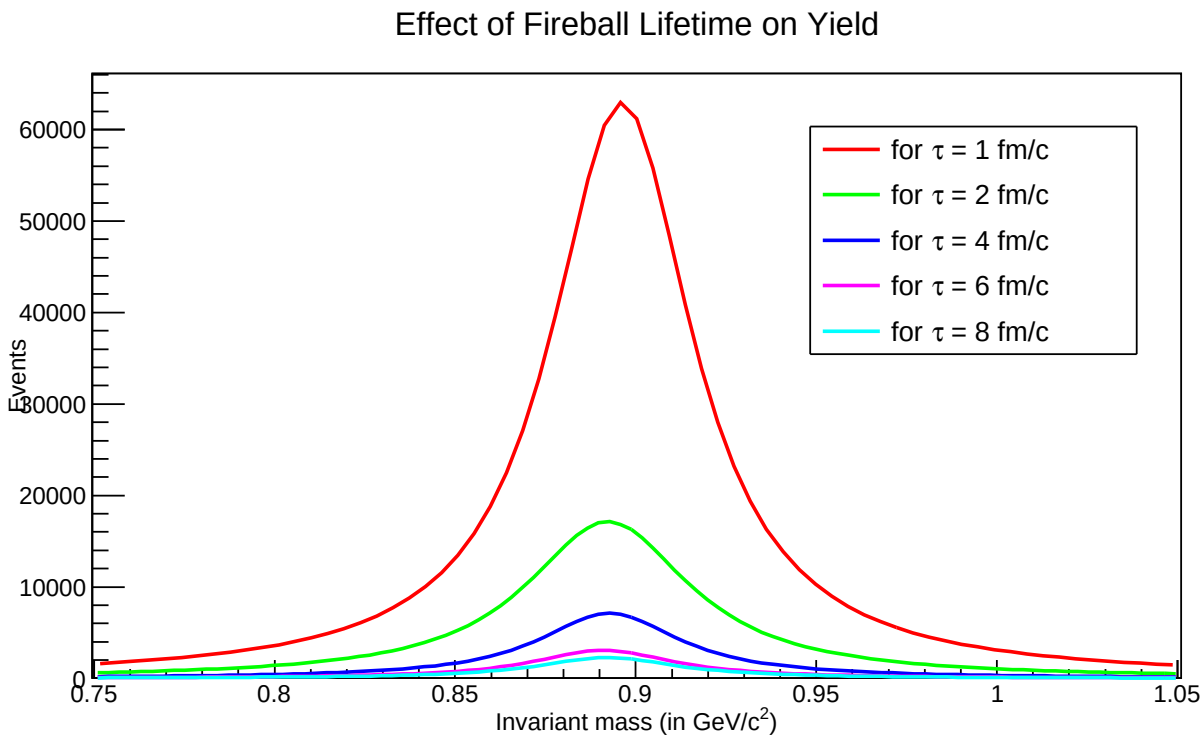

Figure 3.6: K^{*0} invariant mass distributions for different τ_{HC}

to explain this result. The complete code is given in Macro-5 of APPENDIX-I.

3.3 Systematics of Resonance Production - Rescattering and Regeneration

As observed in the AMPT results, the yield of K^{*0} decreases with the increasing hadronic cascade time. To explain this result, we try to study the systematics of K^{*0} production in the fireball and analyse the effect of two important phenomena - Rescattering and Regeneration, occurring within the hadron gas which shape the final K^{*0} yield.

Stable hadrons like π, K, and p are directly observed in heavy ion collision experiments. A few more particles like Λ, Ξ and Ω are also observed by reconstruction from their decay products. Since the lifetime of the hadron gas is $\mathcal{O} \sim 10^{-22}$ seconds which is negligible compared to that of Λ, Ξ and Ω whose lifetime is $\mathcal{O} \sim 10^{-10}$ seconds, they decay while free streaming to the detector. Thus one expects accurate reconstruction for these hadrons. Problem arises in the case of resonances like K^{*0}, Σ^*, Δ^* etc. Their lifetime is typically of the same order as the hadron gas lifetime and hence a fraction of them can decay inside the hadron gas itself. For example in case of K^{*0}, the daughter particles - $K\pi$ could undergo elastic scattering with the medium particles and loose their energy or momentum. We can not reconstruct back the resonance which results in loss in K^{*0} yield. This phenomenon is called "Rescattering". On the other hand, some of the pions and kaons present in the medium can re-generate K^{*0} via pseudo-elastic interactions ($K\pi \rightarrow K^{*0} \rightarrow K\pi$) which results in gain in K^{*0} yield. This is known as "Regeneration". Thus an estimation of resonance yield requires in addition to determination of the chemical freeze-out (CFO) surface, a correct modeling of the late stage fireball expansion between chemical and kinetic freeze-out (KFO) of the interacting hadron medium, called the hadron gas evolution in which elastic and pseudo-elastic collisions are still on. Thus, while the analysis of

stable hadron yields provide us an opportunity to comprehend the thermodynamic state of the fireball at the time of CFO, it is expected that by analysing the yield of resonances one can learn a lot about the hadron gas evolution and the hadronic medium properties between CFO and KFO.

3.3.1 Kinetics and Rate Equations

Let us consider a resonance R that decays into the daughter hadrons A and B with decay width Γ. Now both A and B can interact with the thermal particles that are already present in the hadron gas and scatter elastically that will result in a loss of signal for the parent resonance R. This is rescattering. In case there is a possibility that the daughter particles interact with the medium particles through a pseudo elastic channel to give back R, then this will result in partial recovery of the lost signal. The kinetic equations can be written as follows :

$$R \xrightarrow{k_1} A + B \tag{3.8}$$

$$A \xrightarrow{k_2} A' \tag{3.9}$$

$$A \xrightarrow{k_3} R \tag{3.10}$$

$$B \xrightarrow{k_4} B' \tag{3.11}$$

$$B \xrightarrow{k_5} R \tag{3.12}$$

Here, Eq. (3.8) is the decay of R, Eq. (3.9) and Eq. (3.11) are the rescattering and Eq. (3.10) and Eq. (3.12) are the regeneration processes. Here A' and B' refer to the state of the daughter hadrons A and B respectively due to elastic collision and hence can not yield R by invariant mass reconstruction. The different reaction rates are referred to as k_i, i running from 1 to 5. Denoting the number density of each species of particles as N_i where i can be R (for resonance), A and B (for the daughters), we write down the differential equations governing the rate of change of

N_i from the above kinetic equations :

$$\frac{1}{N_R}\frac{dN_R}{dt} + k_1 - k_3\left(\frac{N_A}{N_R}\right) - k_5\left(\frac{N_B}{N_R}\right) = 0 \tag{3.13}$$

$$\frac{d}{dt}\left(\frac{N_A}{N_R}\right) + (k_2 + k_3 - k_1)\left(\frac{N_A}{N_R}\right) + k_3\left(\frac{N_A}{N_R}\right)^2 + k_5\left(\frac{N_A}{N_R}\right)\left(\frac{N_B}{N_R}\right) = 0 \tag{3.14}$$

$$\frac{d}{dt}\left(\frac{N_B}{N_R}\right) + (k_4 + k_5 - k_1)\left(\frac{N_B}{N_R}\right) + k_5\left(\frac{N_B}{N_R}\right)^2 + k_3\left(\frac{N_A}{N_R}\right)\left(\frac{N_B}{N_R}\right) = 0 \tag{3.15}$$

Assuming $\left(\frac{N_A}{N_R}\right)^2 \sim \left(\frac{N_B}{N_R}\right)^2 \sim \left(\frac{N_A}{N_R}\right)\left(\frac{N_B}{N_R}\right) \ll 1$, then Eq. (3.14) and Eq. (3.15) become :

$$\frac{d}{dt}\left(\frac{N_A}{N_R}\right) + k_A\left(\frac{N_A}{N_R}\right) - k_1 = 0 \tag{3.16}$$

$$\frac{d}{dt}\left(\frac{N_B}{N_R}\right) + k_B\left(\frac{N_B}{N_R}\right) - k_1 = 0 \tag{3.17}$$

where, $k_A = k_2 + k_3 - k_1$ and $k_B = k_4 + k_5 - k_1$. Solving for $\left(\frac{N_A}{N_R}\right)$ and $\left(\frac{N_B}{N_R}\right)$ from the equations, we get :

$$\left(\frac{N_A}{N_R}\right)_{t_2} = \left(\left(\frac{N_A}{N_R}\right)_{t_1} - \left(\frac{k_1}{k_A}\right)\right) exp(-k_A(t_2 - t_1)) + \left(\frac{k_1}{k_A}\right) \tag{3.18}$$

$$\left(\frac{N_B}{N_R}\right)_{t_2} = \left(\left(\frac{N_B}{N_R}\right)_{t_1} - \left(\frac{k_1}{k_B}\right)\right) exp(-k_B(t_2 - t_1)) + \left(\frac{k_1}{k_B}\right) \tag{3.19}$$

Putting Eq. (3.18) and Eq. (3.19) in Eq. (3.13), and solving for N_R we get :

$$\begin{aligned} N_R(t_2) =& N_R(t_1) exp\,[\,(-k_1 + k_3 N_{A_2} + k_5 N_{B_2}(t_2 - t_1) \\ &- \frac{k_3}{k_A} N_{A_1}(exp(-k_A t_2) - exp(-k_A t_1)) \\ &- \frac{k_5}{k_B} N_{B_1}(exp(-k_B t_2) - exp(-k_B t_1))] \end{aligned} \tag{3.20}$$

where,

$$N_{A_1} = \left(\left(\frac{N_A}{N_R} \right)_{t_1} - \left(\frac{k_1}{k_A} \right) \right) exp(k_A t_1)) \tag{3.21}$$

$$N_{A_2} = \left(\frac{k_1}{k_A} \right) \tag{3.22}$$

$$N_{B_1} = \left(\left(\frac{N_B}{N_R} \right)_{t_1} - \left(\frac{k_1}{k_B} \right) \right) exp(k_B t_1)) \tag{3.23}$$

$$N_{B_2} = \left(\frac{k_1}{k_B} \right) \tag{3.24}$$

3.3.2 Time dependence of Reaction rates

The reaction rates k_i's were assumed to be time-independent in the previous section. But in heavy ion collisions, within the fireball the time-dependency of the reaction rates should be considered([3], [4]). Here, we will try to model this and incorporate the time-dependence into the reaction rates.

First consider our desired decay of the resonance :

$$K^{*0} \longrightarrow K\pi \tag{3.25}$$

We assume that both rescattering and regeneration as described previously can take place. The decay products travel through the medium with speed v_i (where i can mean either K or π). The interaction probability is proportional to v_i,the interaction cross-section of the decay product with each particle in the hadronic medium $\sigma_{ij}(v_i)$(where j can refer to either pions, Kaons, nucleons or antinucleons), and the particle density of the fireball (ρ_j). At relativistic speed due to Lorentz contraction, the volume of the system decreases which increases ρ_j by the Lorentz factor, $\gamma_i = \frac{1}{\sqrt{1-v_i^2}}$. But ρ_j also decreases due to the collective expansion of the fireball which is parametrized by the flow velocity v_{flow}, assumed to be of the order of the relativistic speed of sound ($c/\sqrt{3}$). Taking into account both the factors, the time

dependence of the densities as given in [3] is given as :

$$\rho_j(t) = \gamma_i \rho_{0j} \left(\frac{R}{R + v_{flow} t} \right)^3 \tag{3.26}$$

where, 'R' is the fireball radius and ρ_{0j} is the density of j at hadronisation, which can be calculated from chemical freeze-out temperature (T) and chemical potentials (μ_B). Now, the time dependent reaction rates are given by the interaction probabilities :

$$k_i(t) = P_i(t) = \sigma_{ij} \rho_j(t) v_i \tag{3.27}$$

$$= \sum_{v_i} [\sigma_{i\pi} \rho_{0\pi} + \sigma_{iK} \rho_{0K} + \sigma_{iN} \rho_{0N} + \sigma_{i\bar{N}} \rho_{0\bar{N}}] \gamma_i v_i \left(\frac{R}{R + v_{flow} t} \right)^3 \tag{3.28}$$

Now, using average as :

$$\sum_{v_i} \sigma(v_i) v_i \gamma_i \simeq < \sigma > < v_i \gamma_i > = < \sigma > \frac{p_i}{m_i} \tag{3.29}$$

So, the reaction rates become :-

$$k_i(t) = [< \sigma_{i\pi} > \rho_{0\pi} + < \sigma_{iK} > \rho_{0K} + < \sigma_{iN>} \rho_{0N} + < \sigma_{i\bar{N}} > \rho_{0\bar{N}}] \frac{p_i}{m_i} \left(\frac{R}{R + v_{flow} t} \right)^3 \tag{3.30}$$

Here, 'i' can be Kaon or pion. The required density at hadronization (ρ_{0j}) [3] is obtained through :

$$\rho_{0j} = \frac{g}{(2\pi\hbar c)^3} 4\pi m^2 (\lambda_q \gamma_q)^3 T K_2 (\frac{m}{T}) \tag{3.31}$$

3.3.3 Analysis

The cross-section of different elastic and pseudo-elastic interactions are given in Table 3.3.

Here the conversion factor of $1\ mbarn = 25 \times 10^{-7}\ MeV^{-2}$ is used. The inelastic cross-sections σ_{KN} and $\sigma_{\pi N}$ are both zero. The densities are calculated as :

$$\rho_{0j} = \frac{g}{2\pi^2} m^2 T K_2 (\frac{m}{T}) \tag{3.32}$$

Table 3.3: Cross-sections of interactions between K and π

Type	Cross-section	Value in MeV^{-2}
Elastic	σ_{KK}	$20 \times 25 \times 10^{-7}$
Elastic	$\sigma_{K\pi}$	$20 \times 25 \times 10^{-7}$
Elastic	σ_{KN}	$40 \times 25 \times 10^{-7}$
Elastic	$\sigma_{\pi\pi}$	$20 \times 25 \times 10^{-7}$
Elastic	$\sigma_{\pi K}$	$20 \times 25 \times 10^{-7}$
Elastic	$\sigma_{\pi N}$	$40 \times 25 \times 10^{-7}$
Inelastic	$\sigma_{\pi K}$	$20 \times 25 \times 10^{-7}$
Inelastic	$\sigma_{\pi K}$	$20 \times 25 \times 10^{-7}$

where, $K_2(m/T)$ is the second Bessel function.

The other input parameters used in the code are as follows :

Volume at Chemical freeze-out, $V_{CFO} = 5 \times 10^{-4}$ MeV.

Temperature at Chemical freeze-out, $T_{CFO} = 155$ MeV.

The fireball radius can be calculated using :

$$R = \left(\frac{V_{CFO}}{4/3\pi}\right)^{1/3} \tag{3.33}$$

Using these cross-section values, input values and all the equations derived in the previous sections, we modeled a Mathematica code to compute the normalised yield of K^{*0} ($N_R(t)$), daughter Pions ($N_A(t)$) and daughter Kaons ($N_A(t)$), (normalised with the K^{*0} yield at Chemical Freeze-out) as a function of the time elapsed after Chemical Freeze-out (τ). The algorithm of the code is as follows :

- First the densities at hadronisation are calculated for K and π using Eq.(3.32).
- Using Eq.(3.30) the time dependent reaction rates are calculated.
- The initial particle yields are divided by the K^{*0} yield at hadronisation. So, initial values are : N(K^{*0}) =1, N(K)=0 and N(π)=0. These values are actually normalised yields or ratios.

- Using Eq.(3.18) to (3.24) and looping over time interval the K^{*0}, K and π ratios at different τ are calculated.

The full Mathematica code is given in "Mathematica Code-1" of APPENDIX-II. These values are plotted and the plot is shown in Fig. 3.7. The full ROOT code is given in "Macro-1" of APPENDIX-II.

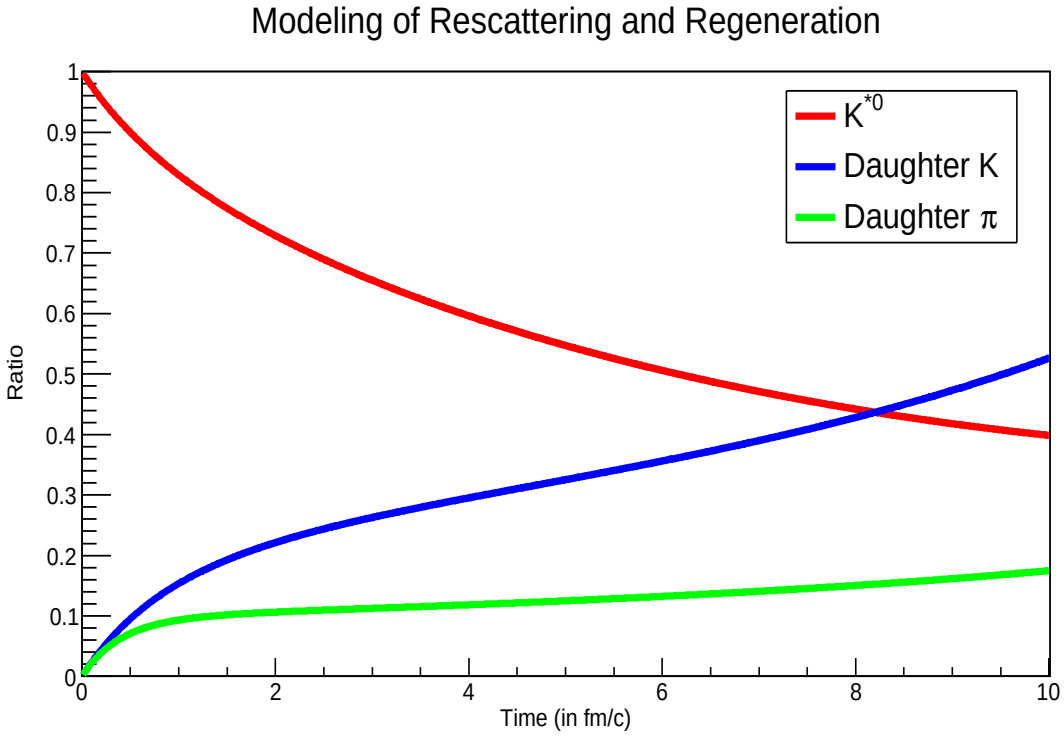

Figure 3.7: Modeling of rescattering and regenration to get K^{*0}, K and π ratios

As seen in the plot, with higher τ, K^{*0} number ratio decreases and K and π number ratios increases. Also it is observed that at some later τ, Kaon yield is higher than Pion ratio. The reason is π has lower mass, so higher speed and the interaction probability derived in Eq.(3.28) is proportional to speed, which means π interaction is more than K. Hence, π yield is less than K yield.

In AMPT simulations , the hadronic cascade time or the hadron gas lifetime (τ_{HC})

is controlled using a parameter NTMAX. With higher values of this parameter, τ_{HC} is higher which leads to higher rescattering. This fall in the K^{*0} number ratio as a result of rescattering and regeneration explains the effect of these in-medium effects on the decrease in K^{*0} yield with τ_{HC} observed in AMPT invariant mass distributions.

But in experiments there is no such parameter to control τ_{HC}. And τ_{HC} has to be calculated from the experimental data. The significance of this model is that using this model and this plot we can calculate τ_{HC} which is the time interval between Chemical and Kinetic Freeze-outs. From the experimental data we can find the K^{*0} number ratios at different centralities and compare that value with our model plot to get the corresponding τ_{HC}. And also the accuracy of the model can be analysed from the experimental data.

Chapter 4

K^{*0} Resonance production in U-U collision at STAR

This chapter presents the results of K^{*0} resonance production in U-U collisions at $\sqrt{s_{NN}}$ = 193GeV at STAR. Specifically we present different variables obtained from the collision data and the invariant mass reconstruction of neutral K^* meson and its derived observables to address the physical mechanism behind the resonance production.

4.1 Introduction

In the last sections we studied the production and properties of K^{*0} resonance using AMPT simulated data. The algorithm for invariant mass reconstruction was developed in ROOT Macros. The next step is to analyse experimental data from STAR to find the actual invariant mass and properties of K^{*0}. The analysis of experimental data is far too complicated than analysing the simulated data and their results can be very different. In simulated data we know the PID of all the particles produced in the collision simulation but in experimental data we have to identify the particles using the energy loss of the tracks and the estimated mass of the tracks which are prone to errors and lead to mis-identification of particles.

The production of resonances in heavy-ion collisions is expected to be sensitive to the properties of strongly interacting matter [1] produced in these collisions. The resonance production may be affected by the onset of deconfined phase of quarks and gluons called the quark gluon plasma (QGP). The importance of the resonance like K^{*0} meson is that it has a small lifetime ($\sim$ 4 fm/c) compared to the one of the hadron gas ($\sim$ 10 fm/c at LHC). So the characteristic properties of K^{*0} such as its mass, invariant mass distribution width and yield could be modified relative to systems where there is no QGP (like small systems of minimum bias pp collisions). Due to short lifetime, decay particles of resonances may undergo rescattering and re-generation effects. These are the two most important process which a typical resonance could undergo. Study of K^{*0} resonance and its properties can provide a lot of information on the hadron gas evolution and the hadronic medium properties between CFO and KFO.

First we will analyse important variables in the collision data. Then we will define the event and track selection cuts used and the cuts used for particle identification. Finally the invariant mass of K^{*0} resonance will be reconstructed and fitted with Breit-Wigner function to get the yield, mass and width of K^{*0}.

4.2 Data Set - Event and Track Variables

The analysis is done from the collision data in U-U collisions at $\sqrt{s_{NN}}$ = 193GeV recorded by the STAR detector. We are using the Time Projection Chamber (TPC) and the Time Of Flight (TOF) detectors of the STAR to carry out the analysis.

The offline data is converted into NTuples in ROOT which organises the raw data to give valuable information on the event variables and track variables. Plots of the important event and track variables as obtained from the data are given below.

4.2.1 Event Variables

Events are the particular collisions. One event contains all the information on all the particles produced in that particular collision. Some important event variables are :

1. Reference Multiplicity :

Reference multiplicity of an event gives the number of tracks in that event lying in the pseudorapidity range of $-0.5 < \eta < 0.5$. The distribution of reference multiplicity is used to define the centrality. The plot of the reference multiplicity distribution is shown in Fig. 4.1

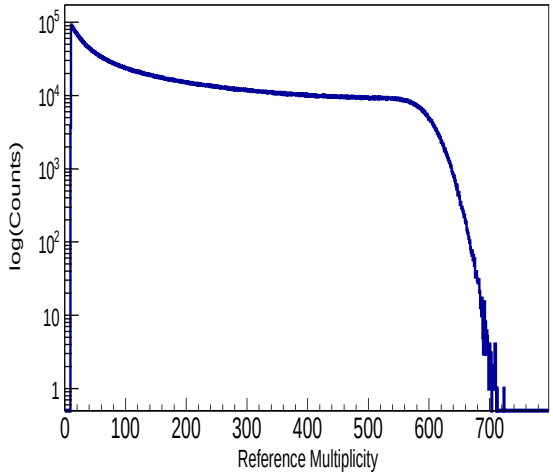

Figure 4.1: Reference Multiplicity distribution

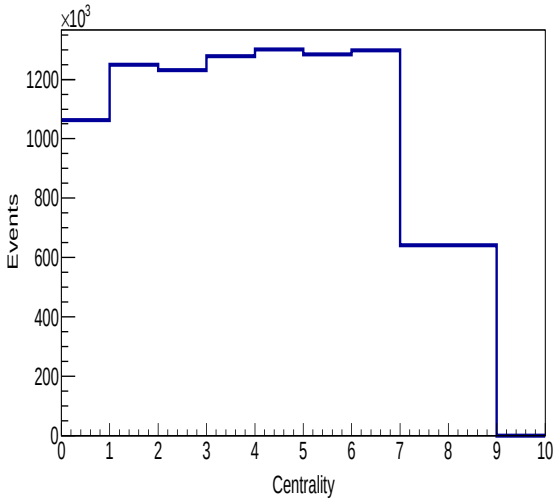

Figure 4.2: Centrality distribution

2. Centrality :

Centrality of a collision gives information on how close the two bunches of nuclei were while colliding. It has a very small value for head-on collision and a large value for peripheral collision. It is directly dependent on the impact parameter which is the distance between the center of the two colliding bunches. As the impact parameter is unknown in the experiment, so direct determination of the centrality of the event

is not possible. It is estimated using the reference multiplicity distribution. The plot of the centrality distribution is shown in Fig. 4.2

3. Vertex :

It gives the position of the primary vertex of the collision which is the point where the two nuclei collide. The plot of the Z-position of the vertex and the 2D plot of the X and Y-positions are shown in Fig 4.3 and Fig. 4.4.

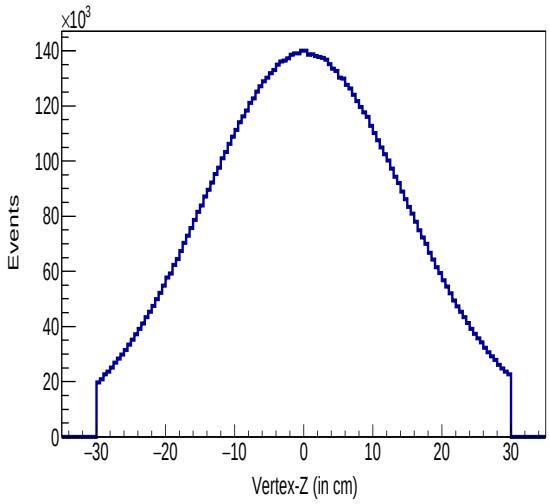

Figure 4.3: Vertex-Z distribution

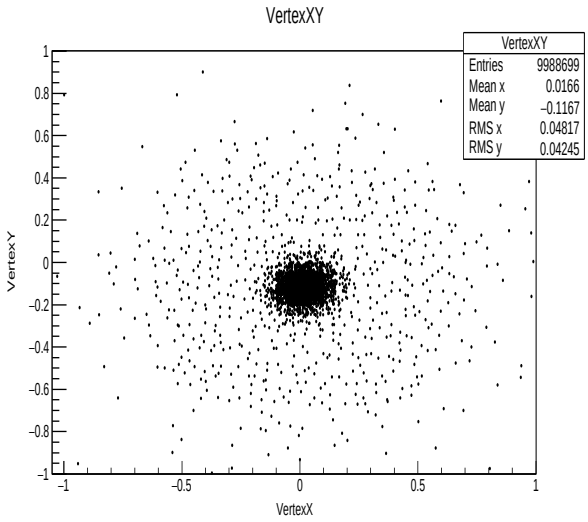

Figure 4.4: Vertex-X vs Vertex-Y distribution

4.2.2 Track Variables

Track variables show the properties of all the tracks present in a particular event. Some important track variables are discussed below. All the plots shown are for charged pion, kaon and proton tracks. **1. Momentum :**

It covers the momentum coordinates (p_x, p_y, p_z) and the total momentum (p) of the track given by :

$$p = \sqrt{p_x^2 + p_y^2 + p_z^2} \tag{4.1}$$

The distributions of p_x, p_y, p_z are shown in Fig. 4.5 , Fig. 4.6 and Fig. 4.7. There is a dip in the distributions of p_x and p_y at value 0, but no such dip in the distribution of p_z. The reason due to the cuts on the transverse momentum p_T which is discussed next.

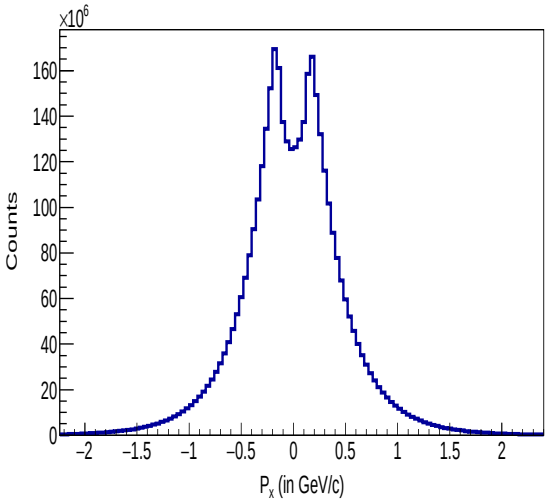

Figure 4.5: p_x distribution

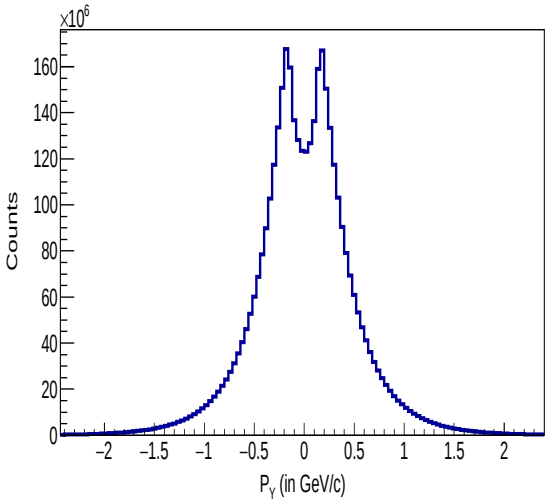

Figure 4.6: p_y distribution

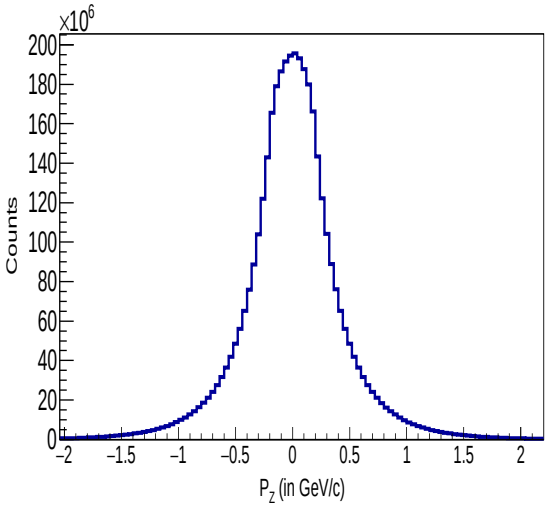

Figure 4.7: p_z distribution

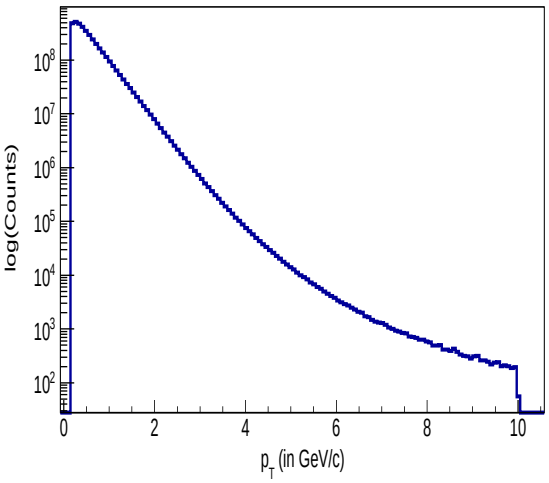

Figure 4.8: p_T distribution

2. Transverse Momentum :

Transverse momentum is momentum of the track in the transverse plane (xy-plane) which is the plane perpendicular to the beam direction (z-axis). It is given by :

$$p_T = \sqrt{p_x^2 + p_y^2} \tag{4.2}$$

The log scale plot of p_x and p_y is shown in Fig 4.8. It is clearly visible in the log plot that p_T is restricted to lie between 0.15 and 10 GeV/c using a selection cut on the tracks. So, the tracks which have p_x and p_y both ~ 0 have been removed so there is a dip in the p_x and p_y distributions at 0. But still the count is non-zero, because the tracks with $p_x \sim 0$ and $p_y \sim 0.15$ Gev/c and vice-versa satisfy the minimum p_T criteria and are not rejected. There is no restriction on p_z being 0, so there is no such dip in p_z distribution.

3. Pseudorapidity

Pseudorapidity (η), is a commonly used spatial coordinate describing the angle of a particle relative to the beam axis. It is given by :

$$\eta = \frac{1}{2} ln \left(\frac{|p| + p_z}{|p| - p_z} \right) = tan^{-1} \left(\frac{p_z}{|p|} \right) \tag{4.3}$$

The η values related to beam angle is shown in Fig. 4.9. The plot of η distribution is shown in Fig. 4.10. In the distribution we can see a dip around value 0. This is due to the finite mass of the particles. For massless photons there will be no dip in the η distribution.

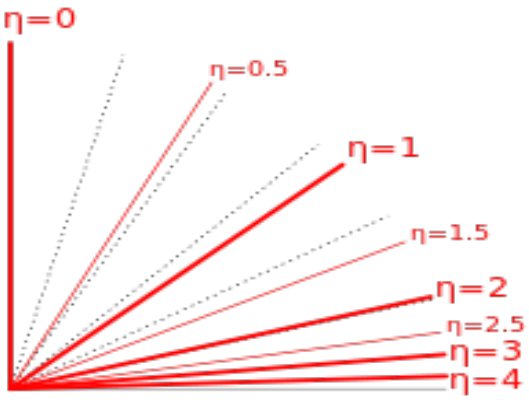

Figure 4.9: η Values

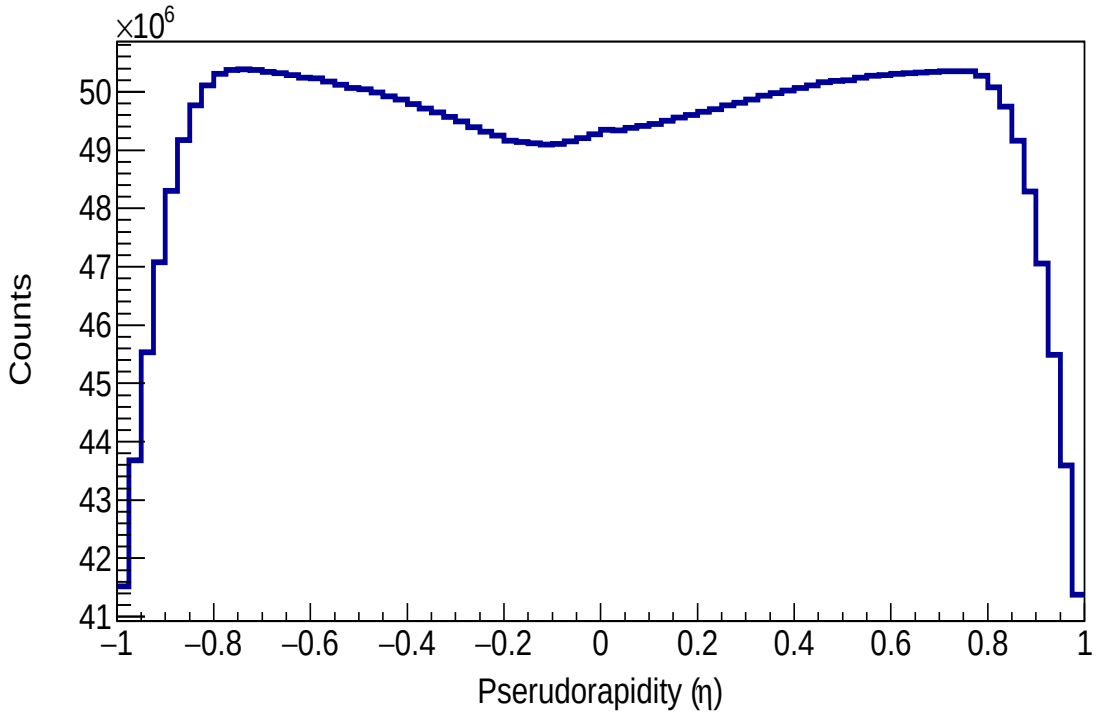

Figure 4.10: η distribution

4. Energy loss

It is the energy lost by the track in the TPC detector. The energy losss of a charged

particle is given by Bethe-Bloch formula :

$$-\frac{dE}{dx} = Kz^2\frac{Z}{A}\frac{1}{\beta^2}\left[\frac{1}{2}lnf(\beta) - \beta^2 - \frac{\delta(\beta\gamma)}{2}\right] \tag{4.4}$$

A plot of energy loss vs momentum is shown in Fig. 4.11. Three bands corresponding to pion, kaon and proton are seen in the plot.

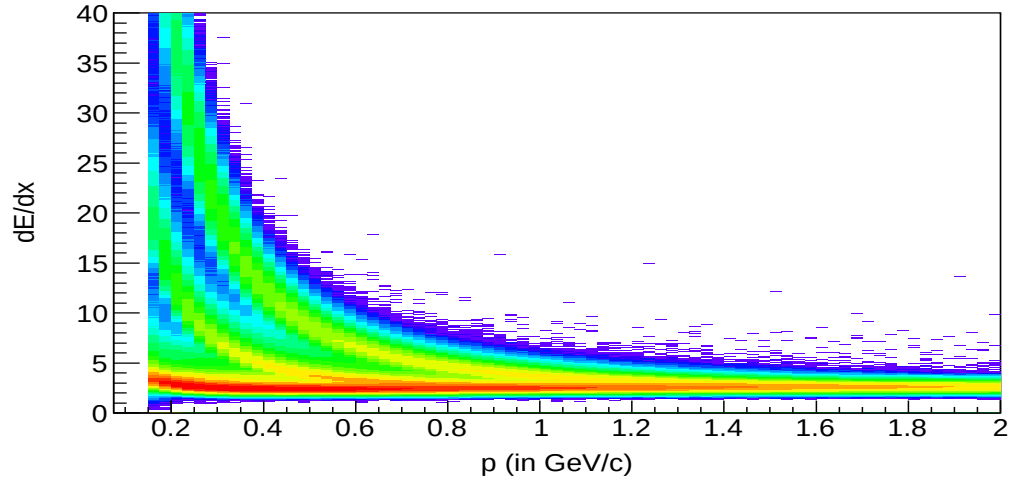

Figure 4.11: dE/dx vs p distribution

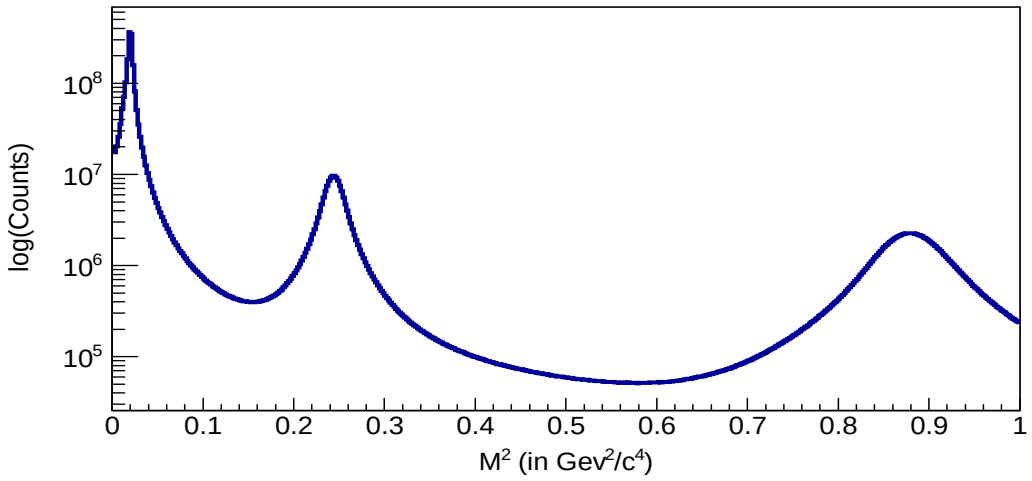

Figure 4.12: m^2 distribution

5. Mass :

The mass of the track is obtained using the TOF detector given by :

$$m^2 = \frac{p^2}{\beta^2 - 1} \tag{4.5}$$

where $\beta = v/c$. The plot of m^2 distribution is shown in Fig. 4.12. There are three peaks corresponding to pion, kaon and proton seen in the plot.

4.3 Event and Track Selection

The heavy ion collision process produces a lot of events with numerous tracks. To improve the quality of the analysis we select good quality events and tracks by using some cuts on the variables described above which are called selection cuts.

Event Cuts

Table 4.1: Event cuts for K^{*0} invariant mass analysis

Cut name	Cut value
V_Z range	-30 < V_Z <30 cm
V_r range	-2 < V_r <2 cm

We have applied two event cuts on the event variables. First is the cut on the Z-position of the vertex (V_z). V_z is selected in the analysis presented here to be between ±30 cm from the interaction point to ensure uniform detector acceptance in the η range studied, because events with V_z far from the interaction point will cause a loss in the acceptance. Similarly V_r given by :

$$V_r = \sqrt{V_x^2 + V_y^2} \tag{4.6}$$

is selected to lie between ±2. The applied event cuts are tabulated in Table 4.1.

Table 4.2: Track cuts for K^{*0} invariant mass analysis

Cut name	Cut value
DCA range	-3 < DCA <3 cm
pT range	0.2 < p_T < 10 GeV/c
TPC Hit Points	NHitsFit>15
η range	-0.5 < η < 0.5

Track Cuts

Since the K^{*0} decays in such a short time that the daughter kaons and pions seem to originate from the interaction point. So, the tracks used in the presented analysis are called "primary tracks", meaning that the vertex is included as one of the tracks fit points. Track candidates whose distance of closest approach (DCA) to the primary interaction vertex was less than 3 cm were selected. To ensure optimal particle identification and momentum resolution it is necessary to apply quality cuts in the tracks measured in the TPC. Each track has a number of fit points associated with it by the STAR tracking software. In order to ensure accurate track momentum reconstruction and good dE/dx resolution, short tracks were eliminated from the analysis by requiring all tracks to have a minimum number of 15 fit points. For all the track candidates, the ratio between the number of TPC track fit points over the maximum possible points was required to be greater than 0.55 to avoid selecting split tracks. To maintain reasonable momentum resolution, only tracks with p_T larger than 0.2 GeV/c are selected. To avoid the acceptance drop in the high-η range all track candidates are required to have $|\eta| < 0.5$. The applied track cuts are tabulated in Table 4.2.

4.4 Particle Selection

The decay products of K^{*0}, pions and kaons, are identified using the combination of Time Projection Chamber (TPC) and Time of Flight (TOF) detectors. In addition to

momentum information, the TPC provides particle identification for charged particles by measuring their ionization energy loss (dE/dx). The plot of DE/dx is shown in Fig. Different bands seen represent Bethe-Bloch distributions folded with the experimental resolutions and correspond to different particle species. In the plot, the energy loss curves for charged pions and kaons can be easily identified. To quantitatively describe the particle identification, the variable N_σ for pions and kaons is defined as follows :

$$N_{\sigma\pi,\sigma K} = \frac{1}{R} log \frac{(dE/dx)_{measured}}{\langle dE/dx \rangle_{\pi,K}} \tag{4.7}$$

where, $(dE/dx)_{measured}$ is the measured energy loss for a track,$\langle dE/dx \rangle_{\pi,K}$ is the expected mean energy loss for charged pions or kaons tracks obtained by parameterizing modified Bethe Bloch function. 'R' is the dE/dx resolution, which varies between 6% and 10% from p+p to central Au+Au events and depends on the characteristics of each track, such as the number of dE/dx hits for a track measured in the TPC, the pseudorapidity of a track, etc. To select charged kaon and pion candidates $N_{\sigma pi} < 2$ and $N_{\sigma K} < 2$ cuts were used.

Even after using the N_{sigma} cuts there is still some ambiguity in the selected tracks. In the dE/dx plot (Fig. 4.11) we can see that as the momentum increases the bands of pion and kaon comes closer and closer and finally merges. So, for high momentum, its difficult to distinguish kaons from pions on the basis of N_{sigma} alone. To resolve this ambiguity we use the TOF detector. TOF detector measures the time of flight and pathlength from which we can get velocity and hence β given by :

$$\beta = \frac{v}{c} \tag{4.8}$$

Using β the mass of the particle is found :

$$M^2 = \frac{p^2}{\beta^2 - 1} \tag{4.9}$$

Table 4.3: PID cuts for K^{*0} invariant mass analysis

Cut name	Cut value
$N_{\sigma pi}$ range	-2 < $N_{\sigma pi}$ < 2
$N_{\sigma K}$ range	-2 < $N_{\sigma K}$ < 2
M_{π}^2 range	0.01038 < M_{π}^2 < 0.029 GeV^2/c^4
M_K^2 range	0.2104 < M_K^2 < 0.2796 GeV^2/c^4

From the data the distribution of the square of mass (M^2) of the charged particles ($c = \pm 1$) is obtained. In the plot three peaks corresponding to pion, kaon and proton are observed. The distribution is fitted with gaussian functions to obtain the mean (μ_{M^2}) and variance($\sigma^2_{M^2}$) for pion and kaon M^2. To identify kaons and pions along with the N_σ cut an additional M^2 cut is used. Tracks with M^2 lying between $\pm 2\sigma^{\pi}_{M^2}$ are selected as pion tracks and M^2 lying between $\pm 2\sigma^{K}_{M^2}$ are selected as kaon tracks. The fitting of the M^2 distribution for pions and kaons is shown in Fig. 4.13.

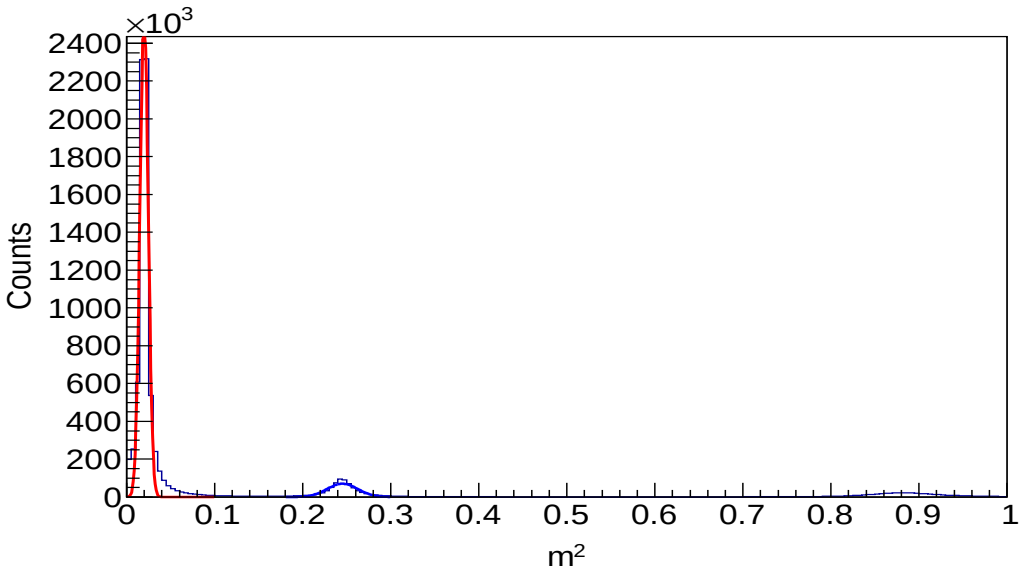

Figure 4.13: M^2 distribution

The particle selection cuts are tabulated in Table 4.3.

4.5 Invariant Mass Reconstruction

4.5.1 Unlike Sign Distribution

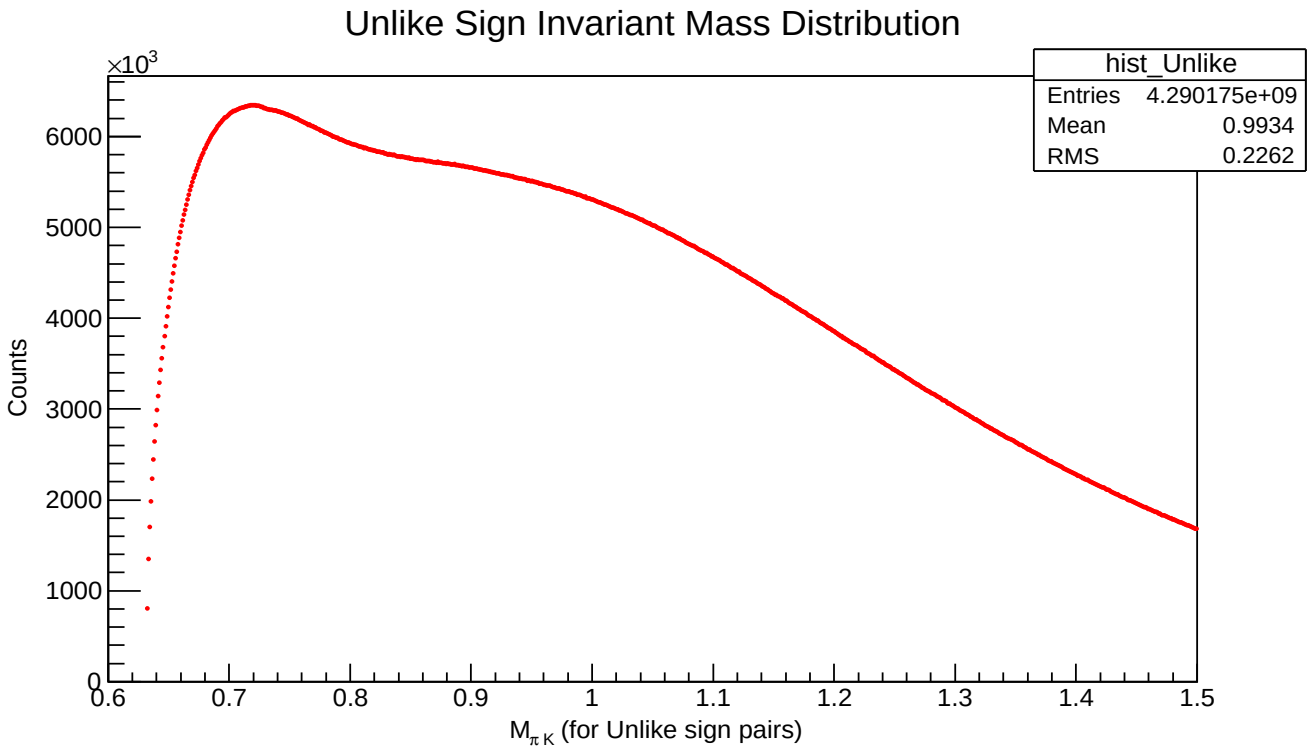

Figure 4.14: Unlike Sign distribution in UU data

In a relativistic heavy ion collision, it is difficult to build the K^{*0} signal by selecting accurately the corresponding pion and kaon daughters, as they are indistinguishable from the other primary tracks in an event. The two decay channels for K^{*0} resonance are as follows :-

$$K^{*0} \longrightarrow K^{+}\pi^{-} \ \ and \ \ \overline{K^{*0}} \longrightarrow K^{-}\pi^{+} \tag{4.10}$$

We reconstruct the invariant mass of K^{*0} by taking the opposite signed pion and

kaon pairs from the same event as defined by the equation below :

$$M_{K^{*0}} = \sqrt{[E_{K^+} + E_{\pi^-}]^2 - [\vec{p}_{K^+} + \vec{p}_{\pi^-}]^2} \tag{4.11}$$

$$M_{\overline{K^{*0}}} = \sqrt{[E_{K^-} + E_{\pi^+}]^2 - [\vec{p}_{K^-} + \vec{p}_{\pi^+}]^2} \tag{4.12}$$

where, $E_\pi = \sqrt{m_\pi^2 + p_\pi^2}$ and $E_\pi = \sqrt{m_K^2 + p_K^2}$. Histogram of invariant mass of K^{*0} obtained using these equations is called Unlike sign distribution as the sign of kaons and pions are opposite. The plot of the unlike sign invariant mass distribution is shown in Fig. 4.14.

However such a event wise construct also includes the background combinations of πK pairs not originating from K^{*0} as we are taking all possible combinations of available opposite sign pions and kaons. So, we dont get a signal peaking at K^{*0} mass in this distribution and expect that there is some background which we need to remove to get a clear signal.

4.5.2 Like Sign Distribution

Due to huge uncorrelated background the signal is not visible in the Unlike sign distribution. To observe the signal we need to subtract the combinatorial background from the same event πK pairs distribution. To find the combinatorial background we used a method called Like Sign method. In this technique, the combinatorial background is constructed through the invariant mass of pions and kaons of same charge from same event given by :

$$M = \sqrt{[E_{K^+} + E_{\pi^+}]^2 - [\vec{p}_{K^+} + \vec{p}_{\pi^+}]^2} \tag{4.13}$$

$$M = \sqrt{[E_{K^-} + E_{\pi^-}]^2 - [\vec{p}_{K^-} + \vec{p}_{\pi^-}]^2} \tag{4.14}$$

Since the number of positive and negative particles produced in relativistic heavy ion collisions are not same, the combinatorial background or the Like Sign distribution

is constructed by taking the geometric mean of number of like sign pairs as shown in equation below :

$$N_{Like} = 2 \times \sqrt{N_{\pi^+K^+} \times N_{\pi^-K^-}} \tag{4.15}$$

The plot of the Like sign distribution is shown in Fig. 4.15

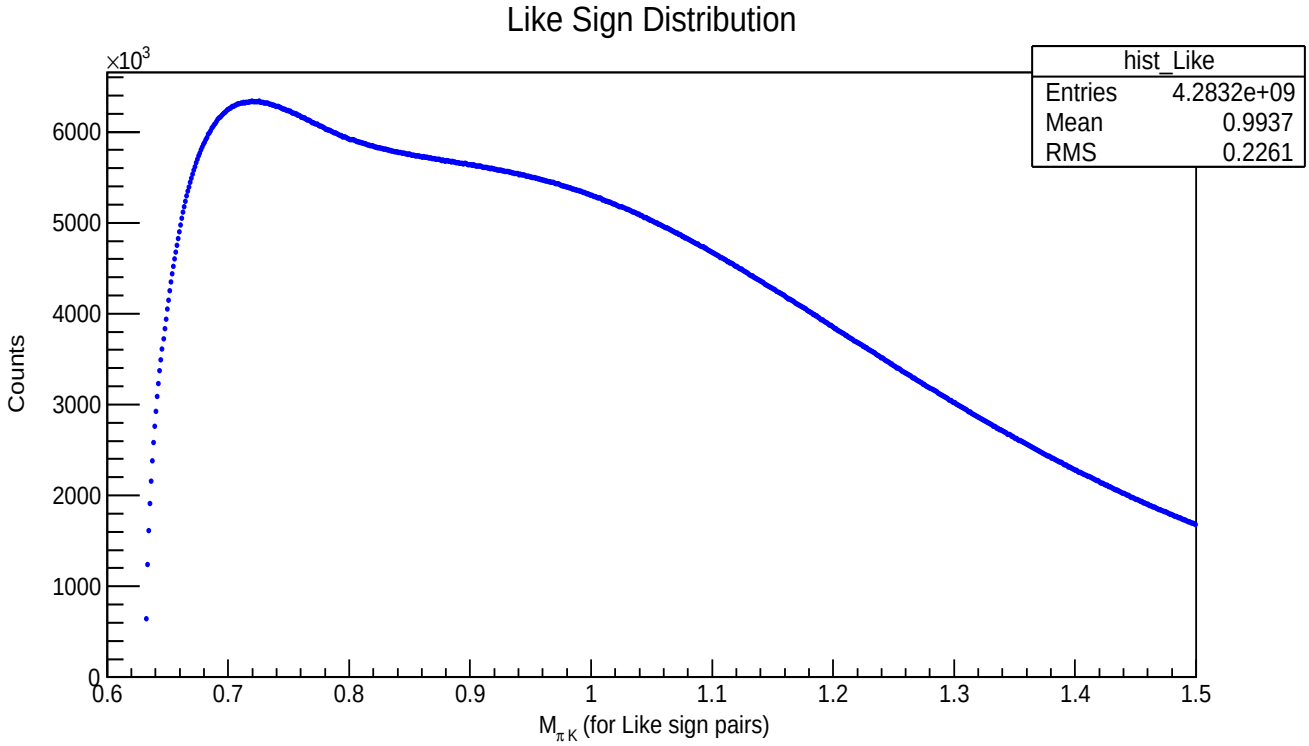

Figure 4.15: Like Sign distribution in UU data

4.5.3 K^{*0} Signal Distribution

The Like sign distribution is scaled with respect to the Unlike sign distribution by finding the scaling factor which is the ratio of the integral of the Unlike sign distribution to the Like sign distribution in the region of invariant mass away from the K^{*0} mass. This scaled Like sign distribution is then subtracted from the Unlike sign distribution to obtain the K^{*0} signal distribution as shown in Fig. 4.16. The signal is fitted with a combination of Breit-Wigner function and a second order polynomial

background to get the Mass, Width and Yield of K^{*0}. The fitted plot is shown in Fig. 4.16. The complete code is given in APPENDIX-III - Invariant Mass of K^{*0}.

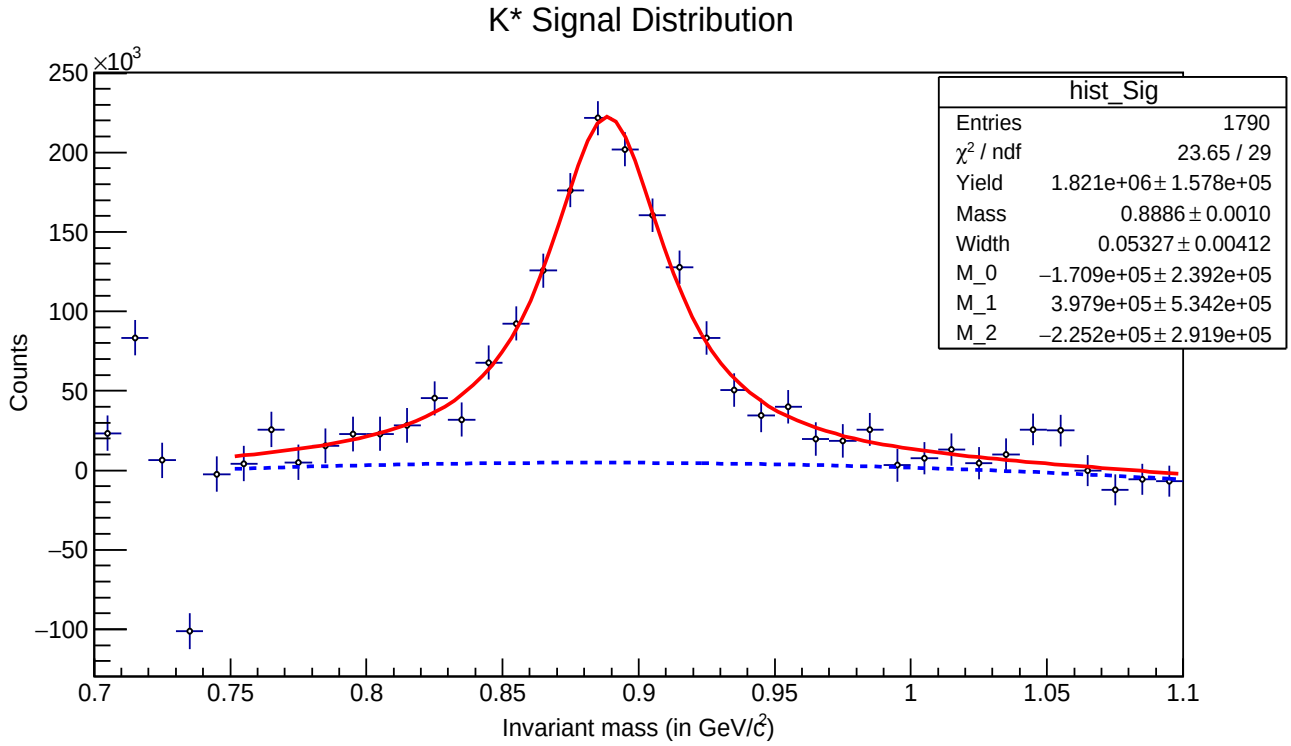

Figure 4.16: K^{*0} Signal distribution in UU data

The fit parameters are tabulated in Table 4.4.

Table 4.4: Fit parameters of K^{*0} invariant mass analysis

Parameter	Value
Yield	$1.82 \times 10^6 \pm 1.578 \times 10^5$
Mass	0.8886 ± 0.0010 GeV/c^2
Width	0.05327 ± 0.00412 GeV/c^2

Chapter 5

Elliptic Flow of K^{*0}

This chapter presents the study on elliptic flow of K^{*0} resonance. First we will try to understand the basics of anisotropic flow in heavy ion collisions and the origin of the flow harmonics. The second order flow harmonic v_2 is called elliptic flow coefficient. Next we will discuss about event planes and the derivation of v_2 from event planes. Finally we will discuss the analysis method to obtain event planes and calculate v_2 of K^{*0} in U-U data of STAR at $\sqrt{s_{NN}} = 193GeV$.

5.1 Introduction

In heavy ion collisions, "flow" means the collective motion of the particles produced in the process due to high pressure arising from compression and heating of nuclear matter. Collective motion of the particles can be divided into two groups - Longitudinal flow and Transverse flow. Logitudinal flow is the expansion of particles along the beam direction. Transverse flow is the collective expansion of the particles in the transverse plane. Transverse flow can further be divided into two groups - Radial flow and Anisotropic flow. Radial flow is the isotropic expansion driven by pressure gradients. Anisotropic flow as the name suggests is the anisotropic expansion arising due to anisotropy in pressure. We are interested in studying the anisotropic flow of K^{*0}.

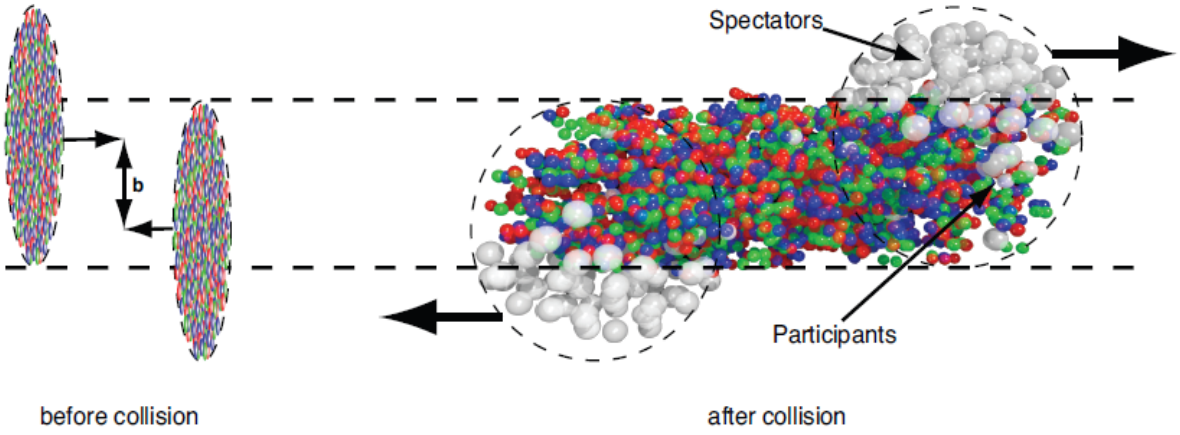

Figure 5.1: Collision of two heavy ions

Heavy-ions are extended objects and the system created in a head-on collision is different from that in a peripheral collision. To study the properties of the created system, collisions are therefore categorized by their centrality which is defined later in the report. Fig. 5.1 shows the colllision of two heavy ions and the production of particles in the overlapped region. The distance between the centres of the two ion is the impact parameter (b). In central collision the overlap region is larger and the particles production and interaction is also more than the peripheral collision. But in peripheral collsions the anisotropy of the overlapped region is more [9].

In non-central heavy ion collisions, the overlap area of two nuclei is not spatially isotropic as shown in Fig. 5.2. The initial spatial anisotropy with respect to the x-z plane (reaction plane) translates into a momentum anisotropy of the produced particles (anisotropic flow) because of the pressure gradient developed among the constituents. The experimental study of these anisotropies is done using flow harmonics v_n. We are interested in one of the flow harmonic called the elliptic flow coefficient (v_2), which gives the measure of the anisotropy in momentum space. The v_2 of charged particles and identified hadrons have been measured at RHIC and LHC.

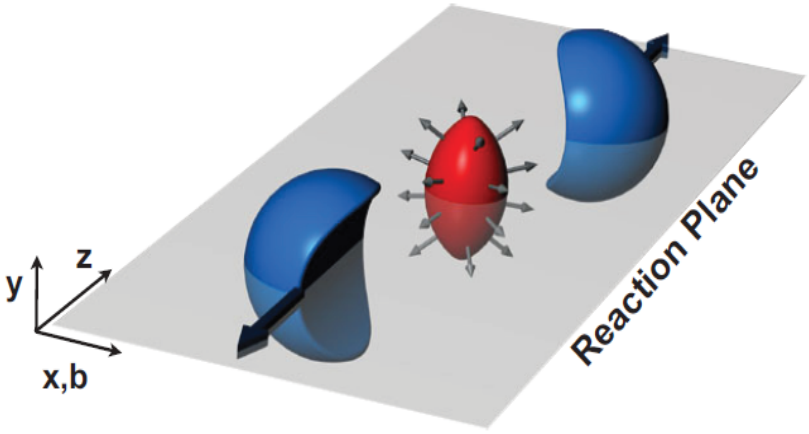

Figure 5.2: Spatial anisotropy in overlapped region of heavy ion collision

It is observed that the v_2 at low transverse momentum at RHIC and LHC is similar which is consistent with the predictions from hydrodynamic models. In particular, the measurement of v_2 of K^{*0} is important because of its small lifetime which is comparable to that of the medium formed in heavy ion collisions. Thus the decay products of K^{*0} can undergo in-medium effects like re-scattering and regeneration in the hadronic phase and the v_2 of K^{*0} may be modified.

5.2 Elliptic flow

In non-central heavy-ion collisions the initial volume of the interacting system is anisotropic in coordinate space. Due to multiple interactions this anisotropy is transferred to momentum space, and is then quantified via so-called flow harmonics v_n. In essence, anisotropic flow analysis is the measurement of flow harmonics v_n, which will be formally defined next.

The azimuthal distribution $r(\varphi)$ of the physical quantity of interest (for instance the azimuthal distribution of total transverse momentum of particles produced in a heavy-ion collision) is a periodic quantity and it is natural to expand it in a Fourier

series[10] :

$$r(\varphi) = \frac{x_0}{2\pi}\frac{1}{\pi}\sum_{n=1}^{\infty}[x_n cos(n\varphi) + y_n sin(n\varphi)] \tag{5.1}$$

where,

$$x_n = \int_0^{2\pi} r(\varphi) cos(n\varphi) d\varphi \tag{5.2}$$

$$y_n = \int_0^{2\pi} r(\varphi) sin(n\varphi) d\varphi \tag{5.3}$$

For each pair of Fourier coefficients, x_n and y_n, the corresponding flow harmonics v_n is defined in the following way

$$v_n \equiv \sqrt{x_n^2 + y_n^2} \tag{5.4}$$

Harmonics v_n can be related explicitly to the starting distribution $r(\varphi)$ in the following way:

$$\begin{aligned} \langle cos(n\varphi)\rangle &= \frac{\int_0^{2\pi} r(\varphi) cos(n\varphi) d\varphi}{\int_0^{2\pi} r(\varphi) d\varphi} \\ &= \frac{\frac{1}{\pi} v_n \int_0^{2\pi} cos^2(n\varphi) d\varphi}{v_0} \\ &= \frac{v_n}{v_0} \end{aligned} \tag{5.5}$$

From the first to second line in the equation above we have used the orthogonality relationship of the sine and cosine functions

$$\begin{aligned} \int_{-\pi}^{\pi} sin(mx) sin(nx) dx &= \pi\delta_{mn}, \\ \int_{-\pi}^{\pi} cos(mx) cos(nx) dx &= \pi\delta_{mn}, \\ \int_{-\pi}^{\pi} sin(mx) cos(nx) dx &= 0, \end{aligned} \tag{5.6}$$

By using a normalized distribution $r(\varphi)$, for which $v_0 = \int_0^{2\pi} r(\varphi) d\varphi = 1$, it follows immediately :

$$v_n = \langle cos(n\varphi)\rangle \tag{5.7}$$

The harmonic v_1 is called directed flow, the harmonic v_2 elliptic flow, the harmonic v_3 triangular flow, etc. When flow harmonics are considered as a function of transverse momentum and rapidity, $v_n(p_T, y)$, we refer to them as differential flow.

5.3 Analysis Method

The experimental measurement of anisotropies is done using the flow harmonics v_n. The invariant yield of particles produced in heavy ion collisions can be expanded in the form of Fourier series [11]:

$$E\frac{d^3N}{dp^3} = \frac{1}{2\pi}\frac{d^3N}{p_T dp_T dy}(1 + \sum_{n=1}^{\infty} v_n(p_T, y)cos[n(\varphi - \Psi_R)]) \tag{5.8}$$

where E is the energy of the particle, p the momentum, p_T the transverse momentum,φ the azimuthal angle, y the rapidity and Ψ_R is the reaction plane angle. The reaction plane is defined as the plane described by the vector between the centres of the colliding nuclei and the direction of the beam axis. The nth order Fourier coefficient v_n is given by [11]:

$$v_n = \langle cos[n(\varphi - \Psi_R)]\rangle \tag{5.9}$$

where the average is taken over all particles in all events.

Together, the first two Fourier coefficients, v_1 and v_2 are known as the anisotropic flow. Separately, the first Fourier cofficient v_1 is called directed flow and the second coefficient v_2 is called elliptic flow because in polar coordinates, for small values of v_2, the azimuthal distribution with non-zero second harmonic describes an ellipse.. This analysis focuses only on the measurement of v_2 given by :

$$v_2 = \langle cos[2(\varphi - \Psi_R)]\rangle \tag{5.10}$$

Elliptic flow is defined as a correlation between the azimuth of an outgoing particle, φ (in our analysis its K^{*0}) and the reaction plane, Ψ_R which is nothing but the azimuth

of impact parameter. For the estimation of v_2 of K^{*0} we have used the Event Plane method which is discussed below.

5.3.1 Event Plane Estimation

In real heavy ion experiment it is not possible measure the impact parameter between the two nuclei and hence the reaction plane is unknown. The reaction plane has to be estimated on an event-by-event basis by using the anisotropy flow itself. Such an estimated reaction plane is called an event plane. The event plane angle can be calculated in terms of the event flow vector Q_n [11]whose components are defined as :

$$X_n = Q_n cos(n\Psi_n) = \sum_{i=1}^{N} W_i cos(n\varphi_i) \tag{5.11}$$

$$Y_n = Q_n sin(n\Psi_n) = \sum_{i=1}^{N} W_i sin(n\varphi_i) \tag{5.12}$$

where W_i is the weight and N is the total number of produced particles in a given acceptance used for flow vector calculation in an event. For this analysis the weight factor are taken as the p_T of the particle. The nth harmonic event plane is given by :

$$\Psi_n = \frac{1}{n} tan^{-1}\left(\frac{\sum_{i=1}^{N} W_i sin(n\varphi_i)}{\sum_{i=1}^{N} W_i cos(n\varphi_i)}\right) = \frac{1}{n} tan^{-1}\left(\frac{Y_n}{X_n}\right) \tag{5.13}$$

The event plane angle Ψ_n determined from the nth harmonic is in the range $0 \leq \Psi_n < 2\pi/n$.

5.3.2 Event Plane Acceptance Correction

There are biases due to the finite acceptance of the detector which cause the particles to be azimuthally anisotropic in the laboratory system. Due to the non uniform acceptance of the detectors the distribution of the azimuthal angle is not flat. They can be removed by making the distribution of event planes isotropic in the laboratory.

There are several different methods to remove the effects of anisotropy. We are using two methods called "Recenter" and "Shift Correction" which are discussed below.

Recenter Correction

The simplest one is to recenter the distributions (X_n, Y_n) by subtracting the (Xn ,Yn) values with their averaged values over all events. The main disadvantage of this method is that it does not remove higher harmonics from the resulting distribution of Ψ_n.

Shift Correction

To remove the higher harmonics additional flattening of the event plane distribution is required. For this Shift Correction method is used. It fits the unweighted laboratory distribution of the event planes, summed over all events, to a Fourier expansion and devises an event-by-event shifting of the planes sneeded to make the final distribution isotropic.

5.3.3 Event Plane Resolution Correction

The finite number of particles in an event, which are available for calculating the event plane, leads to a limited resolution in the measured event plane angle. Therefore, to find the actual v_2 with respect to the real reaction plane, the measured v_2 is divided by a resolution correction factor (R)[11].

$$v_2 = \frac{v_2^{obs}}{R} = \frac{v_2^{obs}}{\langle cos2(\Psi_2 - \Psi_R)\rangle} \tag{5.14}$$

where Ψ_2 is the event plane angle and Ψ_R is the true reaction plane angle. The term $\langle cos2(\Psi_2 - \Psi_R)\rangle$ can be written as,

$$\langle cos2(\Psi_2 - \Psi_R)\rangle = \frac{\sqrt{\pi}}{2\sqrt{2}}\chi_2 exp(-\chi_2^2/4)[I_0(\chi_2^2/4) + I_1(\chi_2^2/4)] \tag{5.15}$$

where, I_x is the modified Bessel function of order x, and

$$\chi_2 = \frac{v_2}{\sigma} \quad and \quad \sigma^2 = \frac{1}{2N}\frac{< w^2 >}{< w >^2} \tag{5.16}$$

where, N is the number of particles and w are the weights used to calculate the flow vector Q_n.

5.3.4 v_2 vs Invariant Mass Method

For analyising v_2 of K^{*0} we have used v_2 vs Invariant Mass Method [11]. This is a standard method for calculating the v_2 of resonances or particles which are identified through their decay products. The v_2 calculted using the event planes is for "Signal + Background" distribution, our aim is to find v_2 for "Signal Only" distribution which is achieved using this method. The steps involved in this method are as follows :

1. First $v_2^{sig+bkg}$ is calculated as a function of πK invariant mass ($M_{\pi K}$), using

$$v_2^{sig+bkg} = \langle cos2(\varphi - \Psi_2)\rangle \tag{5.17}$$

2. It is then divided by resolution factor (R) to get the resolution corrected $v_2^{sig+bkg}$.

$$v_2^{sig+bkg} = \frac{v_2^{obs}}{R} = \frac{v_2^{obs}}{\langle cos2(\Psi_2 - \Psi_R)\rangle} \tag{5.18}$$

3. A 2-D histogram of $v_2^{sig+bkg}$ vs $M_{\pi K}$ distribution is obtained.

4. Then $v_2^{sig+bkg}$ is decomposed as :

$$v_2^{sig+bkg} = v_2^{sig} \times \left(\frac{N_{sig}}{N_{sig}+N_{bkg}}\right) + v_2^{bkg} \times \left(\frac{N_{bkg}}{N_{sig}+N_{bkg}}\right) \tag{5.19}$$

where, N_{sig} and N_{bkg} are the signal and background yields. v_2^{sig}, v_2^{bkg} and $v_2^{sig+bkg}$ are the elliptic flow coefficient for signal, background and total particles. Defining :

$$\alpha = \frac{N_{sig}}{N_{sig}+N_{bkg}} \quad and \quad (1-\alpha) = \frac{N_{bkg}}{N_{sig}+N_{bkg}} \tag{5.20}$$

which are the ratios of yields as a function of πK invariant mass. The term v_2^{bkg} is parameterized by a second order polynomial function of invariant mass.

5. Finally the $v_2^{sig+bkg}$ vs $M_{\pi K}$ distribution is fitted with Eq. (5.20) to get v_2^{sig} as one of the fit parameters.

6. For different p_T intervals $v_2^{sig}(p_T)$ is calculated and the plot of v_2^{sig} vs (p_T) is obtained.

Chapter 6

Elliptic Flow of K^{*0} Analysis and Results

In this chapter we present the complete analysis for the elliptic flow of K^{*0} in U+U data of STAR with $\sqrt{s_{NN}} = 193GeV$. The analysis is done for 9.98 million events. The analysis is done majorly in two parts. First is the Event Plane estimation and correction part and second is the elliptic flow calculation part. Both have different cuts. First we will analyse and discuss Event Planes with the results and plots. And then we will analyse and discuss Elliptic flow with the results and plots.

6.1 Event Planes

6.1.1 Event and Track cuts

Table 6.1: Event cuts for K^{*0} event planes analysis

Cut name	Cut value
V_Z range	-30 $< V_Z <$30 cm
V_r range	-2 $< V_r <$2 cm

We have applied two event cuts on the event variables. V_z is selected to be between $\pm$30 cm from the interaction point to ensure uniform detector acceptance in the η range studied, because events with V_z far from the interaction point will cause a loss

in the acceptance. Similarly V_r is selected to lie between ± 2. The applied event cuts are tabulated in Table 6.1.

Next are the track cuts which are a bit different from those used for the invariant mass analysis. Track candidates whose distance of closest approach (DCA) to the primary interaction vertex was less than 2 cm were selected. Tracks fit points was set to have a minimum number of 15 fit points. For all the track candidates, the ratio between the number of TPC track fit points over the maximum possible points was required to be greater than 0.55 to avoid selecting split tracks. A very strict cut was applied on p_T to reject the high momentum tracks. Only tracks with p_T larger than 0.2 GeV/c and below 2 GeV/c were selected. A slightly loose cut on η, $|\eta| < 1$ was used. The applied track cuts are tabulated in Table 6.2.

Table 6.2: Track cuts for K^{*0} event planes analysis

Cut name	Cut value
DCA range	-2 < DCA <2 cm
p_T range	0.2 < p_T < 2 GeV/c
TPC Hit Points	NHitsFit>15
η range	-1 < η < 1

6.1.2 Event Plane Estimation

The event and track cuts mentioned above are applied on the 9.98 million events and using Eq. (5.13), the event planes are estimated to obtain a distribution of Ψ_2. The nth harmonic event plane Ψ_n lies in the range $0 \leq \Psi_n < 2\pi/n$. So, Ψ_2 lies in the range $0 \leq \Psi_2 < \pi$. The plot of the Ψ_2 distribution is shown in Fig. 6.1.

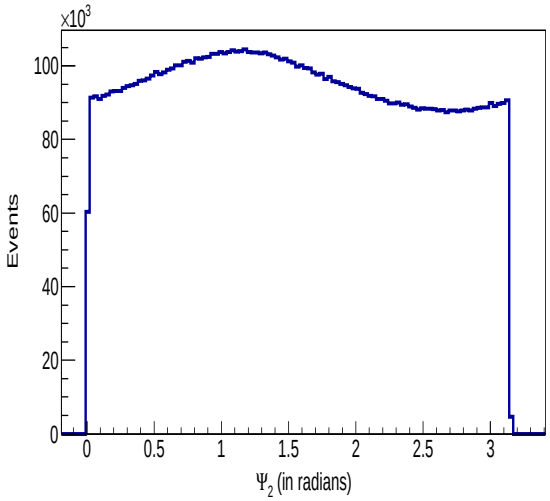

Figure 6.1: Event plane (Ψ_2) distribution

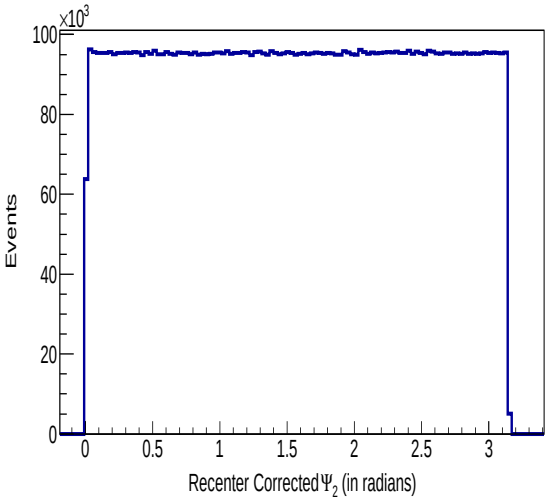

Figure 6.2: Recenter corrected Ψ_2 distribution

6.1.3 Event Plane Flattening

It is expected that the event plane distribution should be uniform between 0 to π. But the plot of the event plane shows that it is not flat or isotropic. For finding v_2, event planes are corrected to make the distribution flat. For this two methods namely - Recenter and Shift corrections as discussed in the Analysis Method part. Now the corrected distribution is fitted with the following funtion.

$$f = p_0[1 + 2p_1 cos(2\Psi_2) + 2p_2 sin(2\Psi_2)] \tag{6.1}$$

Very small values of the coefficients p_0, p_1 and p_2 would suggest that the distribution is flat. The fit parameters are tabulated in Table 6.3. The plot of the Recenter corrected and Shift corrected event plane distribution are shown in Fig. 6.2 and Fig. 6.3 respectively.

Table 6.3: Fit parameters for Recenter and Shift Corrected event plane fitting

Parameter	Value
p_0	$1.9 \times 10^5 \pm 60.57$
p_1	$3.72 \times 10^{-6} \pm 2.25 \times 10^{-4}$
p_2	$-6.59 \times 10^{-7} \pm 2.24 \times 10^{-4}$

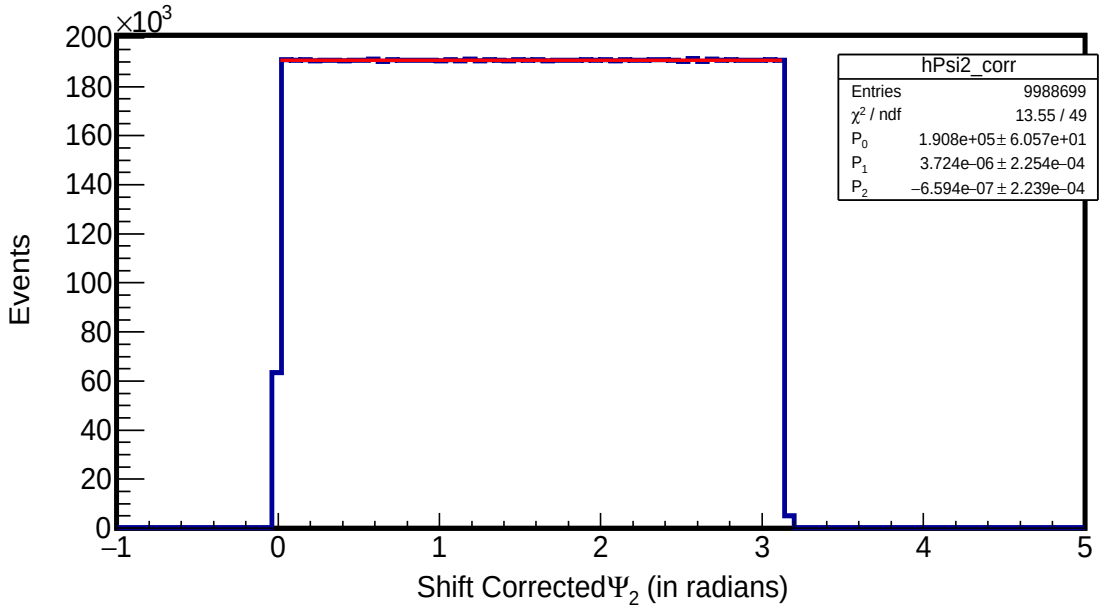

Figure 6.3: Recenter and Shift corrected Ψ_2 distribution

6.1.4 Event Plane Resolution Correction

Due to the finite number of particle produced in a heavy ion collision, there is a finite resolution of the estimated event plane angle. Thus the estimated v_2 has to be corrected for such event plane resolution (R). The plot of the resolution as a function of centrality is shown in Fig. 6.4.

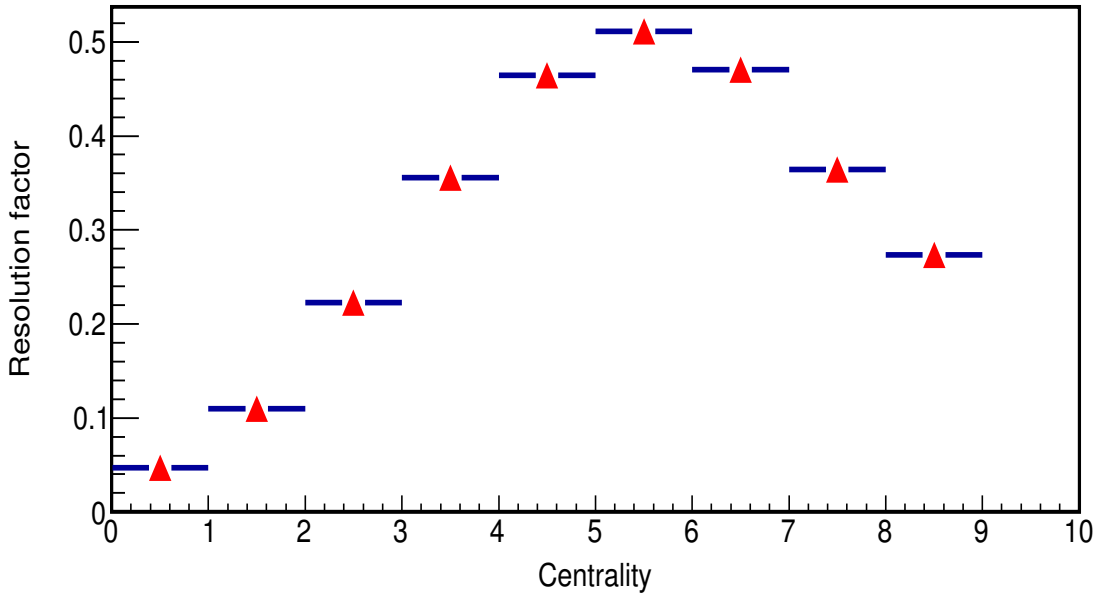

Figure 6.4: Resolution correction factor vs Centrality

6.1.5 Flow Vector

The event planes are obtained from the flow vector (Q) as given in Eq. (5.13):

$$\Psi_n = \frac{1}{n} tan^{-1}\left(\frac{\sum_{i=1}^{N} W_i sin(n\varphi_i)}{\sum_{i=1}^{N} W_i cos(n\varphi_i)}\right) = \frac{1}{n} tan^{-1}\left(\frac{Y_n}{X_n}\right)$$

where,

$$X_n = Q_n cos(n\Psi_n) = \sum_{i=1}^{N} W_i cos(n\varphi_i)$$

$$Y_n = Q_n sin(n\Psi_n) = \sum_{i=1}^{N} W_i sin(n\varphi_i)$$

We are interested in finding the elliptic flow v_2, so we do the analysis for second harmonic n=2. The plot of the components of the flow vector (Q) - (X_2, Y_2) or

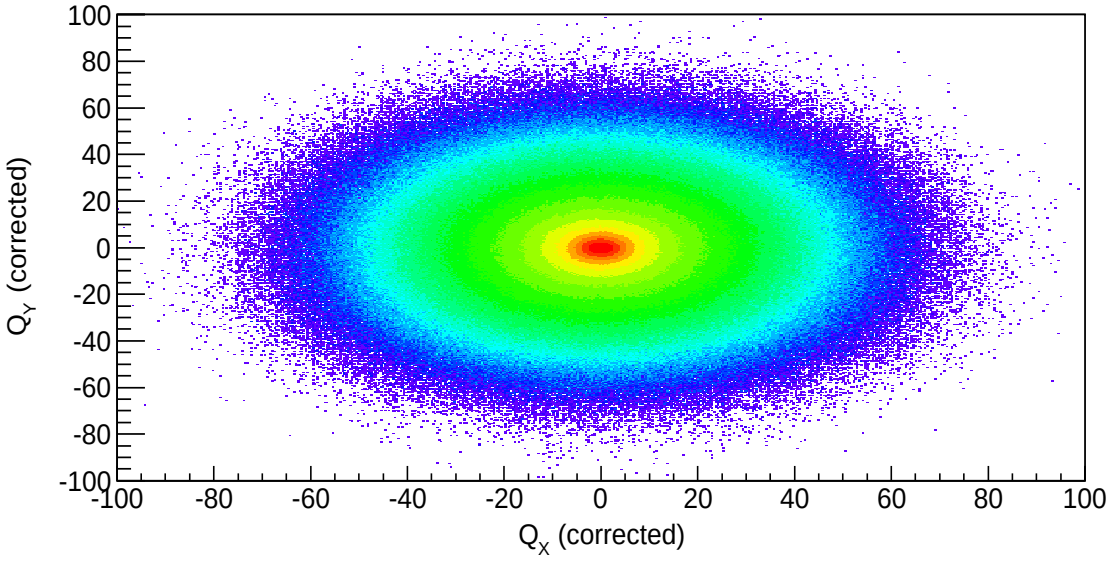

Figure 6.5: Q_X vs Q_Y distribution

(Q_X, Q_Y) is shown in Fig. 6.5. It is observed that the plot of Q_X vs Q_Y distributions are elliptic in nature with Q_X as the major axis and Q_Y as the minor axis. This justifies the name of the flow coefficient for the harmonic n=2 to be elliptic flow.

The complete code is given in APPENDIX-III - Event Planes.

6.2 Elliptic Flow

6.2.1 Event and Track cuts

Two event cuts on the event variables are applied. V_z is selected to be between $\pm$30 cm from the interaction point to ensure uniform detector acceptance in the η range studied. Similarly V_r is selected to lie between $\pm$2. The applied event cuts are tabulated in Table 6.4. Next are the track cuts which are a bit different from those

Table 6.4: Event cuts for K^{*0} elliptic flow analysis

Cut name	Cut value
V_Z range	-30 $< V_Z <$30 cm
V_r range	-2 $< V_r <$2 cm

used for the event planes. Track candidates whose distance of closest approach (DCA) to the primary interaction vertex was less than 3 cm were selected. Tracks fit points was set to have a minimum number of 15 fit points. For all the track candidates, the ratio between the number of TPC track fit points over the maximum possible points was required to be greater than 0.55 to avoid selecting split tracks. Tracks with p_T larger than 0.2 GeV/c and below 10 GeV/c were selected. A slightly loose cut on η, $|\eta| < 1$ was used. The applied track cuts are tabulated in Table 6.5. The drawback

Table 6.5: Track cuts for K^{*0} elliptic flow analysis

Cut name	Cut value
DCA range	-3 $< DCA <$3 cm
p_T range	0.2 $< p_T <$ 10 GeV/c
TPC Hit Points	NHitsFit>15
η range	-1 $< \eta <$ 1

of full event plane method is that the same set of tracks are used for the estimation of event plane and v_2. This may bias the measurement of v_2. To remove this auto-

correlation effect, the tracks used for the calculation of event planes are rejected for the calculation of $v_2^{sig+bkg}$ using the eventplanes.

6.2.2 Particle Selection

The decay products of K^{*0}, pions and kaons, are identified using the combination of Time Projection Chamber (TPC) and Time of Flight (TOF) detectors. In addition to momentum information, the TPC provides particle identification for charged particles by measuring their ionization energy loss (dE/dx). The plot of DE/dx is shown in Fig. Different bands seen represent Bethe-Bloch distributions folded with the experimental resolutions and correspond to different particle species. In the plot, the energy loss curves for charged pions and kaons can be easily identified. To quantitatively describe the particle identification, the variable N_σ for pions and kaons is defined as follows :

$$N_{\sigma\pi,\sigma K} = \frac{1}{R} log \frac{(dE/dx)_{measured}}{\langle dE/dx \rangle_{\pi,K}} \tag{6.2}$$

where, $(dE/dx)_{measured}$ is the measured energy loss for a track,$\langle dE/dx \rangle_{\pi,K}$ is the expected mean energy loss for charged pions or kaons tracks obtained by parameterizing modified Bethe Bloch function. 'R' is the dE/dx resolution, which varies between 6% and 10% from p+p to central Au+Au events and depends on the characteristics of each track, such as the number of dE/dx hits for a track measured in the TPC, the pseudorapidity of a track, etc. To select charged kaon and pion candidates $N_{\sigma pi} < 2$ and $N_{\sigma K} < 2$ cuts were used.

Even after using the N_{sigma} cuts there is still some ambiguity in the selected tracks. In the dE/dx plot (Fig. 4.11) we can see that as the momentum increases the bands of pion and kaon comes closer and closer and finally merges. So, for high momentum, its difficult to distinguish kaons from pions on the basis of N_{sigma} alone. To resolve this ambiguity we use the TOF detector. TOF detector measures the time of flight

and pathlength from which we can get velocity and hence β given by :

$$\beta = \frac{v}{c} \tag{6.3}$$

Using β the mass of the particle is found :

$$M^2 = \frac{p^2}{\beta^2 - 1} \tag{6.4}$$

From the data the distribution of the square of mass (M^2) of the charged particles ($c = \pm 1$) is obtained. In the plot three peaks corresponding to pion, kaon and proton are observed. The distribution is fitted with gaussian functions to obtain the mean (μ_{M^2}) and variance($\sigma^2_{M^2}$) for pion and kaon M^2. To identify kaons and pions along with the N_σ cut an additional M^2 cut is used. Tracks with M^2 lying between $\pm 2\sigma^{\pi}_{M^2}$ are selected as pion tracks and M^2 lying between $\pm 2\sigma^{K}_{M^2}$ are selected as kaon tracks. The fitting of the M^2 distribution for pions and kaons is shown in Fig. 4.13. The particle selection cuts are tabulated in Table 6.6.

Table 6.6: PID cuts for K^{*0} elliptic flow analysis

Cut name	Cut value
$N_{\sigma pi}$ range	-2 < $N_{\sigma pi}$ <2
$N_{\sigma K}$ range	-2 < $N_{\sigma K}$ <2
M^2_π range	$0.01038 < M^2_\pi < 0.029\ GeV^2/c^4$
M^2_K range	$0.2104 < M^2_K < 0.2796\ GeV^2/c^4$

6.2.3 v_2 vs Invariant mass

Using the corrected event planes, $v_2^{sig+bkg}$ is calculated as a function of πK invariant mass ($M_{\pi K}$), using

$$v_2^{sig+bkg} = \langle cos2(\varphi - \Psi_2)\rangle \tag{6.5}$$

To find the actual $v_2^{sig+bkg}$ with respect to the real reaction plane, it is divided by the event plane resolution correction factor (R) obtained above as shown in Fig. 6.4.

$$v_2^{sig+bkg} = \frac{v_2^{obs}}{R} = \frac{v_2^{obs}}{\langle cos2(\Psi_2 - \Psi_R)\rangle} \tag{6.6}$$

A 2-D histogram of $v_2^{sig+bkg}$ vs $M_{\pi K}$ distribution is obtained. This distribution is fitted with the following function :

$$f = v_2^{sig} \times \left(\frac{N_{sig}}{N_{sig}+N_{bkg}}\right) + v_2^{bkg} \times \left(\frac{N_{bkg}}{N_{sig}+N_{bkg}}\right) \tag{6.7}$$

to obtain the fit parameter v_2^{Sig} which is the desired elliptic flow coefficient of K^{*0}. The fitted plot of the $v_2^{sig+bkg}$ vs $M_{\pi K}$ distribution for p_T bins 0.8 to 1.4 GeV/c and 1.4 to 2.0 are shown in Fig. 6.6 and 6.7 below.

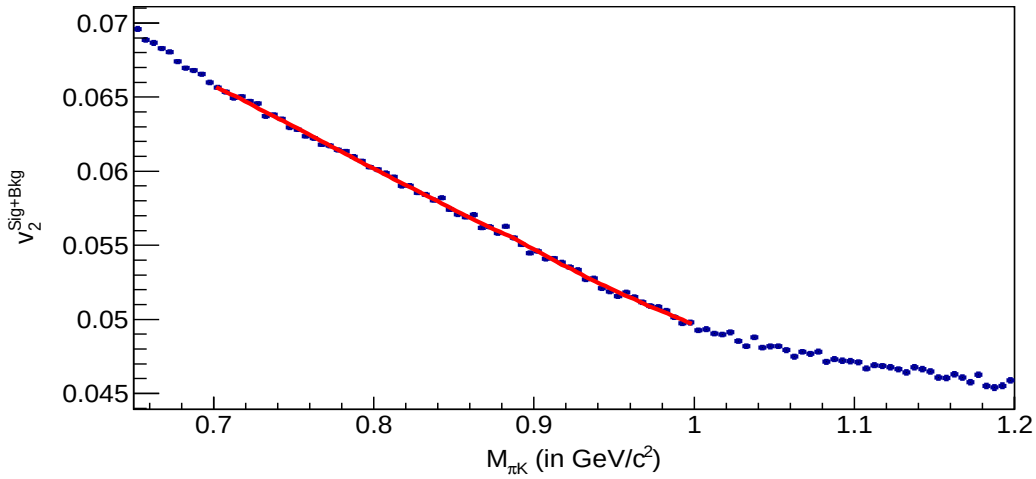

Figure 6.6: $v_2^{Sig+Bkg}$ vs $M_{\pi K}$ plot

The complete code is given in APPENDIX-III - Elliptic flow.

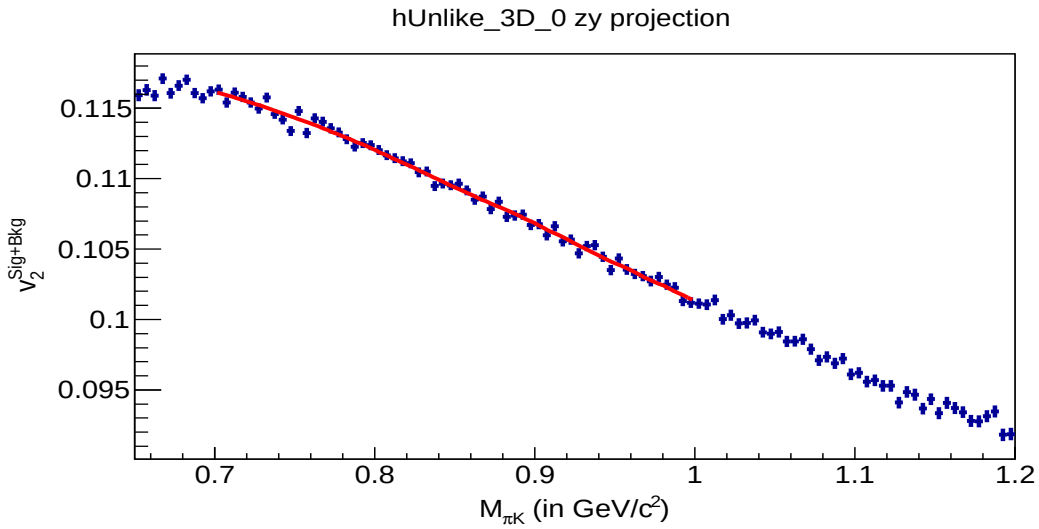

Figure 6.7: $v_2^{Sig+Bkg}$ vs $M_{\pi K}$ plot

6.2.4 v_2 vs p_T

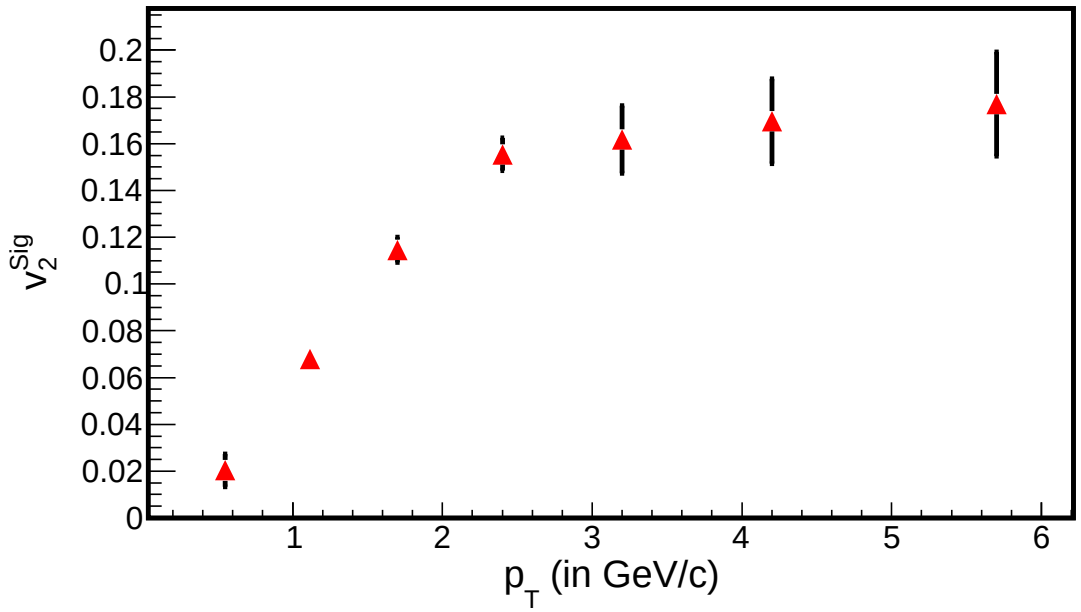

Figure 6.8: v_2^{Sig} vs p_T plot

Finally the elliptic flow coefficient of K^{*0} is plotted as a function of transverse

momentum. The plot of v_2^{Sig} vs p_T is shown in Fig. 6.8. The non-zero values of v_2^{Sig} confirms the presence of anisotropy in K^{*0} collective flow. v_2^{Sig} is p_T dependent. Its low for low p_T and increases with p_T and finally saturates for high p_T.

6.2.5 Comparision with Φ

Φ is meson resonance is quite similar with K^{*0} in terms of mass and decay daughters.

$$M_\Phi = 1.019 GeV/c^2 \qquad \Phi \longrightarrow K^+K^- \tag{6.8}$$

The v_2 of K^{*0} obtained is compared with the v_2 of Φ obtained from the analysis done by Mr. Vipul Bairathi. Fig. 6.9 shows the comparision of the two plots. Both these particles are mesons and their masses are close, so we expect similar v_2 distribution for both. The plot shows that the distribution for K^{*0} and Φ are very similar and this suggests that our analysis gives expected results.

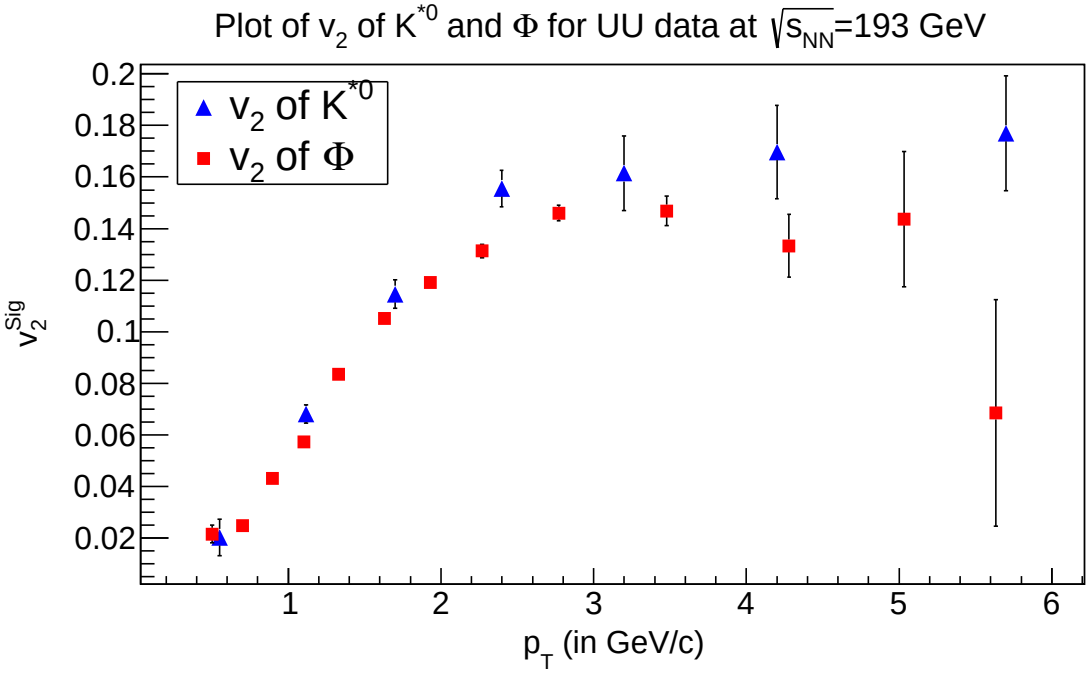

Figure 6.9: Plot for v_2^{Sig} of K^{*0} and Φ

6.2.6 Centrality dependence

v_2 vs p_T distribution for different centralities is obtained and the comparision plot is shown in Fig. 6.10. It shows the centrality dependence of v_2.

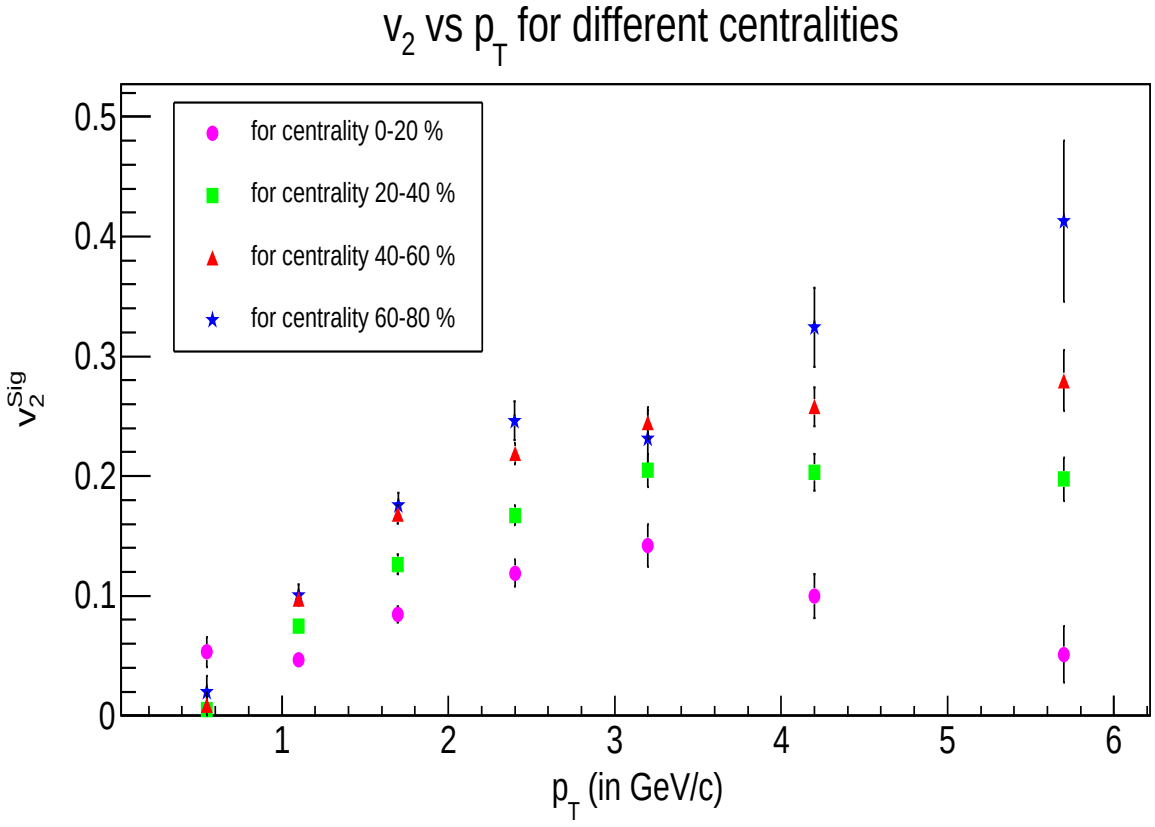

Figure 6.10: Plot for v_2^{Sig} vs p_T for different centralities

The plot shows that v_2^{Sig} increases with increase in centrality. The reason is - for central collisions (low centrality $0-20\%$) there is less spatial anisotropy, so the momentum anisotropy v_2 is also low and for peripheral collisions (high centrality $60-80\%$) there is high spatial anisotropy, so the momentum anisotropy v_2 is also high. Also it is observed that the difference in v_2 is more for central$(0-20\%)$ and mid-central$(20-40\%)$ collisions but for higher centralities the difference decreases. v_2 for mid-central$(40-60\%)$ and peripheral$(60-80\%)$ collisions are very close. The

reason is, for peripheral collisions, though the initial spatial anisotropy is very high, but the yield of K^{*0} is very small compared to the central collisions. So, the increase in anisotropy is compensated by decrease in yield.

Chapter 7

Summary and Conclusions

In this project we have investigated the resonance production and properties in heavy ion collision experiments using simulations and experimental data analysis . Specifically we have studied the production of K^{*0} resonance in these collisions. Resonances are particles with very short lifetime which decay into their daughter particles and cannot be detected directly in the detectors. Their detection is possible indirectly by the invariant mass reconstruction of their daughter particles. The short-lived resonances are a very useful tool in high energy collisions to study the dynamics and properties of the strongly interacting medium. We are interested in K^{*0} resonance, having short lifetime $\sim$ 4 fm/c which is comparable to the time interval between the two frezze-outs – also called the lifetime of the hadron gas or the hadronic cascade time ($\sim$ 10 fm/c). It means K^{*0} can decay within the hadron gas which is the phase between the chemical freeze-out (when inelastic collision ceases) and the kinetic freeze out (when elastic collision ceases). The daughter particles - K and π can interact with the medium particles which can affect the K^{*0} yield. These are called in-medium effects classified into two categories - Rescattering and Regeneration. Owing to short lifetime, the characteristic properties such as mass, width, yield and transverse momentum spectra of K^{*0} is very sensitive to the dynamics and in-medium effects. In rescattering the daughter particles can undergo elastic scattering with other hadrons

present in the medium and loose its energy or momentum. We cannot reconstruct back the resonance which leads to signal loss. On the other hand, in regeneration the pions and kaons in the medium can re-generate K^{*0} via pseudo-elastic interactions ($K\pi \rightarrow K^{*0} \rightarrow K\pi$). This regeneration process leads to gain in K^{*0} signal. We have tried to model these in-medium effects in a Mathematica code to obtain K^{*0} as a function of hadronic cascade time, τ_{HC}.

First we have studied the properties of some of the important resonances in heavy ion collision like their lifetimes, mass, decay channels and branching ratios. We derived the expression for invariant mass of the resonance for a two-body decay. Then we derived the cross-section expression of the resonance. We solved the Quantum Scattering Theory problem of the resonance scattering using Partial Wave Analysis method and Phase shift method to obtain the Breit-Wigner distribution for the resonance cross-section. The invariant mass distribution which corresponds to cross-section distribution will also follow the Breit-Wigner distribution.

To carry out the simulations of the heavy ion collision we studied and used A Multi-Phase Transport (AMPT) model. We carried out the study of K^{*0} production using AMPT model. The invariant mass distribution of the K^{*0} was obtained and fitted with a sum of Breit-Wigner and a linear background function to obtain the mass, width and yield of K^{*0}. In the AMPT model one can vary the termination time of hadronic cascade using a paramater - "NTMAX". Larger the hadronic cascade time, more is the re-scattering among the decay daughters (πK) of the K^{*0} meson. We observed that reconstructed K^{*0} signal is lost with the increase in hadronic cascade time. The reason for this decrease in signal can be found be studying the systematics of the in-medium effects.

We derived the kinetics and rate equations for Rescattering and Regeneration for

the general resonance decay $R \rightarrow A + B$. The solution of the rate equations are calculated. We derived the time dependence of the reaction rates. The rate equation solutions and the time dependent reaction rates were then used to develop a model using Mathematica coding to demonstrate the effects of rescattering and regeneration on the time evolution of the K^{*0} yield. The plot obtained from the code shows the decrease in K^{*0} ratio with τ (time elapsed after Chemical Freeze-out). This model explains our observation of the decrease in reconstructed K^{*0} signal as a function of τ_{HC} in the AMPT analysis and confirms the effect of rescattering and regeneration on the resonance yield. This model is very significant because using this model we can calculate the hadron gas lifetime in experiments. From the experimental data K^{*0} number ratios at different collision energies can be obtained and compared with our model plot to get the corresponding τ_{HC}.

After studying K^{*0} resonance using simulations, next we want to analyse K^{*0} resonance production and properties is experimental data using the model and algorithm used in simulation. So, the second part involves experimental data analysis of U-U collision data of STAR at center of mass energy of 193 GeV ($\sqrt{s_{NN}} = 193GeV$). The analysis was done to get the invariant mass distribution of K^{*0} from experiment and fitted with Breit-Wigner function to obtain different parameters like the mass, yield and width of the resonance to verify the results obtained from simulation. The results obtained were close to the PDG values, so the analysis is quite efficient. Finally we studied "Elliptic flow" (v_2)of K^{*0}, which is a very important property of the resonance. Elliptic flow gives the experimental measurement of the momentum anisotropies produced in the collective flow of the particle. The plot of v_2 as a function of p_T for minimum-bias was obtained. v_2 low for low p_T and increases with p_T and finally saturates for high p_T. The non-zero value of v_2 confirms the presence of

anisotropy in collective flow of K^{*0}. The elliptic flow of K^{*0} was then compared with that of Φ which is also a meson resonance like K^{*0} with mass close to K^{*0}. The plot shows that v_2 of both the particles are quite similar as expected which confirms the efficiency of the analysis. To show the centrality dependance, v_2 vs p_T is plotted for different centralities. Comparision of the plots shows that v_2 increases with increase in centrality. This confirms the presence of different degree of anisotropies for different centralities. For central collisions, initial spatial anisotropy is less, so the momentum anisotropy v_2 is less. And for peripheral collisions, initial spatial anisotropy is more, so the momentum anisotropy v_2 is more. It was also observed that the difference in v_2 is more for central and mid-central collisions but v_2 for mid-central and peripheral collisions is very close. The reason is, for peripheral collisions, though the initial spatial anisotropy is very high, but the yield of K^{*0} is very small. So, the increase in anisotropy is compensated by decrease in yield.

References

[1] *Multiphase transport model for relativistic heavy ion collisions*, Zi-Wei Lin, Che Ming Ko, Bao-An Li, Subrata Pal and Bin Zhang , **Physical Review C 72, 064901 (2005)**

[2] *Review of Particle Physics*, K. Nakamura et al. (Particle Data Group), **J. Phys. G 37, 075021 (2010)**

[3] *Strange Hadron Resonances: Freeze-Out Probes in Heavy-Ion Collisions*, C. Markert, G. Torrieri and J. Rafelski, **arXiv:hep-ph/0206260**

[4] *Strange hadron resonances as a signature of freeze-out dynamics*, Giorgio Torrieri, Johann Rafelski, **Physics Letters B 509 (2001) 239245**

[5] Introduction to Quantum Mechanics, David J. Griffiths

[6] Introduction to High Energy Physics, Donald H. Perkins

[7] Introductory Nuclear Physics, Kenneth S. Krane

[8] Introduction to Elementary Particles, David J. Griffiths

[9] *Elliptic Flow: A Brief Review*, Raimond Snellings, **New J.Phys.13:055008,2011**

[10] *Flow study in relativistic Nuclear Collisions by Fourier Expansion of azimuthal particle distributions*, S. Voloshin, Y. Zhang, **Z.Phys.C70:665-672,1996**

[11] *Methods for analyzing anisotropic flow in relativistic nuclear collisions* , S. Voloshin, A. M. Poskanzer, **Phys.Rev.C58:1671-1678,1998**

Appendix A

A.1 APPENDIX - I

A.1.1 Macro - 1

Shell script for AMPT event generation in batch mode for 50000 events for Au-Au collision at $\sqrt{s_{NN}} = 200$ GeV. The output of this code are text files containing the following information of all the produced particles - particle identification, mass, position coordinates, momentum coordinates.

```
#!/bin/bash
random=2000
while [ "$random" -lt 2005 ];  do
    cd /home/arabinda/AMPT/run
    cp -r ../test $random
    cd $random
    make
    sh exec &
    sleep 5
     echo $random' complete!'
#     random = $random + 1
```

```
    let "random+=1"
done
```

A.1.2 Macro - 2

Root Macro for analyising the produced particle information from the text files and generating a root file having all the information organised in a tree structure.

```
#include<TTree.h>
#include<TRandom.h>
#include<iostream.h>
#include<math.h>
#include<stdio.h>
#include<fstream.h>
#include<map.h>
#include"TH2F.h"
#include"TCanvas.h"
#include"TLorentzVector.h"
#include"TROOT.h"

int makeAmptroot_50000()
{
//input file variables:
  Int_t    evn,tn,nov,npp,npt,nesp,ninesp,nest,ninest,pid,counter=0;
   // for ampt.dat
Float_t    px,py,pz,mass,X,Y,Z,t,b;
//for ampt.dat
```

```
  Int_t    evn2, it, na, nb, nab, sr, stat, tpp;
  Float_t    nx, ny, zz, theta_p , phi_p , theta_t , phi_t , imp, psi;
//Booked variables in tree:
const Int_t  mul  = 90000;
const Int_t  nucl = 600;
      Int_t  Event=0, Na, Nb ,Nab, Mult, Npartp, Npartt, Nesp,
      Ninesp, Nest, Ninest;
      Float_t  Imp, Theta_p , Phi_p , Theta_t , Phi_t, Psi;
      // event variables
      Int_t  Stat[nucl], PID[mul];        //particle variables
    Float_t  Nx[nucl], Ny[nucl], Nz[nucl], Px[mul], Py[mul], Pz[mul];
    Float_t   XX[mul],  YY[mul],  ZZ[mul], TT[mul], Mass[mul];
    //particle variables
    Char_t outfile[100];
//define a root file:---------------------------------------------
  //sprintf(outfile,"AuAu_200_SM_10mb_folder%d.root",folder);
    TFile *fampt = new TFile("AuAu_200_SM_for_50000_events.root",
    "recreate");
    TTree *tr    = new TTree("tr","Reconst ntuple");
//Define event branches:------------------------------------------
 tr->Branch("Event", &Event,"Event/I");
 tr->Branch("Mult",  &Mult, "Mult/I");
 // multiplicity = tracks
 tr->Branch("Npartp",&Npartp,"Npartp/I");
 tr->Branch("Npartt",&Npartt,"Npartt/I");
 tr->Branch("Nesp",  &Nesp, "Nesp/I");
```

```
 tr->Branch("Ninesp",&Ninesp, "Ninesp/I");
 tr->Branch("Nest",  &Nest, "Nest/I");
 tr->Branch("Ninest",&Ninest, "Ninest/I");
 tr->Branch("Imp",&Imp, "Imp/F");
 tr->Branch("Na", &Na, "Na/I");       //Na = Projectile mass no.
 tr->Branch("Nb", &Nb, "Nb/I");       //Nb = Target mass no.
 tr->Branch("Nab",&Nab, "Nab/I");
 tr->Branch("Psi",&Psi, "Psi/F");
//particle branches:
  tr->Branch("Nx",  &Nx,  "Nx[Nab]/F");   // Nab = na+nb;
  tr->Branch("Ny",  &Ny,  "Ny[Nab]/F");
  tr->Branch("Nz",  &Nz,  "Nz[Nab]/F");
  tr->Branch("Stat",&Stat,"Stat[Nab]/I");
  tr->Branch("PID", &PID, "PID[Mult]/I");
  tr->Branch("Px",  &Px,  "Px[Mult]/F");
  tr->Branch("Py",  &Py,  "Py[Mult]/F");
  tr->Branch("Pz",  &Pz,  "Pz[Mult]/F");
  tr->Branch("Mass",&Mass,"Mass[Mult]/F");
  tr->Branch("XX",  &XX,  "XX[Mult]/F");
  tr->Branch("YY",  &YY,  "YY[Mult]/F");
  tr->Branch("ZZ",  &ZZ,  "ZZ[Mult]/F");
  tr->Branch("TT",  &TT,  "TT[Mult]/F");
//*********************************************************************************
cout<<"making .root file from .dat file..."<<endl;
ifstream infile[3000];
ifstream infile2[3000];
```

```
// infile2 = npart-xy.dat


char *fname   = new char[100];
char *fname2 = new char[100];
 for(int fid=2000; fid<2015; fid++) //Loop on folder;
 usually one root file for each folder
{//1 folder loop
  sprintf(fname,"%d/ana/ampt.dat",fid);
       sprintf(fname2,"%d/ana/npart-xy.dat",fid);
       infile[fid].open(fname);
       infile2[fid].open(fname2);
while(infile2[fid])
    {//2 event loop
      //infile[fid]>>evn>>tn>>nov>>b>>npp>>npt>>nesp>>
      ninesp>>nest>>ninest>>psi;
    infile[fid]>>evn>>tn>>nov>>b>>npp>>npt>>nesp>>ninesp
    >>nest>>ninest;       // for AMPT without psi
     infile2[fid]>>evn2>>it>>na>>nb>>imp;
    //infile2[fid]>>evn2>>it>>na>>theta_p>>phi_p>>nb>>theta_t
    >>phi_t>>imp;
 if(infile2[fid].eof()) break;
 if(infile[fid].eof())  break;
 //cout<<"No of particles in "<<evn<<" event = "<<nov<<endl;
       Event   = Event+1;
       Mult    = nov;
       Imp     = imp;
```

```
        Npartp  = npp;
        Npartt  = npt;          tpp = npp+npt;
        Nesp    = nesp;
        Ninesp  = ninesp;
        Nest    = nest;
        Ninest  = ninest ;
        Na      = na;
        Nb      = nb;           nab = na+nb;
        Nab     = nab;
Psi     = psi;
//for U+U only:
        Theta_p = theta_p;
        Phi_p   = phi_p;
        Theta_t = theta_t;
        Phi_t   = phi_t;


 //*****************************ampt.dat particle loop**************
 for(int j=0;j<nov;j++)                              //particle loop
    {
            infile[fid]>>pid>>px>>py>>pz>>mass>>X>>Y>>Z>>t;
            Px[j] = px;
            Py[j] = py;
            Pz[j] = pz;
          Mass[j] = mass;
           PID[j] = pid;
      XX[j] = X;
```

```
    YY[j] = Y;
    ZZ[j] = Z;
    TT[j] = t;
  }
 //***************************ampt.dat particle loop ends**************
//~~~~~~~~~~~ npart-xy.dat loop ~~~~~~~~~~~~~~~~~~
   for(int k=0;k<nab;k++)
{
          infile2[fid]>>nx>>ny>>sr>>stat>>zz>>theta_p>>phi_p;
           Nx[k] = nx;
           Ny[k] = ny;
   Nz[k] = zz;
         Stat[k] = stat;
}
//~~~~~~~~~~~npart-xy.dat loop ends~~~~~~~~~~~~~~~~~~~~

   if(evn!=evn2) {
     cout<<"..Error.Event mismatch... evn= "<<evn<<"\txy-ev= "
     <<evn2<<"  f_id= "<<fid<<endl;
       break;
      }
tr->Fill();

if(Event%100==0)
{
cout<<"am= "<<evn<<"  xy-ev= "<<evn2<<"  f_id= "<<fid<<"  Mul= "
```

```
<<Mult<<"\tnpart="<<tpp<<endl;
}
}//2 event loop end

}//1 folder loop end
fampt->cd();
tr->Write();
fampt->Close();
cout<<" Tree written succesfully "<<endl;
return 0;
}//main end
```

A.1.3 Macro - 3

Root Macro for calculating the invariant mass and generating for unlike sign and like sign distributions.

```
#define my_analysis_50000_cxx
#include "my_analysis_50000.h"
#include <TH1.h>
#include <TH2.h>
#include <TH3.h>
#include <TStyle.h>
#include <TCanvas.h>
void my_analysis_50000::Loop()
```

```
{
   if (fChain == 0) return;
   Long64_t nentries = fChain->GetEntriesFast();
   Long64_t nevn = fChain->GetEntries();
   Long64_t nbytes = 0, nb = 0;
   TVector3 aTrack2;
   TVector3 aTrack1;
   TVector3 aTrack;
   TVector3 aTrack21;
   TVector3 aTrack22;
   TVector3 aTrack23;
   TVector3 aTrack24;
   Float_t pi=3.14159;
   TH1F *hKStarMInv_pp = new TH1F("hKStarMInv_pp","Inv mass
    dist K+ Pi+",900,0.6,1.5);
 TH1F *hKStar_pt_pp = new TH1F("hKStar_pt_pp","Transverse
momentum dist K+ Pi+",100,0,10);
 TH1F *hKStar_phi_pp = new TH1F("hKStar_phi_pp","Phi dist K+
Pi+",800,-7,7);
   TH1F *hKStarMInv_mm = new TH1F("hKStarMInv_mm","Inv mass
   dist K- Pi-",900,0.6,1.5);
 TH1F *hKStar_pt_mm = new TH1F("hKStar_pt_mm","Transverse
momentum dist K- Pi-",100,0,10);
 TH1F *hKStar_phi_mm = new TH1F("hKStar_phi_mm","Phi dist
dist K- Pi-",800,-7,7);
   TH1F *hKStarMInv_pm1 = new TH1F("hKStarMInv_pm1","Inv mass
```

```
  dist K+ Pi-",900,0.6,1.5);
 TH1F *hKStar_pt_pm1 = new TH1F("hKStar_pt_pp","Transverse
 momentum dist K+ Pi-",100,0,10);
 TH1F *hKStar_phi_pm1 = new TH1F("hKStar_phi_pp","Phi dist
 dist K+ Pi-",800,-7,7);
   TH1F *hKStarMInv_pm2 = new TH1F("hKStarMInv_pm2","Inv mass
   dist K- Pi+",900,0.6,1.5);
 TH1F *hKStar_pt_pm2 = new TH1F("hKStar_pt_pm2","Transverse
 momentum dist K- Pi+",100,0,10);
 TH1F *hKStar_phi_pm2 = new TH1F("hKStar_phi_pm2","Phi dist
 dist K- Pi+",800,-7,7);
   TH1F *himp = new TH1F("himp", "imp param",2000, 0., 20.);
   // These variables are re-defined
   Float_t b =0.0;
   Int_t mul=0;
   Int_t evn=0; //rihan
    Int_t countevent = 0;
    Int_t a=0;
    Int_t countentry = 0;
Int_t count1=0,count2=0,count3=0;
    //   float v2sine = 0;
 cout<<" Ampt_phiv2 program Starts for "<<nevn<<" Events.."<<endl;
// Event Loop starts-------------------------------------
 for (Long64_t jentry=0; jentry<nentries;jentry++)
 // for (Long64_t jentry=0; jentry<10000 ;jentry++)
   {//event loop
```

```
Long64_t ientry = LoadTree(jentry);
if (ientry < 0) break;
nb = fChain->GetEntry(jentry);   nbytes += nb;
        evn++;   //rihan
 // if(evn==1001) continue;
b = Imp;              //Impact;
//Int_t npart = Event_NpartP + Event_NpartT;
Int_t npart = Npartp+ Npartt;  // Npart;
if(evn%1000==0)
  cout << "---- Processing event # " <<evn<<"\tmult = "<<mul
  <<"\tnpart = "<<npart<<endl;
//select centrality
mul = Mult;                 //refmult;
//TVector3 aTrack;
Double_t EK_pipl[10000]={0.},EK_kmn[10000]={0.};
// declearation for K* mass
Double_t EK_pimn[10000]={0.},EK_kpl[10000]={0.};
// declearation for K* mass
Int_t g=0,h=0;
Int_t i=0,j=0;
Double_t   px_pipl[10000],py_pipl[10000],pz_pipl[10000];
Double_t   px_pimn[10000],py_pimn[10000],pz_pimn[10000];
Double_t   px_kpl[10000],py_kpl[10000],pz_kpl[10000];
Double_t   px_kmn[10000],py_kmn[10000],pz_kmn[10000];
himp->Fill(b);
//TVector3 aTrack1;
```

```
//2nd particle loop starts
for(Int_t tr=0; tr<mul; tr++)
  {
    aTrack1.SetXYZ(Px[tr],Py[tr],Pz[tr]);
     count1++;
   Float_t pt = aTrack1.Perp();
   if(pt==0.0) continue;
count2++;
    Float_t eta2 = aTrack1.Eta();
    if(TMath::Abs(eta2) > 1.0) continue;
count3++;
    if(PID[tr] == 211)
      {
EK_pipl[g] = pow((Px[tr]*Px[tr]+Py[tr]*Py[tr]+Pz[tr]*Pz[tr]
 + Mass[tr]*Mass[tr]),0.5);
px_pipl[g]=Px[tr];
py_pipl[g]=Py[tr];
pz_pipl[g]=Pz[tr];
g++;
      }//pion plus selection ends
    //pion minus selection
    //TVector3 aTrackpimn;
    if(PID[tr] == -211)
      {
EK_pimn[h] = pow((Px[tr]*Px[tr]+Py[tr]*Py[tr]+Pz[tr]*Pz[tr]
 + Mass[tr]*Mass[tr]),0.5);
```

```
px_pimn[h]=Px[tr];
py_pimn[h]=Py[tr];
pz_pimn[h]=Pz[tr];
h++;
//cout <<"pi- selctd" <<  Px[tr] << " "<<Py[tr]<< " "<<Pz[tr]
<<" " <<Mass[tr]<<endl;
      }//pion minus selection ends
    //kaon plus selection
    //TVector3 aTrackkpl;
    if(PID[tr] == 321)
      {
EK_kpl[i] = pow((Px[tr]*Px[tr]+Py[tr]*Py[tr]+Pz[tr]*Pz[tr]
+ Mass[tr]*Mass[tr]),0.5);
       px_kpl[i]=Px[tr];
py_kpl[i]=Py[tr];
pz_kpl[i]=Pz[tr];
i++;
      }//kaon plus selection ends
    //kaon minus selection
    //TVector3 aTrackpimn;
    if(PID[tr] == -321)
      {
//aTrackkmn.SetXYZ(Px[tr],Py[tr],Pz[tr]);
//Float_t ptkmn = aTrackkmn.Pt();
//Float_t phikmn = aTrackkmn.Phi();
//Float_t etakmn = aTrackkmn.Eta();
```

```
EK_kmn[j] = pow((Px[tr]*Px[tr]+Py[tr]*Py[tr]+Pz[tr]*Pz[tr]
+ Mass[tr]*Mass[tr]),0.5);
px_kmn[j]=Px[tr];
py_kmn[j]=Py[tr];
pz_kmn[j]=Pz[tr];
j++;
//cout <<"K- selctd " <<  Px[tr] << " "<<Py[tr]<< " "
<<Pz[tr]<<" " <<Mass[tr]<<endl;
 }//kaon minus selection ends
       }//2nd particle loop ends
//TVector3 aTrack2;
//same event calculation K+ Pi-
for(Int_t m=0; m<h; m++)
  {
    for(Int_t l=0; l<i; l++)
      {
aTrack21.SetXYZ((px_pimn[m]+px_kpl[l]),(py_pimn[m]+
py_kpl[l]),(pz_pimn[m]+pz_kpl[l]));
Double_t phi_pm1 = aTrack21.Phi();
Double_t pt_pm1 = aTrack21.Pt();
Double_t eta_pm1 =aTrack21.Eta();
Double_t  KStarInvM_pm1 =    pow((pow((EK_pimn[m]+
EK_kpl[l]),2) - (pow((px_pimn[m]+px_kpl[l]),2)+pow
((py_pimn[m]+py_kpl[l]),2) +pow((pz_pimn[m]+pz_kpl[l]),
2))),0.5);
//hKStarMInv_pm1->Fill(KStarInvM_pm1,pt_pm1,phi_pm1);
```

```
hKStarMInv_pm1->Fill(KStarInvM_pm1);
hKStar_pt_pm1->Fill(pt_pm1);

hKStar_phi_pm1->Fill(phi_pm1);
//cout <<"K+ Pi- selctd" <<endl;
      }//l loop
  }//m loop same event loop ends K+ Pi-
//same event calculation K- Pi+
for(Int_t m=0; m<g; m++)
  {
    for(Int_t l=0; l<j; l++)
      {
aTrack22.SetXYZ((px_pipl[m]+px_kmn[l]),(py_pipl[m]
+py_kmn[l]),(pz_pipl[m]+pz_kmn[l]));
Double_t phi_pm2 = aTrack22.Phi();
Double_t pt_pm2 = aTrack22.Pt();
Double_t eta_pm2 =aTrack22.Eta();
Double_t  KStarInvM_pm2 =    pow((pow((EK_pipl[m]
+EK_kmn[l]),2) - (pow((px_pipl[m]+px_kmn[l]),2)+
pow((py_pipl[m]+py_kmn[l]),2) +pow((pz_pipl[m]+
pz_kmn[l]),2))),0.5);
//hKStarMInv_pm2->Fill(KStarInvM_pm2,pt_pm2,phi_pm2);
hKStarMInv_pm2->Fill(KStarInvM_pm2);
hKStar_pt_pm2->Fill(pt_pm2);
hKStar_phi_pm2->Fill(phi_pm2);
//cout <<"K- Pi+ selctd" <<endl;
```

```
      }//l loop
       }//m loop same event loop ends K- Pi+
//for plus plus invariant mass
//same event calculation K+ Pi+
for(Int_t m=0; m<g; m++)
  {
    for(Int_t l=0; l<i; l++)
      {
aTrack23.SetXYZ((px_pipl[m]+px_kpl[l]),(py_pipl[m]+py_kpl[l]),
(pz_pipl[m]+pz_kpl[l]));
Double_t phi_pp = aTrack23.Phi();
Double_t pt_pp = aTrack23.Pt();
Double_t eta_pp =aTrack23.Eta();
Double_t  KStarInvM_pp =    pow((pow((EK_pipl[m]+EK_kpl[l])
,2) - (pow((px_pipl[m]+px_kpl[l]),2)+pow((py_pipl[m]
+py_kpl[l]),2) +pow((pz_pipl[m]+pz_kpl[l]),2))),0.5);
hKStarMInv_pp->Fill(KStarInvM_pp);
hKStar_pt_pp->Fill(pt_pp);
hKStar_phi_pp->Fill(phi_pp);
//cout <<"K+ Pi+ selctd" <<endl;
      }//l loop
       }//m loop same event loop ends K+ Pi+
//for minus minus invariant mass
//same event calculation K- Pi-
for(Int_t m=0; m<h; m++)
  {
```

```
    for(Int_t l=0; l<j; l++)
      {
aTrack24.SetXYZ((px_pimn[m]+px_kmn[l]),(py_pimn[m]
+py_kmn[l]),(pz_pimn[m]+pz_kmn[l]));
Double_t phi_mm = aTrack24.Phi();
Double_t pt_mm = aTrack24.Pt();
Double_t eta_mm =aTrack24.Eta();
Double_t  KStarInvM_mm =    pow((pow((EK_pimn[m]+
EK_kmn[l]),2) - (pow((px_pimn[m]+px_kmn[l]),2)+
pow((py_pimn[m]+py_kmn[l]),2) +pow((pz_pimn[m]+
pz_kmn[l]),2))),0.5)   ;
hKStarMInv_mm->Fill(KStarInvM_mm);
hKStar_pt_mm->Fill(pt_mm);
hKStar_phi_mm->Fill(phi_mm);
      }//l loop
       }//m loop same event loop ends K- Pi-
      } // Event loop ends
         cout<<" Total Entries  = "<<count1<<endl;
cout<<" Entries without pt=0,  = "<<count2<<endl;
cout<<" Entries without pt=0 and eta<1  = "<<count3<<endl;
    TFile *f = new TFile("Ampt_phi_v2_50000.root","RECREATE");
    f->cd();
    hKStarMInv_pp->Write();
    hKStarMInv_mm->Write();
    hKStarMInv_pm1->Write();
    hKStarMInv_pm2->Write();
```

```
    himp->Write();
    f->Close();
cout<<" Program End... EXIT...\n";
}
```

A.1.4 Macro - 4

Root Macro for calculating the signal distribution, plotting it and fitting with a Breit-Wigner distribution function.

```
#include <iostream>
#include <TObject.h>
#include <TTree.h>
#include <TFile.h>
#include <TCanvas.h>
#include <TH1F.h>
#include <TH2F.h>
#include <TPaveText.h>
#include <TStyle.h>
#include <TPaletteAxis.h>
#include <TColor.h>
#include <TF1.h>
#include <TMath.h>

void plot()
{
```

```
gStyle->SetOptFit(1);
gStyle->SetOptStat("e");
  //put the fit range here
  Float_t FitMin = 0.7;
  Float_t FitMax = 1.15;

TFile *f=new TFile("Ampt_phi_v2_50000.root","READ");
TH1F *hist_pp=(TH1F*)f->Get("hKStarMInv_pp");
TH1F *hist_mm=(TH1F*)f->Get("hKStarMInv_mm");
TH1F *hist_pm1=(TH1F*)f->Get("hKStarMInv_pm1");
TH1F *hist_pm2=(TH1F*)f->Get("hKStarMInv_pm2");
 hist_pp->Sumw2();
 hist_mm->Sumw2();
 hist_pm1->Sumw2();
 hist_pm2->Sumw2();

TCanvas *Can1=new TCanvas("Can1","Unlike sign Histograms"
,100,100,1000,500);
TCanvas *Can2=new TCanvas("Can2","Like Sign Histograms",
100,100,1000,500);
TH1F *hist_Sig=(TH1F*)hist_pm1->Clone(); //Cloning
hist_Sig->SetName("hist_Sig");
hist_Sig->Add(hist_pm2,1);
hist_pp->Add(hist_mm,1);
Can1->cd();
hist_Sig->SetTitle("Invariant mass of K^{+} #pi^{-} and
```

```
K^{-} #pi^{+}");
hist_Sig->GetXaxis()->SetTitle("Invariant mass
(in GeV/c^{2})");
hist_Sig->GetXaxis()->CenterTitle();
hist_Sig->GetYaxis()->SetTitle("Events");
hist_Sig->GetYaxis()->CenterTitle();
hist_Sig->SetMarkerStyle(3);
hist_Sig->Draw("L");
Can2->cd();
hist_pp->SetTitle("Invariant mass of K^{+} #pi^{+}
and K^{-} #pi^{-}");
hist_pp->GetXaxis()->SetTitle("Invariant mass
(in GeV/c^{2})");
hist_pp->GetXaxis()->CenterTitle();
hist_pp->GetYaxis()->SetTitle("Events");
hist_pp->GetYaxis()->CenterTitle();
hist_pp->SetMarkerStyle(3);
hist_pp->Draw("L");

TH1F *hist_Sig2=(TH1F*)hist_Sig->Clone();  //Cloning
hist_Sig2->SetName("hist_Sig2");
hist_Sig2->Add(hist_pp,-1);

TCanvas *Can3=new TCanvas("Can3","Histogram for Signal",
100,100,1000,500);
Can3->cd();
```

```
hist_Sig2->SetTitle("K* Signal Distribution for NTMAX = 3");
hist_Sig2->Rebin(10);
hist_Sig2->GetXaxis()->SetRangeUser(0.6,1.2);
hist_Sig2->GetXaxis()->SetTitle("Invariant mass
(in GeV/c^{2})");
hist_Sig2->GetXaxis()->CenterTitle();
hist_Sig2->GetYaxis()->SetTitle("Events");
hist_Sig2->GetYaxis()->CenterTitle();
hist_Sig2->Draw();
hist_Sig2->SetMarkerStyle(24);
//Function is Breit Wigner + linear residual background
 (see "SBW" below)
 TF1 *funFit;
 funFit = new TF1("funFit",SBW, FitMin, FitMax, 5);
 funFit->SetParNames("Yield","Mass","Width","Slope","Const");
 funFit->SetParameters(4000,0.890,0.05,-5000,6000);
 funFit->SetParLimits(0,1.0e+0,1.0e+10);
 funFit->SetParLimits(1,0.88,0.91);
 funFit->SetParLimits(2,0.047,0.055);
 //funFit->FixParameter(1,0.896);
 //funFit->FixParameter(2,0.05);

 hist_Sig2->Fit(funFit,"ERI");
 cout << "Chi2/NDF   "<<funFit->GetChisquare()
 /funFit->GetNDF()<<endl;
  //this is to draw the limear backgroud
```

```
 TF1 *background;
 background = new TF1("background",Poly1, FitMin,
 FitMax, 2);
 background->SetParameters(funFit->GetParameter(4),
  funFit->GetParameter(3));
 background->SetLineColor(kBlue);
 background->SetLineStyle(2);
 background->Draw("same l");
cout<<"Fitted Parameters : \n\n"<<endl;
cout<<"Yield = "<<funFit->GetParameter(0)<<"  +-
"<<funFit->GetParError(0)<<endl;
cout<<"Mass = "<<funFit->GetParameter(1)<<"  +-
 "<<funFit->GetParError(1)<<endl;
cout<<"WIdth = "<<funFit->GetParameter(2)<<"  +-
 "<<funFit->GetParError(2)<<endl;
cout<<"Linear Bkg Slope = "<<funFit->GetParameter(3)<<"
 +-  "<<funFit->GetParError(3)<<endl;
cout<<"Linear Bkg Intercept = "<<funFit->GetParameter(4)
<<"  +-  "<<funFit->GetParError(4)<<endl;


}
Double_t SBW(Double_t *x, Double_t *par)
{
  //(BreitWigner + (A + B*M)
  return par[0]*par[2]*0.001*10/(2*3.14159)/((x[0]-par[1])
  **2+par[2]**2/4.) + par[4]+par[3]*x[0];
```

```
}
Double_t Poly1(Double_t *x, Double_t *par)
{
  return par[0]+par[1]*x[0];
}
```

A.1.5 Macro - 5

Root Macro for plotting the yield dependence on τ_{HC}.

```
#include <iostream>
#include <TObject.h>
#include <TTree.h>
#include <TFile.h>
#include <TCanvas.h>
#include <TH1F.h>
#include <TH2F.h>
#include <TPaveText.h>
#include <TStyle.h>
#include <TPaletteAxis.h>
#include <TColor.h>
#include <TF1.h>
#include <TMath.h>

void compare_plot()
{
```

```
gStyle->SetOptStat("");
  //put the fit range here
  Float_t FitMin = 0.7;
  Float_t FitMax = 1.15;

TFile *f=new TFile("compare_plot.root","READ");

TF1 *fit_5=(TF1*)f->Get("signal_Ntmax5");
TF1 *fit_10=(TF1*)f->Get("signal_Ntmax10");
TF1 *fit_20=(TF1*)f->Get("signal_Ntmax20");
TF1 *fit_30=(TF1*)f->Get("signal_Ntmax30");
TF1 *fit_40=(TF1*)f->Get("signal_Ntmax40");
TF1 *fit_150=(TF1*)f->Get("signal_Ntmax150");

TCanvas *Can2=new TCanvas("Can2","Histograms for K* Signal
 at different #tau"
,100,100,1000,500);
Can2->cd();
fit_5->SetTitle("Effect of Fireball Lifetime on Yield");
fit_5->GetXaxis()->SetRangeUser(0.75,1.05);
fit_5->GetXaxis()->SetTitle("Invariant mass (in GeV/c^{2})");
fit_5->GetYaxis()->SetTitle("Events");
fit_5->GetXaxis()->CenterTitle();
fit_5->GetYaxis()->CenterTitle();
fit_5->Draw();
fit_10->Draw("same");
```

```
fit_20->Draw("same");
fit_30->Draw("same");
fit_40->Draw("same");
fit_5->SetLineColor(2);
fit_10->SetLineColor(3);
fit_20->SetLineColor(4);
fit_30->SetLineColor(6);
fit_40->SetLineColor(7);

TLegend *leg=new TLegend(0.65,0.5,0.89,0.8);
leg->AddEntry(fit_5,"for #tau = 1 fm/c","L");
leg->AddEntry(fit_10,"for #tau = 2 fm/c","L");
leg->AddEntry(fit_20,"for #tau = 4 fm/c","L");
leg->AddEntry(fit_30,"for #tau = 6 fm/c","L");
leg->AddEntry(fit_40,"for #tau = 8 fm/c","L");
leg->Draw("same");
}
```

A.2 APPENDIX - II

A.2.1 Mathematica Code - 1

Mathematica code for modeling of rescattering and regeneration.

```
(* rescatter estimates the change in the number of resonance R
 due to rescattering and regeneration effect as R moves through
 hot hadronic medium at the LHC energies where muB=0. The medium
```

```
  is assumed to be made of pi K and N*)
(* All quantities in the entire code if not mentioned are in MeV *)

(* Setting some numbers *)
(* 25*10^(-7) is the conversion from mb to MeV^(-2)*)
x=20 25 10^(-7);
y=40 25 10^(-7);

(* Cross sections: for eg. sigmapipi
= cross section between pi pi  *)
sigmapipi=x;
sigmapiK=x;
sigmapiN=y;
sigmaKpi=sigmapiK;
sigmaKK=x;
sigmaKN=y;
(* some functions *)
mat={};
(* number density *)
Density[g_,mass_,T_]:=g/(2 Pi^2) mass^2 T N[BesselK[2,mass/T]];

(* Rate of scattering = crossection*(no. of scatterers)
*velocity *)
RateAKFO[sigmaApi_,sigmaAkaon_,sigmaAnucleon_,densitypion_
,densitykaon_,
densitynucleon_,R_,v_,vA_,t1_,t2_]:=(sigmaApi densitypion
```

```
+sigmaAkaon
 densitykaon+sigmaAnucleon densitynucleon)
 (R/(R+v (t2+t1)/2))^3 vA;
(* Main function *)

(* only decay (100% opaque fireball, thus K* number depends
 directly
 on number of decays that take place) *)
rescatteronlydecay[Rdecay_,NR0_,t_]:=NR0 Exp[-Rdecay t];

(* realistic decay, rescatter, regeneration *)
rescatter[Rmass_, Amass_,Bmass_,Rdecay_,esigmaApi_,esigmaAkaon_
,esigmaAnucleon_,esigmaBpi_,
esigmaBkaon_,esigmaBnucleon_,inesigmaApi_,inesigmaAkaon_,
inesigmaAnucleon_,
inesigmaBpi_,inesigmaBkaon_,inesigmaBnucleon_,NR0_,VolumeCFO_
,TCFO_,t_,tint_]
:=Module[{pstar,vflow,vA,vB,Radius,densitypion,densitykaon
,densitynucleon,eRateA,
eRateB,ineRateA,ineRateB,NRt1,NRt2,NAt1,NAt2,NBt1,NBt2,NA1
,NA2,NB1,NB2,t1,t2,k1,
k2,k3,k4,k5,kA,kB,nrt},
pstar=1/(2 Rmass)Sqrt[(Rmass^2-(Amass+Bmass)^2)
(Rmass^2-(Amass-Bmass)^2)];
vflow=0.5;
vA=pstar/Amass;
```

```
vB=pstar/Bmass;
Radius=(VolumeCFO/(4/3 Pi))^(1/3);
densitypion=Density[9,140,TCFO];
densitykaon=Density[8,495,TCFO];
densitynucleon=Density[40,1000,TCFO];
eRateA[ta_,tb_]:=RateAKFO[esigmaApi,esigmaAkaon,
esigmaAnucleon,
densitypion,densitykaon,densitynucleon,Radius
,vflow,vA,ta,tb];
eRateB[ta_,tb_]:=RateAKFO[esigmaBpi,
esigmaBkaon,esigmaBnucleon,
densitypion,densitykaon,densitynucleon,
Radius,vflow,vB,ta,tb];
(*check inelastic exprsn*)
ineRateA[ta_,tb_]:=RateAKFO[inesigmaApi,
inesigmaAkaon,inesigmaAnucleon
,densitypion,densitykaon,densitynucleon
,Radius,vflow,vA,ta,tb];
ineRateB[ta_,tb_]:=RateAKFO[inesigmaBpi,
inesigmaBkaon,inesigmaBnucleon,
densitypion,densitykaon,densitynucleon,
Radius,vflow,vB,ta,tb];

t1=0;
t2=tint;
NRt1=NR0;
```

```
NAt1=0;

NBt1=0;

nrt={};

Do[

k1=Rdecay;

k2=eRateA[t1,t2];

k3=ineRateA[t1,t2];

k4=eRateB[t1,t2];

k5=ineRateB[t1,t2];

kA=k2+k3-k1;

kB=k4+k5-k1;


NA1=(NAt1/NRt1-k1/kA) Exp[kA t1];

NA2=k1/kA;

NB1=(NBt1/NRt1-k1/kB) Exp[kB t1];

NB2=k1/kB;


NRt2=NRt1 Exp[(-k1+k3 NA2+k5 NB2)

(t2-t1)-k3/kA NA1

(Exp[-kA t2]-Exp[-kA t1])-k5/kB NB1

(Exp[-kB t2]-Exp[-kB t1])];

NAt2=NRt2 ((NAt1/NRt1-k1/kA) Exp[-kA

 (t2-t1)]+k1/kA);

NBt2=NRt2 ((NBt1/NRt1-k1/kB) Exp[-kB

(t2-t1)]+k1/kB);
```

```
t1=t2;
t2=t2+tint;
NRt1=NRt2;
NAt1=NAt2;
NBt1=NBt2;
nrt=Join[nrt,{{200 t1,NRt1,NAt1,NBt1}}];
mat=Join[mat,{{200 t1,NRt1,NAt1,NBt1}}];
,{i,t/tint}];
{t1,NRt1,NAt1,NBt1,nrt}
];

kstarbyk1=rescatter[892, 140,496,50,sigmapipi
,sigmapiK,sigmapiN,sigmaKpi,
sigmaKK,sigmaKN,0,sigmapiK,0,sigmaKpi,0,0,1,
5 10^(-4),155,10/200,0.01/200];
(* Plotting *)
kstarbyk0pl=Plot[rescatteronlydecay[50,1,x/200]
,{x,0,10},PlotStyle->
{Black,Dashed}];
kstarbyk1pl=ListPlot[{Table[{kstarbyk1[[5,i,1]],
kstarbyk1[[5,i,2]]},{i,Length[kstarbyk1[[5]]]}],
Table[{kstarbyk1[[5,i,1]],kstarbyk1[[5,i,3]]},
{i,Length[kstarbyk1[[5]]]}],Table
[{kstarbyk1[[5,i,1]],
kstarbyk1[[5,i,4]]},{i,Length[kstarbyk1[[5]]]}],
Table[{kstarbyk1[[5,i,1]],kstarbyk1[[5,i,2]]
```

```
+kstarbyk1[[5,i,3]]},{i,Length[kstarbyk1[[5]]]}]},
Joined->True,Frame->True,
FrameLabel->{"Time (in fm/c)","Ratio"},Axes->False,
PlotStyle->{Blue,Red,Green,Brown},PlotRange->{{0,10},
{0,1.5}}];

m1l=Graphics[{Red,Line[{{0,1},{0.2,1}}]}];
m2l=Graphics[{Blue,Line[{{0,1},{0.2,1}}]}];
m3l=Graphics[{Green,Line[{{0,1},{0.2,1}}]}];
m4l=Graphics[{Brown,Line[{{0,1},{0.2,1}}]}];
m5l=Graphics[{Black,Dashed,Line[{{0,1},{0.2,1}}]}];
(* Plot of normalised (with respect to Initial K* yield)
 K*
and daughter pion and kaon with time elapsed since
chemical
 freezeout *)
kstarratioplot=Show[kstarbyk1pl,kstarbyk0pl,Epilog->
{Inset[ "K^*+
 Daughter  \[Pi]",{6.5,1.35}],Inset[ m4l,{9,1.35}]
 ,Inset["K^*",
 {7,1.25}],Inset[m2l,{9,1.25}],Inset["K^*
 (100% opaque fireball)",
 {6,1.15}],Inset[m5l,{9,1.15}],
 Inset[ "Daughter K",{7,1.05}],Inset[m3l,{9,1.05}],
 Inset[ "Daughter \[Pi]",{7,0.95}],Inset[m1l,{9,0.95}]}]
```

```
Export["kstarratioplot.eps",kstarratioplot];
Export["F:\proj\Thesis project\Rescattering and regeneration
\out.txt",mat,"Table"]
```

A.2.2 Macro - 1

Root Macro for plotting results of Mathematica code and generate yield ratio vs τ plot.

```
 #include <iostream>
#include <TObject.h>
#include <TTree.h>
#include <TFile.h>
#include <TCanvas.h>
#include <TH1F.h>
#include <TH2F.h>
#include <TPaveText.h>
#include <TStyle.h>
#include <TPaletteAxis.h>
#include <TColor.h>
#include <TF1.h>
#include <TMath.h>

void plot_rescatter()
{
Int_t i=0;
```

```
Double_t t[2000],NR[2000],NA[2000],NB[2000];
ifstream fin;
char *fname = new char[100];
sprintf(fname,"/home/arabinda/prog/out.txt");
fin.open(fname);
cout<<fname<<endl;
while(fin)
{
fin>>t[i]>>NR[i]>>NA[i]>>NB[i];
 i++;
if(fin.eof()) break;
}
fin.close();
cout<<"Total data points = "<<i<<endl;
cout<<NA[i];
TGraph *graph1=new TGraph(i,t,NR);
graph1->SetTitle("Modeling of Rescattering and Regeneration");
graph1->GetYaxis()->SetRangeUser(0,1);
graph1->GetYaxis()->SetTitle("Ratio");
graph1->GetYaxis()->CenterTitle();
graph1->GetXaxis()->SetRangeUser(0,10);
graph1->GetXaxis()->SetTitle("Time (in fm/c)");
graph1->GetXaxis()->CenterTitle();
TGraph *graph2=new TGraph(i,t,NB);
TGraph *graph3=new TGraph(i,t,NA);
graph1->SetMarkerStyle(5);
```

```
graph2->SetMarkerStyle(3);
graph3->SetMarkerStyle(3);
graph1->SetLineColor(kRed);
graph1->SetLineWidth(4);
graph2->SetLineColor(kBlue);
graph2->SetLineWidth(4);
graph3->SetLineColor(kGreen);
graph3->SetLineWidth(4);
TCanvas *Can1=new TCanvas("Can1","Histograms",100,100,1000,600);
Can1->cd();
graph1->Draw("AC");
graph2->Draw("same");
graph3->Draw("same");
TLegend *leg=new TLegend(0.7,0.8,0.89,0.6);
leg->AddEntry(graph1,"K^{*0}","L");
leg->AddEntry(graph2,"Daughter K","L");
leg->AddEntry(graph3,"Daughter #pi","L");
leg->Draw("same");
}
```

A.3 APPENDIX-III

A.3.1 Invariant Mass of K^{*0}

```
#define star_uu_193_new_cxx
```

```
#include "star_uu_193_new.h"
#include <TH2.h>
#include <TStyle.h>
#include <TCanvas.h>
//Don't change:
Int_t        runID,Refmult,BBC_cons,ZDC_cons,centrality,ntrack;
Double_t     vertexX,vertexY,vertexZ,VPD_vertexZ;
//Don't change:
Int_t       track_charge,track_nHitsFit,track_nHitsdEdx,track_ToFflag;
Double_t nsigma_pi,nsigma_k,nsigma_p,nsigma_e;
Double_t track_px,track_py,track_pz,track_dca,track_dEdx,track_beta;
Long64_t nEvent_total = 0;
Double_t m2,p,p2,fitratio,pt,eta1,m_k=0.493, m_pi=0.139;
TVector3 aTrack;
//Pion and kaon energy matrices
Double_t EK_pipl[50000]={0.},EK_pimn[50000]={0.};
Double_t EK_kmn[50000]={0.},EK_kpl[50000]={0.};
//Pion and kaon momentum matrices
Double_t   px_pipl[50000],py_pipl[50000],pz_pipl[50000];
Double_t   px_pimn[50000],py_pimn[50000],pz_pimn[50000];
Int_t charge_pipl[50000], charge_pimn[50000];
Double_t   px_kpl[50000],py_kpl[50000],pz_kpl[50000];
Double_t   px_kmn[50000],py_kmn[50000],pz_kmn[50000];
Int_t charge_kpl[50000], charge_kmn[50000];
void star_uu_193_new::Loop()
{
```

```
TH1F *hRefMult=new TH1F("hRefMult","Reference Multiplicity",
1000,0.,1000);
TH1D *hKStarMInv_pp=new TH1D("hKStarMInv_pp","Inv mass dist
 for K+ Pi+",900,0.6,1.5);
TH1D *hKStarMInv_pm=new TH1D("hKStarMInv_pm","Inv mass dist
 for K+ Pi-",900,0.6,1.5);
TH1D *hKStarMInv_mp=new TH1D("hKStarMInv_mp","Inv mass dist
 for K- Pi+",900,0.6,1.5);
TH1D *hKStarMInv_mm=new TH1D("hKStarMInv_mm","Inv mass dist
 for K- Pi-",900,0.6,1.5);
TH1D *hNsigmapipl=new TH1D("hNsigmapipl","NSigma for pi+",
120,-3,3);
TH1D *hNsigmapimn=new TH1D("hNsigmapimn","NSigma for pi-",
120,-3,3);
TH1D *hcharge_pipl=new TH1D("hcharge_pipl","Charge for pi+",
10,-5,5);
TH1D *hcharge_pimn=new TH1D("hcharge_pimn","Charge for pi-"
,10,-5,5);
TH1D *hPt_kstar_unlike=new TH1D("hPt_kstar_unlike","K* Pt
 distribution",100,0.0,10.0);
TH1D *hPt_kstar_like=new TH1D("hPt_kstar_like","K* Pt
distribution",100,0.0,10.0);
if (fChain == 0) return;

Long64_t nentries = fChain->GetEntriesFast();
Long64_t nbytes = 0, nb = 0;
```

```
TStopwatch sw1,sw2;
 sw1.Start(1);
 sw2.Start(1);

 //Event loop :
 for (Long64_t jentry=0; jentry<nentries;jentry++)
   {//event loop starts
     Long64_t ientry = LoadTree(jentry);
     if (ientry < 0) break;
     nb = fChain->GetEntry(jentry);   nbytes += nb;
     runID       =                 RunID;
     Refmult     = (int)            Refm;
     BBC         = (int)             BBC;
     ZDC         = (int)             ZDC;
     centrality  = (int)            Cent;
     ntrack      = (int)          nTrack;
     vertexX     = (double)           Vx;
     vertexY     = (double)           Vy;
     vertexZ     = (double)           Vz;
     VPD_vertexZ = (double)        VpdVz;
    //Event Cuts
     //      if(centrality==0) continue;
     if(TMath::Abs(vertexZ)>=30.) continue;
     if(TMath::Abs(sqrt(vertexX*vertexX+vertexY*vertexY))
     >=2.) continue;
```

```
      hRefMult->Fill(Refmult);
      Int_t g=0,h=0,i=0,j=0;
      for(int it=0;it<ntrack;it++){
      track_px    = (double)      Px[it];
        track_py    = (double)      Py[it];
        track_pz    = (double)      Pz[it];
        track_dca   = (double)     DCA[it];
        track_beta  = (double)    Beta[it];
        nsigma_pi   = (double)   nSigpi[it];
        nsigma_k    = (double)    nSigk[it];
        nsigma_p    = (double)    nSigp[it];
    if(DEdx[it]>=0){
          track_dEdx  =  (double) DEdx[it];
        }
        else if(DEdx[it]<0){
          track_dEdx = (double) DEdx[it] * (-1.0);
        }
        track_charge      = (int)     Charge[it];
        if(!track_charge)    track_charge = -1.0;
        track_nHitsFit    = (int)      nHits[it];
        track_nHitsdEdx   = (int)      ndEdx[it];
        track_ToFflag     = (int)     tofFlg[it];
if(track_nHitsFit<=15) continue;
if(TMath::Abs(track_charge)>1.) continue;
        if(track_dca>3) continue;
    aTrack.SetXYZ(track_px,track_py,track_pz);
```

```
        eta1=aTrack.Eta();
if(TMath::Abs(eta1)>0.5) continue;
        pt=aTrack.Perp();
if(pt<=0.2 || pt>=10.0) continue;
p=aTrack.Mag();
p2=p*p;
m2=p2*((1/pow(track_beta,2))-1);
//now fill your hists/ do your thing here:
p2=pow(track_px,2)+pow(track_py,2)+pow(track_pz,2);
m2=p2*((1/pow(track_beta,2))-1);
aTrack.SetXYZ(track_px,track_py,track_pz);
pt=aTrack.Perp();
eta1=aTrack.Eta();
if(pt<=0.2 || pt>=10.0) continue;
        if(TMath::Abs(eta1) >= 0.5) continue;
     if(TMath::Abs(nsigma_pi)<2.0)
  {
      if(m2>0.010382 && m2<=0.029218)       {
if(track_charge==1)
  {// Pion plus selection starts
    EK_pipl[g] = pow((p2+0.019321),0.5);
    px_pipl[g]=track_px;
    py_pipl[g]=track_py;
    pz_pipl[g]=track_pz;
    g++;
  }// Pion plus selection ends
```

```
if(track_charge==-1)
  {// Pion minus selection starts
    EK_pimn[h] = pow((p2+0.019321),0.5);
    px_pimn[h]=track_px;
    py_pimn[h]=track_py;
    pz_pimn[h]=track_pz;
        h++;
  }// Pion minus selection ends
      }
      }//Pion selection ends

//Kaon selection
if(TMath::Abs(nsigma_k)<=2.0)
  {
    if(m2>=0.2104 && m2<=0.2796)
      {
if(track_charge==1)
  {// Kaon plus selection starts
    EK_kpl[i] = pow((p2+0.243049),0.5);
    px_kpl[i]=track_px;
    py_kpl[i]=track_py;
    pz_kpl[i]=track_pz;
    //charge_kpl[i]=track_charge;
    i++;
  }// Kaon plus selection ends
```

```
if(track_charge==-1)
  {// Kaon minus selection starts
    EK_kmn[j] = pow((p2+0.243049),0.5);
    px_kmn[j]=track_px;
    py_kmn[j]=track_py;
    pz_kmn[j]=track_pz;
    //charge_kmn[j]=track_charge;
    j++;
  }// Kaon minus selection ends
      }
  }//Kaon selection ends


      }//track loop ends
      //cout<<"g = "<<g<<" , h = "<<h<<" , i = "<<i<<" ,
      j = "<<j<<endl;

      //-----------------------------------------------------------------

                              //K* reconstruction

      //-----------------------------------------------------------------

      TLorentzVector Pion,Kaon,K_star;

      //Unlike Sign
      //Same event calculation for K+ pi-
```

```
      for(Int_t m=0; m<h; m++)
        {
          Pion.SetXYZM(px_pimn[m],py_pimn[m],pz_pimn[m],
          m_pi);
  Double_t pieta = Pion.PseudoRapidity();
  for(Int_t l=0; l<i; l++)
            {
      Kaon.SetXYZM(px_kpl[l],py_kpl[l],pz_kpl[l],
      m_k);
      K_star=Kaon+Pion;

      Double_t  KStarInvM_pm = K_star.M();
      pt=K_star.Pt();
      hPt_kstar_unlike->Fill(pt);
      eta1=K_star.PseudoRapidity();
      hKStarMInv_pm->Fill(KStarInvM_pm);

    }//l loop
}//m loop

      //Same event calculation for K- pi+
      for(Int_t m=0; m<g; m++)
        {
          Pion.SetXYZM(px_pipl[m],py_pipl[m],pz_pipl[m]
          ,m_pi);
  Double_t pieta = Pion.PseudoRapidity();
```

```
          for(Int_t l=0; l<j; l++)
            {
      Kaon.SetXYZM(px_kmn[l],py_kmn[l],pz_kmn[l]
      ,m_k);
      K_star=Kaon+Pion;

      Double_t  KStarInvM_mp = K_star.M();
      pt=K_star.Pt();
      hPt_kstar_unlike->Fill(pt);
      eta1=K_star.PseudoRapidity();
      hKStarMInv_mp->Fill(KStarInvM_mp);

    }//l loop
}//m loop

      //Like Sign
      //Same event calculation for K+ pi+
      for(Int_t m=0; m<g; m++)
        {
  Pion.SetXYZM(px_pipl[m],py_pipl[m],pz_pipl[m]
  ,m_pi);
          for(Int_t l=0; l<i; l++)
            {
              //if(charge_pipl[m]+charge_kpl[l]==0)
              continue;
      Kaon.SetXYZM(px_kpl[l],py_kpl[l],pz_kpl[l
```

```
      ],m_k);
      K_star=Kaon+Pion;

      Double_t  KStarInvM_pp = K_star.M();
      pt=K_star.Pt();
      hPt_kstar_like->Fill(pt);
      eta1=K_star.PseudoRapidity();
      hKStarMInv_pp->Fill(KStarInvM_pp);
    }//l loop
}//m loop

      //Same event calculation for K- pi-
      for(Int_t m=0; m<h; m++)
        {
  Pion.SetXYZM(px_pimn[m],py_pimn[m],pz_pimn[m],
  m_pi);
         for(Int_t l=0; l<j; l++)
           {
             //if(charge_pimn[m]+charge_kmn[l]==0)
              continue;
      Kaon.SetXYZM(px_kmn[l],py_kmn[l],pz_kmn[l]
      ,m_k);
      K_star=Kaon+Pion;

      Double_t  KStarInvM_mm = K_star.M();
      pt=K_star.Pt();
```

```
      hPt_kstar_like->Fill(pt);
              eta1=K_star.PseudoRapidity();
      hKStarMInv_mm->Fill(KStarInvM_mm);
    }//l loop
}//m loop



      nEvent_total++;
      //screen print:
      if(nEvent_total%3000==0)
       {
 cout<<" Analyzed Event = "<<nEvent_total<<" cent
  = "<<centrality<<" ntrack = "<<ntrack<<endl;
 sw2.Stop();
 sw2.Print();
 sw2.Start(1);
       }


    }//================event loop ends=======================;
  //   WriteHist();


  cout<<" writing files......."<<endl;
  f = new TFile(outfile,"recreate");
```

```
  f->cd();


  hRefMult->Write();
  hKStarMInv_pp->Write();
  hKStarMInv_pm->Write();
  hKStarMInv_mp->Write();
  hKStarMInv_mm->Write();
  hPt_kstar_unlike->Write();
  hPt_kstar_like->Write();
  f->Close();
  cout<<"\n File written.\n ******** Events = "
  <<nEvent_total<<"********\n"<<endl;


  sw2.Stop();
  sw1.Stop();
  sw1.Print();


}//main code ends
```

A.3.2 Event Planes

```
#define star_uu_193_new_cxx
#include "star_uu_193_new.h"
#include <TH2.h>
#include <TStyle.h>
#include <TCanvas.h>
#include<TVector.h>
```

```
#include "run.h"
Int_t        runID,Refmult,BBC_cons,ZDC_cons,centrality,
ntrack;
Double_t     vertexX,vertexY,vertexZ,VPD_vertexZ;
Int_t       track_charge,track_nHitsFit,track_nHitsdEdx,
track_ToFflag;
Double_t nsigma_pi,nsigma_k,nsigma_p,nsigma_e;
Double_t track_px,track_py,track_pz,track_dca,track_dEdx
,track_beta;
Long64_t nEvent_total = 0;
Double_t m,m2,p2,x2,y2,qx,qy,qx_east,qy_east,qx_west,
qy_west,Psi2,
Psi2_east,Psi2_west,Phi,fitratio,pt,eta1,eta2;
Float_t pi=3.1416;
Int_t n_flag=783,index;
TVector3 aTrack;
TVector2 aTrack2,aTrack3;
char input_rootfile[250];
void star_uu_193_new::Loop()
{
  sprintf(input_rootfile,"/Users/ranbirsingh/arabinda
  /flow/output/event_planes/recenter/file_all.root");
  cout<<"Input file joint = "<<input_rootfile<<endl;
  TFile *f=new TFile(input_rootfile,"READ");
  //Fullrange eta
  TProfile2D *QX_fw=(TProfile2D*)f->Get("QX_fw");
```

```
TProfile2D *QX_w=(TProfile2D*)f->Get("QX_w");
TProfile2D *QX_fe=(TProfile2D*)f->Get("QX_fe");
TProfile2D *QX_e=(TProfile2D*)f->Get("QX_e");
TProfile2D *QY_fw=(TProfile2D*)f->Get("QY_fw");
TProfile2D *QY_w=(TProfile2D*)f->Get("QY_w");
TProfile2D *QY_fe=(TProfile2D*)f->Get("QY_fe");
TProfile2D *QY_e=(TProfile2D*)f->Get("QY_e");
//Subrange eta
TProfile2D *QXs_fw=(TProfile2D*)f->Get("QXs_fw");
TProfile2D *QXs_w=(TProfile2D*)f->Get("QXs_w");
TProfile2D *QXs_fe=(TProfile2D*)f->Get("QXs_fe");
TProfile2D *QXs_e=(TProfile2D*)f->Get("QXs_e");
TProfile2D *QYs_fw=(TProfile2D*)f->Get("QYs_fw");
TProfile2D *QYs_w=(TProfile2D*)f->Get("QYs_w");
TProfile2D *QYs_fe=(TProfile2D*)f->Get("QYs_fe");
TProfile2D *QYs_e=(TProfile2D*)f->Get("QYs_e");
// define here:
TH1D *hPsi2=new TH1D("hPsi2","Event Plane Angle
#Psi_{2}",200,-1.,5.);
TH1D *hPsi2_east=new TH1D("hPsi2_east","Event Plane
Angle #Psi_{2} for #eta subrange(0.05 to 1.0)",200,-1.,5.);
TH1D *hPsi2_west=new TH1D("hPsi2_west","Event Plane
Angle #Psi_{2}for #eta subrange(-1.0 to-0.05 )",200,-1.0,5.);
TH2D *hQxQy=new TH2D("hQxQy","Histogram of Qx vs Qy
after recenter correction",500,-100.,100.,500,-100.,100.);
TH1F *hRefMult=new TH1F("hRefMult","Reference
```

```
 Multiplicity",1000,0.,1000);
//define TProfile for Shift correction factors
 TProfile *Sin_corr=new TProfile("Sin_corr",
 "Profile Histogram of Sin_corr vs i",25,0,25,-2.,2.);
 TProfile *Cos_corr=new TProfile("Cos_corr",
 "Profile Histogram of Cos_corr vs i",25,0,25,-2.,2.);
 TProfile *Sin_corr_east=new TProfile
 ("Sin_corr_east","Profile Histogram of
  Sin_corr_east vs i",25,0,25,-2.,2.);
 TProfile *Cos_corr_east=new TProfile
 ("Cos_corr_east","Profile Histogram of Cos_corr_east
 vs i",25,0,25,-2.,2.);
 TProfile *Sin_corr_west=new TProfile("Sin_corr_west",
 "Profile Histogram of Sin_corr_west vs i",25,0,25,-2.,2.);
 TProfile *Cos_corr_west=new TProfile("Cos_corr_west",
 "Profile Histogram of Cos_corr_west vs i",25,0,25,-2.,2.);
 if (fChain == 0) return;
Long64_t nentries = fChain->GetEntriesFast();
 Long64_t nbytes = 0, nb = 0;
 TStopwatch sw1,sw2;
 sw1.Start(1);
 sw2.Start(1);
 for (Long64_t jentry=0; jentry<nentries;jentry++)
   {//event loop starts
     Long64_t ientry = LoadTree(jentry);
     if (ientry < 0) break;
```

```
      nb = fChain->GetEntry(jentry);   nbytes += nb;
      runID       =                  RunID;
      Refmult     = (int)             Refm;
      BBC         = (int)              BBC;
      ZDC         = (int)              ZDC;
      centrality  = (int)             Cent;
      ntrack      = (int)           nTrack;
      vertexX     = (double)            Vx;
      vertexY     = (double)            Vy;
      vertexZ     = (double)            Vz;
      VPD_vertexZ = (double)         VpdVz;
      for(Int_t j=0;j<n_flag;j++)
        {
          if(numbers[j]==runID)
            index=j;
        }
      if(TMath::Abs(vertexZ)>=30.) continue;
      if(TMath::Abs(sqrt(vertexX*vertexX+vertexY*vertexY)
      )>=2.) continue;
      hRefMult->Fill(Refmult);
      hRunid->Fill(index);
      qx=0,qy=0,qx_east=0,qy_east=0,qx_west=0,qy_west=0;
for(int it=0;it<ntrack;it++){
track_px    = (double)      Px[it];
        track_py    = (double)      Py[it];
        track_pz    = (double)      Pz[it];
```

```
        track_dca   = (double)     DCA[it];
        track_beta  = (double)    Beta[it];
        nsigma_pi   = (double)   nSigpi[it];
        nsigma_k    = (double)    nSigk[it];
        nsigma_p    = (double)    nSigp[it];
        if(DEdx[it]>=0){
          track_dEdx  =  (double) DEdx[it];
        }
        else if(DEdx[it]<0){
          track_dEdx = (double) DEdx[it] * (-1.0);
        }
        track_charge      = (int)     Charge[it];
        if(!track_charge)    track_charge = -1.0;
        track_nHitsFit    = (int)      nHits[it];
        track_nHitsdEdx   = (int)      ndEdx[it];
        track_ToFflag     = (int)     tofFlg[it];
if(track_nHitsFit<=15) continue;
if(TMath::Abs(track_charge)>1) continue;
if(track_dca>2) continue;

//now fill your hists do your thing here:
aTrack.SetXYZ(track_px,track_py,track_pz);
eta1=aTrack.Eta();
if(TMath::Abs(eta1)>1) continue;
        pt=aTrack.Perp();
if(pt<=0.2 || pt>=2.0) continue;
```

```
        m=aTrack.Mag();
      aTrack2.Set(track_px,track_py);
Phi=aTrack2.Phi();
if(Phi<0.)Phi=Phi+2*pi;
//hPhi->Fill(Phi);
//hEta->Fill(eta1);
Double_t qx_corr,qy_corr,qxs_corr,qys_corr;
if(eta1>=0.)
  {
    if(vertexZ>=0.)
      {
qx_corr=QX_fe->GetBinContent(index+1,centrality+1);
qy_corr=QY_fe->GetBinContent(index+1,centrality+1);
      }
    if(vertexZ<0.)
      {
qx_corr=QX_e->GetBinContent(index+1,centrality+1);
qy_corr=QY_e->GetBinContent(index+1,centrality+1);
      }
    qx=qx+(pt*cos(2*Phi)-qx_corr);
    qy=qy+(pt*sin(2*Phi)-qy_corr);
  }
if(eta1<0.)
  {
    if(vertexZ>=0.)
      {
```

```
qx_corr=QX_w->GetBinContent(index+1,centrality+1);
qy_corr=QY_w->GetBinContent(index+1,centrality+1);
      }
    if(vertexZ<0.)
      {
qx_corr=QX_fw->GetBinContent(index+1,centrality+1);
qy_corr=QY_fw->GetBinContent(index+1,centrality+1);
      }
    qx=qx+(pt*cos(2*Phi)-qx_corr);
    qy=qy+(pt*sin(2*Phi)-qy_corr);
  }
//For subrange eta
if(eta1>0.05)
  {
    if(vertexZ>=0.)
      {
qxs_corr=QXs_fe->GetBinContent(index+1,centrality+1);
qys_corr=QYs_fe->GetBinContent(index+1,centrality+1);
      }
    if(vertexZ<0.)
      {
qxs_corr=QXs_e->GetBinContent(index+1,centrality+1);
qys_corr=QYs_e->GetBinContent(index+1,centrality+1);
      }
    qx_east=qx_east+(pt*cos(2*Phi)-qxs_corr);
    qy_east=qy_east+(pt*sin(2*Phi)-qys_corr);
```

```
  }
if(eta1<-0.05)
  {
    if(vertexZ>=0.)
      {
qxs_corr=QXs_w->GetBinContent(index+1,centrality+1);
qys_corr=QYs_w->GetBinContent(index+1,centrality+1);
      }
    if(vertexZ<0.)
      {
qxs_corr=QXs_fw->GetBinContent(index+1,centrality+1);
qys_corr=QYs_fw->GetBinContent(index+1,centrality+1);
      }
    //qx_west=qx_west+(pt*cos(2*Phi));
    //qy_west=qy_west+(pt*sin(2*Phi));
    qx_west=qx_west+(pt*cos(2*Phi)-qxs_corr);
    qy_west=qy_west+(pt*sin(2*Phi)-qys_corr);
  }
      }//track loop ends
      aTrack3.Set(qx,qy);
      Psi2=0.5*aTrack3.Phi();

      aTrack3.Set(qx_east,qy_east);
      Psi2_east=0.5*aTrack3.Phi();

      aTrack3.Set(qx_west,qy_west);
```

```
      Psi2_west=0.5*aTrack3.Phi();

      if(Psi2<0.)Psi2=Psi2+pi;
      if(Psi2_east<0.)Psi2_east=Psi2_east+pi;
      if(Psi2_west<0.)Psi2_west=Psi2_west+pi;
     //Fill Histograms
      hPsi2->Fill(Psi2);
      hPsi2_east->Fill(Psi2_east);
      hPsi2_west->Fill(Psi2_west);
      hQxQy->Fill(qx,qy);

      //Find Shift correction factors :
      Int_t i;
      for(i=1;i<21;i++)
{
  // For fullrange eta
  Sin_corr->Fill(i,sin(i*2*Psi2));
  Cos_corr->Fill(i,cos(i*2*Psi2));

  // For subrange eta
  Sin_corr_east->Fill(i,sin(i*2*Psi2_east));
  Cos_corr_east->Fill(i,cos(i*2*Psi2_east));

  Sin_corr_west->Fill(i,sin(i*2*Psi2_west));
  Cos_corr_west->Fill(i,cos(i*2*Psi2_west));
}
```

```
      nEvent_total++;
      if(nEvent_total%20000==0)
{
  cout<<" Analyzed Event = "<<nEvent_total<<"
   cent = "<<centrality<<" ntrack = "<<ntrack<<
   "  runid = "<<index<<endl;
  sw2.Stop();
  sw2.Print();
  sw2.Start(1);
}
       }//================event loop ends======================;
   cout<<" writing files......."<<endl;
  f = new TFile(outfile,"recreate");
  f->cd();
  hPsi2->Write();
  hPsi2_west->Write();
  hPsi2_east->Write();
  Sin_corr->Write();
  Cos_corr->Write();
  Sin_corr_west->Write();
  Cos_corr_west->Write();
  Sin_corr_east->Write();
  Cos_corr_east->Write();
  hRefMult->Write();
  hQxQy->Write();
  f->Close();
```

```
  cout<<"\n File written.\n ******** Events = "
  <<nEvent_total<<"********\n"<<endl;
  sw2.Stop();
  sw1.Stop();
  sw1.Print();
}//main code ends
```

A.3.3 Elliptic flow

```
#define star_uu_193_new_cxx
#include "star_uu_193_new.h"
#include <TH2.h>
#include <TStyle.h>
#include <TCanvas.h>
#include<TVector.h>
#include<TVector2.h>
#include "run.h"
Int_t        runID,Refmult,BBC_cons,ZDC_cons,
centrality,ntrack;
Double_t     vertexX,vertexY,vertexZ,VPD_vertexZ;
Int_t       track_charge,track_nHitsFit,track_
nHitsdEdx,track_ToFflag;
Double_t nsigma_pi,nsigma_k,nsigma_p,nsigma_e;
Double_t track_px,track_py,track_pz,track_dca,
track_dEdx,track_beta;
Long64_t nEvent_total = 0;
Double_t m,m2,p2,x2,y2,qx,qy,qx_east,qy_east,qx_west,
```

```
qy_west,Psi2,Psi2_east,Psi2_west,Phi,fitratio,pt,eta1,eta2;
Float_t pi=3.1416;
Int_t n_flag=783,index;
TVector3 aTrack;
TVector2 aTrack2,aTrack3;
//Pion and kaon momentum matrices
Double_t   px_pipl[5000],py_pipl[5000],pz_pipl[5000]
,qx_pipl[5000],qy_pipl[5000];
Double_t   px_pimn[5000],py_pimn[5000],pz_pimn[5000]
,qx_pimn[5000],qy_pimn[5000];
Int_t      flowid_pimn[5000], flowid_pipl[5000];
Double_t   px_kpl[5000],py_kpl[5000],pz_kpl[5000],
qx_kpl[5000],qy_kpl[5000];
Double_t   px_kmn[5000],py_kmn[5000],pz_kmn[5000],
qx_kmn[5000],qy_kmn[5000];
Int_t      flowid_kmn[5000], flowid_kpl[5000];
//Resolution factors
      Double_t Resfactors[9] =
{0.0478008,
 0.109947,
 0.22172,
 0.356743,
 0.464873,
 0.511486,
 0.470687,
 0.362439,
```

```
 0.273237};
void star_uu_193_new::Loop()
{
  // Histogram must be declared in header file:
  char input_rootfile1[250];
  char input_rootfile2[250];
  char name1[250];
  char name2[250];
  char name3[250];
  char name4[250];
  char name5[250];
  char title[250];
  //Input Files
  sprintf(input_rootfile1,"/Users/ranbirsingh/arabinda/
  flow/output/event_planes/recenter/file_all.root");
  sprintf(input_rootfile2,"/Users/ranbirsingh/arabinda/
  flow/output/event_planes/shift/file_all.root");
  cout<<"Input file joint for Recenter = "
  <<input_rootfile1<<endl;
  cout<<"Input file joint for Shift = "
  <<input_rootfile2<<endl;
  TFile *f=new TFile(input_rootfile1,"READ");
  TProfile2D *QX_fw=(TProfile2D*)f->Get("QX_fw");
  TProfile2D *QX_w=(TProfile2D*)f->Get("QX_w");
  TProfile2D *QX_fe=(TProfile2D*)f->Get("QX_fe");
  TProfile2D *QX_e=(TProfile2D*)f->Get("QX_e");
```

```
TProfile2D *QY_fw=(TProfile2D*)f->Get("QY_fw");
TProfile2D *QY_w=(TProfile2D*)f->Get("QY_w");
TProfile2D *QY_fe=(TProfile2D*)f->Get("QY_fe");
TProfile2D *QY_e=(TProfile2D*)f->Get("QY_e");
TProfile2D *QXs_fw=(TProfile2D*)f->Get("QXs_fw");
TProfile2D *QXs_w=(TProfile2D*)f->Get("QXs_w");
TProfile2D *QXs_fe=(TProfile2D*)f->Get("QXs_fe");
TProfile2D *QXs_e=(TProfile2D*)f->Get("QXs_e");
TProfile2D *QYs_fw=(TProfile2D*)f->Get("QYs_fw");
TProfile2D *QYs_w=(TProfile2D*)f->Get("QYs_w");
TProfile2D *QYs_fe=(TProfile2D*)f->Get("QYs_fe");
TProfile2D *QYs_e=(TProfile2D*)f->Get("QYs_e");
TFile *f2=new TFile(input_rootfile2,"READ");
TProfile *Sin_corr=(TProfile*)f2->Get("Sin_corr");
TProfile *Cos_corr=(TProfile*)f2->Get("Cos_corr");
TProfile *Sin_corr_west=(TProfile*)f2->Get("Sin_corr_west");
TProfile *Cos_corr_west=(TProfile*)f2->Get("Cos_corr_west");
TProfile *Sin_corr_east=(TProfile*)f2->Get("Sin_corr_east");
TProfile *Cos_corr_east=(TProfile*)f2->Get("Cos_corr_east");
TH1F *hRefMult=new TH1F("hRefMult","Reference Multiplicity"
,1000,0.,1000);
TH1F *hPt=new TH1F("hPt","Pt distribution",100,0.0,10.0);
TH1F *hPt_kstar=new TH1F("hPt_kstar","K* Pt distribution",
100,0.0,10.0);
TH1D *hPsi2=new TH1D("hPsi2","Event Plane Angle #Psi_{2}"
,200,-1.,5.);
```

```
  TH1D *hPsi2_east=new TH1D("hPsi2_east","Event Plane Angle
   #Psi_{2} for #eta subrange(0.05 to 1.0)",200,-1.,5.);
  TH1D *hPsi2_west=new TH1D("hPsi2_west","Event Plane Angle
   #Psi_{2}for #eta subrange(-1.0 to-0.05 )",200,-1.0,5.);
  TH1D *hPsi2_corr=new TH1D("hPsi2_corr","Event Plane Angle
   #Psi_{2}",200,-1.,5.);
  TH1D *hPsi2_east_corr=new TH1D("hPsi2_east_corr","Event
   Plane Angle #Psi_{2} for #eta subrange(0.05 to 1.0)",200,-1.,5.);
  TH1D *hPsi2_west_corr=new TH1D("hPsi2_west_corr","Event
  Plane Angle #Psi_{2}for #eta subrange(-1.0 to-0.05 )",200,-1.0,5.);

  TH2D *hQxQy_rcorr=new TH2D("hQxQy_rcorr","Histogram of
   Qx vs Qy after recenter correction",500,-100.,100.,500,-100.,100.);
  TH2D *hQxQy_scorr=new TH2D("hQxQy_scorr","Histogram of
  Qx vs Qy after Shift correction",500,-100.,100.,500,-100.,100.);
TProfile *Res_factor=new TProfile("Res_factor","Profile
Histogram of Resolution factor vs Centrality",10,0,10,-2.,2.);
  TH2D *hLike[9];
  TH2D *hLike_spectra[9];
  TH2D *hUnlike[9];
  TH2D *hUnlike_spectra[9];
  TH3D *hUnlike_3D[9];
  for(Int_t cent_id=0;cent_id<9;cent_id++)
    {
      sprintf(name1,"hLike_%d",cent_id);
      sprintf(name2,"hUnlike_%d",cent_id);
```

```
    sprintf(name3,"hUnlike_3D_%d",cent_id);
    sprintf(name4,"hLike_spectra_%d",cent_id);
    sprintf(name5,"hUnlike_spectra_%d",cent_id);
    hLike[cent_id]=new TH2D(name1,name1,100,0.,10.,
    900,0.6,1.5);
    hLike_spectra[cent_id]=new TH2D(name4,name4,100
    ,0.,10.,900,0.6,1.5);
    hUnlike[cent_id]=new TH2D(name2,name2,100,0.,10.
    ,900,0.6,1.5);
    hUnlike_spectra[cent_id]=new TH2D(name5,name5,100,
    0.,10.,900,0.6,1.5);
    hUnlike_3D[cent_id]=new TH3D(name3,name3,100,0.,10.,
    900,0.6,1.5,200,-5.,5.);
  }
if (fChain == 0) return;
Long64_t nentries = fChain->GetEntriesFast();
Long64_t nbytes = 0, nb = 0;
TStopwatch sw1,sw2;
sw1.Start(1);
sw2.Start(1);
  for (Long64_t jentry=0; jentry<nentries;jentry++)
  {//event loop starts
    Long64_t ientry = LoadTree(jentry);
    if (ientry < 0) break;
    nb = fChain->GetEntry(jentry);   nbytes += nb;
   runID       =                  RunID;
```

```
      Refmult     = (int)            Refm;
      BBC         = (int)             BBC;
      ZDC         = (int)             ZDC;
      centrality  = (int)            Cent;
      ntrack      = (int)          nTrack;
      vertexX     = (double)           Vx;
      vertexY     = (double)           Vy;
      vertexZ     = (double)           Vz;
      VPD_vertexZ = (double)        VpdVz;
      for(Int_t j=0;j<n_flag;j++)
        {
          if(numbers[j]==runID)
            index=j;
        }
      if(TMath::Abs(vertexZ)>=30.) continue;
      if(TMath::Abs(sqrt(vertexX*vertexX+vertexY*vertexY))>=2.)
       continue;
      hRefMult->Fill(Refmult);
       qx=0,qy=0,qx_east=0,qy_east=0,qx_west=0,qy_west=0;
      Double_t s,c,dPsi2=0,dPsi2_east=0,dPsi2_west=0;
      Int_t i;
      for(int it=0;it<ntrack;it++){
track_px    = (double)      Px[it];
        track_py    = (double)      Py[it];
        track_pz    = (double)      Pz[it];
        track_dca   = (double)     DCA[it];
```

```
        track_beta  = (double)    Beta[it];
        nsigma_pi   = (double)   nSigpi[it];
        nsigma_k    = (double)    nSigk[it];
        nsigma_p    = (double)    nSigp[it];
        if(DEdx[it]>=0){
          track_dEdx  =  (double) DEdx[it];
        }
        else if(DEdx[it]<0){
          track_dEdx = (double) DEdx[it] * (-1.0);
        }
        track_charge      = (int)     Charge[it];
        if(!track_charge)    track_charge = -1.0;
        track_nHitsFit    = (int)      nHits[it];
        track_nHitsdEdx   = (int)      ndEdx[it];
        track_ToFflag     = (int)     tofFlg[it];
if(track_nHitsFit<=15) continue;
if(TMath::Abs(track_charge)>1) continue;
if(track_dca>2) continue;
aTrack.SetXYZ(track_px,track_py,track_pz);
eta1=aTrack.Eta();
if(TMath::Abs(eta1)>1) continue;
        pt=aTrack.Perp();
if(pt<=0.2 || pt>=2.0) continue;
        m=aTrack.Mag();
   aTrack2.Set(track_px,track_py);
Phi=aTrack2.Phi();
```

```
if(Phi<0.)Phi=Phi+2*pi;
Double_t qx_corr,qy_corr,qxs_corr,qys_corr;
if(eta1>=0.)
  {
    if(vertexZ>=0.)
      {
qx_corr=QX_fe->GetBinContent(index+1,centrality+1);
qy_corr=QY_fe->GetBinContent(index+1,centrality+1);
      }
    if(vertexZ<0.)
      {
qx_corr=QX_e->GetBinContent(index+1,centrality+1);
qy_corr=QY_e->GetBinContent(index+1,centrality+1);
      }
    qx=qx+(pt*cos(2*Phi)-qx_corr);
    qy=qy+(pt*sin(2*Phi)-qy_corr);
  }
if(eta1<0.)
  {
    if(vertexZ>=0.)
      {
qx_corr=QX_w->GetBinContent(index+1,centrality+1);
qy_corr=QY_w->GetBinContent(index+1,centrality+1);
      }
    if(vertexZ<0.)
      {
```

```
qx_corr=QX_fw->GetBinContent(index+1,centrality+1);
qy_corr=QY_fw->GetBinContent(index+1,centrality+1);
      }
    qx=qx+(pt*cos(2*Phi)-qx_corr);
    qy=qy+(pt*sin(2*Phi)-qy_corr);
  }
if(eta1>0.05)
  {
    if(vertexZ>=0.)
      {
qxs_corr=QXs_fe->GetBinContent(index+1,centrality+1);
qys_corr=QYs_fe->GetBinContent(index+1,centrality+1);
      }
    if(vertexZ<0.)
      {
qxs_corr=QXs_e->GetBinContent(index+1,centrality+1);
qys_corr=QYs_e->GetBinContent(index+1,centrality+1);
      }
    qx_east=qx_east+(pt*cos(2*Phi)-qxs_corr);
    qy_east=qy_east+(pt*sin(2*Phi)-qys_corr);
  }
    if(eta1<-0.05)
  {
    if(vertexZ>=0.)
      {
qxs_corr=QXs_w->GetBinContent(index+1,centrality+1);
```

```
qys_corr=QYs_w->GetBinContent(index+1,centrality+1);
      }
    if(vertexZ<0.)
      {
qxs_corr=QXs_fw->GetBinContent(index+1,centrality+1);
qys_corr=QYs_fw->GetBinContent(index+1,centrality+1);
      }
    qx_west=qx_west+(pt*cos(2*Phi)-qxs_corr);
    qy_west=qy_west+(pt*sin(2*Phi)-qys_corr);
  }
      }//track loop ends
       aTrack3.Set(qx,qy);
      Psi2=0.5*aTrack3.Phi();
      aTrack3.Set(qx_east,qy_east);
      Psi2_east=0.5*aTrack3.Phi();
      aTrack3.Set(qx_west,qy_west);
      Psi2_west=0.5*aTrack3.Phi();
      if(Psi2<0.) Psi2=Psi2+pi;
      if(Psi2_east<0.) Psi2_east=Psi2_east+pi;
      if(Psi2_west<0.) Psi2_west=Psi2_west+pi;
      hPsi2->Fill(Psi2);
      hPsi2_east->Fill(Psi2_east);
      hPsi2_west->Fill(Psi2_west);
      Double_t Psi2_corr,Psi2_east_corr,Psi2_west_corr;
      Psi2_corr=Psi2+dPsi2;
      Psi2_west_corr=Psi2_west+dPsi2_west;
```

```
    Psi2_east_corr=Psi2_east+dPsi2_east;


    if(Psi2_corr<0.) Psi2_corr=Psi2_corr+pi;
    if(Psi2_east_corr<0.) Psi2_east_corr=Psi2_east_corr+pi;
    if(Psi2_west_corr<0.) Psi2_west_corr=Psi2_west_corr+pi;
    hPsi2_corr->Fill(Psi2_corr);
    hPsi2_east_corr->Fill(Psi2_east_corr);
    hPsi2_west_corr->Fill(Psi2_west_corr);
    Double_t Q_vec,qx_scorr,qy_scorr,qx_east_scorr,
    qy_east_scorr,qx_west_scorr, qy_west_scorr;
    TVector2 Q2_vec;
    Q2_vec.Set(qx,qy);
    Q_vec=Q2_vec.Mod();
    qx_scorr=Q_vec*cos(2*Psi2_corr);
    qy_scorr=Q_vec*sin(2*Psi2_corr);
          Q2_vec.Set(qx_east,qy_east);
    Q_vec=Q2_vec.Mod();
    qx_east_scorr=Q_vec*cos(2*Psi2_east_corr);
    qy_east_scorr=Q_vec*sin(2*Psi2_east_corr);
    Q2_vec.Set(qx_west,qy_west);
    Q_vec=Q2_vec.Mod();
    qx_west_scorr=Q_vec*cos(2*Psi2_west_corr);
    qy_west_scorr=Q_vec*sin(2*Psi2_west_corr);
    hQxQy_rcorr->Fill(qx,qy);
    hQxQy_scorr->Fill(qx_scorr,qy_scorr);
Double_t Res;
```

```
      Res = cos(2*(Psi2_east_corr-Psi2_west_corr));
      Res_factor->Fill(centrality,Res);
       Double_t p,m2,qx2,qy2,m_k=0.493, m_pi=0.139;
      i=0;
      Int_t g=0,h=0,j=0;
      //V2 track loop starts
      for(int it=0;it<ntrack;it++)
{
track_px    = (double)      Px[it];
        track_py    = (double)      Py[it];
        track_pz    = (double)      Pz[it];
        track_dca   = (double)     DCA[it];
        track_beta  = (double)    Beta[it];
        nsigma_pi   = (double)   nSigpi[it];
        nsigma_k    = (double)    nSigk[it];
        nsigma_p    = (double)    nSigp[it];
        if(DEdx[it]>=0){
          track_dEdx  =  (double) DEdx[it];
        }
        else if(DEdx[it]<0){
          track_dEdx = (double) DEdx[it] * (-1.0);
        }
        track_charge      = (int)     Charge[it];
        if(!track_charge)    track_charge = -1.0;
        track_nHitsFit    = (int)      nHits[it];
        track_nHitsdEdx   = (int)      ndEdx[it];
```

```
        track_ToFflag     = (int)     tofFlg[it];
if(track_nHitsFit<=15) continue;
if(TMath::Abs(track_charge)>1.) continue;
        if(track_dca>3) continue;
aTrack.SetXYZ(track_px,track_py,track_pz);
        eta1=aTrack.Eta();
if(TMath::Abs(eta1)>1) continue;
        pt=aTrack.Perp();
if(pt<=0.15 || pt>=10.0) continue;
p=aTrack.Mag();
p2=p*p;
m2=p2*((1/pow(track_beta,2))-1);
hPt->Fill(pt);
Int_t flow_id=0;
if(pt>0.2 && pt<2.  && track_dca<=2. && track_nHitsFit>15
 && TMath::Abs(eta1)<=1.) flow_id=1;
aTrack2.Set(track_px,track_py);
Phi=aTrack2.Phi();
if(Phi<0.) Phi=Phi+2*pi;
qx2=pt*cos(2*Phi);
qy2=pt*sin(2*Phi);
if(TMath::Abs(nsigma_pi)<2.0)
          {
    if(m2>=0.01038 && m2<=0.029)
      {
      if(track_charge==1)
```

```
{// Pion plus selection starts
  //EK_pipl[g] = pow((p2+0.019321),0.5);
  px_pipl[g]=track_px;
  py_pipl[g]=track_py;
  pz_pipl[g]=track_pz;
  qx_pipl[g]=qx2;
  qy_pipl[g]=qy2;
  flowid_pipl[g]=flow_id;
  g++;
}// Pion plus selection ends


      if(track_charge==-1)
{// Pion minus selection starts
  //EK_pimn[h] = pow((p2+0.019321),0.5);
  px_pimn[h]=track_px;
  py_pimn[h]=track_py;
  pz_pimn[h]=track_pz;
  qx_pimn[h]=qx2;
  qy_pimn[h]=qy2;
  flowid_pimn[h]=flow_id;
  //charge_pimn[h]=track_charge;
  //hNsigmapimn->Fill(nsigma_pi);
  //hcharge_pimn->Fill(track_charge);
  h++;
}// Pion minus selection ends
    }
```

```
  }//Pion selection ends


//Kaon selection
        if(TMath::Abs(nsigma_k)<2.0)
          {
            if(m2>=0.2104 && m2<=0.2796)
              {
                if(track_charge==1)
                  {// Kaon plus selection starts
    //EK_kpl[i] = pow((p2+0.243049),0.5);
                    px_kpl[i]=track_px;
                    py_kpl[i]=track_py;
                    pz_kpl[i]=track_pz;
                    qx_kpl[i]=qx2;
    qy_kpl[i]=qy2;
    flowid_kpl[i]=flow_id;
    //charge_kpl[i]=track_charge;
                     i++;
                  }// Kaon plus selection ends


                if(track_charge==-1)
                  {// Kaon minus selection starts
                    px_kmn[j]=track_px;
                    py_kmn[j]=track_py;
                    pz_kmn[j]=track_pz;
    qx_kmn[j]=qx2;
```

```
   qy_kmn[j]=qy2;
   flowid_kmn[j]=flow_id;
   //charge_kmn[j]=track_charge;
   j++;
                }// Kaon minus selection ends
           }
        }//Kaon selection ends


}      TLorentzVector Pion,Kaon,K_star;
     //Same event calculation for K+ pi+
     for(Int_t m=0; m<g; m++)
       {
 Pion.SetXYZM(px_pipl[m],py_pipl[m],pz_pipl[m],m_pi);
         for(Int_t l=0; l<i; l++)
           {
             //if(charge_pipl[m]+charge_kpl[l]==0) continue;
     Kaon.SetXYZM(px_kpl[l],py_kpl[l],pz_kpl[l],m_k);
     K_star=Kaon+Pion;

     Double_t  KStarInvM_pp = K_star.M();
     pt=K_star.Pt();
     hPt_kstar->Fill(pt);
             hLike[centrality]->Fill(pt,KStarInvM_pp);
     eta1=K_star.PseudoRapidity();
     if(TMath::Abs(eta1)<=0.5)
```

```
{
  hLike_spectra[centrality]->Fill(pt,KStarInvM_pp);
}
            }//l loop
}//m loop


      //Same event calculation for K- pi-
      for(Int_t m=0; m<h; m++)
        {
  Pion.SetXYZM(px_pimn[m],py_pimn[m],pz_pimn[m],m_pi);
          for(Int_t l=0; l<j; l++)
            {
              //if(charge_pimn[m]+charge_kmn[l]==0) continue;
      Kaon.SetXYZM(px_kmn[l],py_kmn[l],pz_kmn[l],m_k);
      K_star=Kaon+Pion;

      Double_t  KStarInvM_mm = K_star.M();
      pt=K_star.Pt();
      hPt_kstar->Fill(pt);
              hLike[centrality]->Fill(pt,KStarInvM_mm);
              eta1=K_star.PseudoRapidity();
      if(TMath::Abs(eta1)<=0.5)
{
  hLike_spectra[centrality]->Fill(pt,KStarInvM_mm);
}
```

```
    }//l loop
}//m loop
      TVector2 dQ;
      TVector2 Q,Q_i,Q_east,Q_east_i,Q_west,Q_west_i,;


      for(Int_t m=0; m<h; m++)
        {
          Pion.SetXYZM(px_pimn[m],py_pimn[m],pz_pimn[m],m_pi);
  Double_t pieta = Pion.PseudoRapidity();
  for(Int_t l=0; l<i; l++)
            {
      Kaon.SetXYZM(px_kpl[l],py_kpl[l],pz_kpl[l],m_k);
      Double_t keta = Kaon.PseudoRapidity();
      K_star=Kaon+Pion;
      Double_t Rapidity=K_star.Rapidity();
      if(TMath::Abs(Rapidity)>1.) continue;
      Q.Set(qx_scorr,qy_scorr);
      Q_i=Q;
      Q_east.Set(qx_east_scorr,qy_east_scorr);
      Q_west.Set(qx_west_scorr,qy_west_scorr);
      Q_east_i=Q_east;
      Q_west_i=Q_west;
      if(flowid_kpl[l]==1)
{
  //TVector2 dQ;
  dQ.Set(qx_kpl[l],qy_kpl[l]);
```

```
  Q_i=Q_i-dQ;
  if(keta < -0.05) Q_west_i=Q_west_i-dQ;
  if(keta > 0.05) Q_east_i=Q_east_i-dQ;
}
      if(flowid_pimn[m]==1)
{
  //TVector2 dQ;
  dQ.Set(qx_pimn[m],qy_pimn[m]);
  Q_i=Q_i-dQ;
  if(pieta < -0.05) Q_west_i=Q_west_i-dQ;
  if(pieta > 0.05) Q_east_i=Q_east_i-dQ;
}
      Double_t Psi;
      Double_t K_star_eta=K_star.PseudoRapidity();
      if(K_star_eta<0.) Psi=0.5*Q_east_i.Phi();
      if(K_star_eta>=0.) Psi=0.5*Q_west_i.Phi();
      Phi=K_star.Phi();
      if(Phi<0.) Phi=Phi+2*pi;
      Double_t  KStarInvM_pm = K_star.M();
      pt=K_star.Pt();
      hPt_kstar->Fill(pt);
              hUnlike[centrality]->Fill(pt,KStarInvM_pm);
      Double_t v2_factor=cos(2*(Phi-Psi))/(sqrt(Resfactors
      [centrality]));
              hUnlike_3D[centrality]->Fill(pt,KStarInvM_pm,v2_factor);
          if(TMath::Abs(K_star_eta)<=0.5)
```

```
{
  hUnlike_spectra[centrality]->Fill(pt,KStarInvM_pm);
}
      }//l loop
        }
        for(Int_t m=0; m<g; m++)
        {
          Pion.SetXYZM(px_pipl[m],py_pipl[m],pz_pipl[m],m_pi);
  Double_t pieta = Pion.PseudoRapidity();
          for(Int_t l=0; l<j; l++)
            {
      Kaon.SetXYZM(px_kmn[l],py_kmn[l],pz_kmn[l],m_k);
      Double_t keta = Kaon.PseudoRapidity();
      K_star=Kaon+Pion;
      Double_t Rapidity=K_star.Rapidity();
      if(TMath::Abs(Rapidity)>1.) continue;

      //TVector2 Q,Q_i,Q_east,Q_east_i,Q_west,Q_west_i,;
      Q.Set(qx_scorr,qy_scorr);
      Q_i=Q;
      Q_east.Set(qx_east_scorr,qy_east_scorr);
      Q_west.Set(qx_west_scorr,qy_west_scorr);
      Q_east_i=Q_east;
      Q_west_i=Q_west;
      if(flowid_kmn[l]==1)
{
```

```
  //TVector2 dQ;
  dQ.Set(qx_kmn[l],qy_kmn[l]);
  Q_i=Q_i-dQ;
  if(keta < -0.05) Q_west_i=Q_west_i-dQ;
  if(keta > 0.05) Q_east_i=Q_east_i-dQ;
}
      if(flowid_pipl[m]==1)
{
  dQ.Set(qx_pipl[m],qy_pipl[m]);
  Q_i=Q_i-dQ;
  if(pieta < -0.05) Q_west_i=Q_west_i-dQ;
  if(pieta > 0.05) Q_east_i=Q_east_i-dQ;
}
      Double_t Psi;
      Double_t K_star_eta=K_star.PseudoRapidity();
      if(K_star_eta<0.) Psi=0.5*Q_east_i.Phi();
      if(K_star_eta>=0.) Psi=0.5*Q_west_i.Phi();
      Phi=K_star.Phi();
      if(Phi<0.) Phi=Phi+2*pi;
      Double_t  KStarInvM_mp = K_star.M();
      pt=K_star.Pt();
      hPt_kstar->Fill(pt);
              hUnlike[centrality]->Fill(pt,KStarInvM_mp);
      Double_t v2_factor=cos(2*(Phi-Psi))/(sqrt(Resfactors
      [centrality]));
              hUnlike_3D[centrality]->Fill(pt,KStarInvM_mp,
```

```
              v2_factor);
      if(TMath::Abs(K_star_eta)<=0.5)
{
  hUnlike_spectra[centrality]->Fill(pt,KStarInvM_mp);
}
    }//l loop
        }//m loop same event loop ends K- Pi+
      nEvent_total++;
     if(nEvent_total%3000==0)
       {
 cout<<" Analyzed Event = "<<nEvent_total<<" cent =
 "<<centrality<<" ntrack = "<<ntrack<<"  runid =
 "<<index<<endl;
 sw2.Stop();
 sw2.Print();
 sw2.Start(1);
      }
   }//================event loop ends======================;
  cout<<" writing files......."<<endl;
  f = new TFile(outfile,"recreate");
  f->cd();
  hPsi2->Write();
  hPsi2_west->Write();
  hPsi2_east->Write();
  hPsi2_corr->Write();
  hPsi2_west_corr->Write();
```

```
 hPsi2_east_corr->Write();
 Res_factor->Write();
 hQxQy_rcorr->Write();
 hQxQy_scorr->Write();
 hRefMult->Write();
 hPt->Write();
 hPt_kstar->Write();
   for(Int_t i=0;i<9;i++)
   {
     hLike[i]->Write();
     hLike_spectra[i]->Write();
     hUnlike[i]->Write();
     hUnlike_spectra[i]->Write();
     hUnlike_3D[i]->Write();
   }
   f->Close();
 cout<<"\n File written.\n ******** Events = "
 <<nEvent_total<<"********\n"<<endl;
 sw2.Stop();
 sw1.Stop();
 sw1.Print();
}//main code ends
```