Centrality dependent $\frac{dN}{d\eta}$ at $\eta = 0$

Bjørn H. Samset, Truls M. Larsen and Trine S. Tveter

February 16, 2001

Contents

Run number	Number of events
2529	83k
2530	412k
2531	124k
2535	172k
Total	791k

Table 1: The number of events used in this analysis

1 Introduction

This is a brief description of the second iteration of a charged particle production analysis, presenting a $< dN/d\eta >$ per participant pair consistent with those presented by PHOBOS and PHENIX.

General motivation: BRAHMS has two detectors capable of measuring charged particle multiplicities at midrapidity — the mult barrel and TPM1. Extracting $dN/d\eta$ from TPM1 is not straightforward, as tracking efficiencies, geometric acceptance and background from secondary particles e.g. produced in the beampipe must all be corrected for. Centrality is detrermined from the particle multiplicity measured by the tile barrel, and $< N_{part} >$ for the different centrality classes is obtained from a Glauber model calculation based on numbers recently published by D. Kharzeev and M. Nardi. ([1])

The results presented here are from runs 2529, 2530, 2531 and 2535, and are now corrected for efficiency by taking into account both our tracking efficiency *as a function of the number of clusters* and the fact that the failure probability of the TrackFollowFinder is also dependent on the number of clusters. See below for details.

Also, the analysis is done both for the older GetCentrality–method developed by J. H. Lee and the new method described by C. E. Jørgensen and C. H. Christensen in [2]. I'll first go through the entire analysis using JH's cuts, and then give the results from the other one at the end and compare (the reason for this is simply that the first analysis was already finished when the new centrality algorithms were published).

The raw number of events analyzed is shown in table 1. After requiring a vertex cut (see below) and that the BB and ZDC vertex positions agree to within 15cm, this sample shrunk to 216k events.

2 Vertex cut

The numbers given below are calculated with a vertex fiducial area of $\pm 15cm$, calculated by BB timing. It has been brought to my attention later that this wide a cut may cause the centrality classes (see below) to become invalid.

Class number	% of min-bias	$< N_{part} >$
1	0-6	339
2	6-10	282
3	10-20	220
4	20-30	154
5	30-40	102
6	40-50	63

Table 2: The centrality classes as selected by JH's GetCentrality

I have therefore also done the analysis with a vertex cut of $\pm 5cm$ to see if this had an effect. While drastically reducing my statistics, the effect on the average number of tracks in the TPC in each centrality class was less than 1%. Also, the peak to background for the track y–projections did not change.

3 Centrality classes, JH style

In the following, centrality is first determined via a macro written by J.H.Lee. It takes the min-bias spectrum to be trigger 4 with the additional requirement of at least one hit in the tile array, and selects the centrality class for each event based on the corrected tile multiplicity.

The definition of the classes and my naming convention is shown in table 2. The number of participating nucleons, $< N_{part} >$, is calculated by a rough extrapolation of numbers given in [1] - see appendix B.

4 Particles per event and efficiency correction

This section contains what is essentially new in the analysis, since efficiency corrections and problems with TrackFollowFinder are now dealt with as a function of the number of clusters in TPM1.

4.1 TFF crash probability

The first thing to note is that TrackFollowFinder (TFF) has a tendency to "give up" in a few percent of the events. The reason is that the hit search algorithm used by TFF may cause too many track candidates to be produced at high occupancies. As figure 1 shows, TFF tends to crash more frequently with high occupancy in TPM1. Since we can assume that there is some approximately linear relationship between $N_{clusters}$ and N_{tracks}, this means that we lose high–multiplicity events and thus reduce the overall $dN/d\eta$.

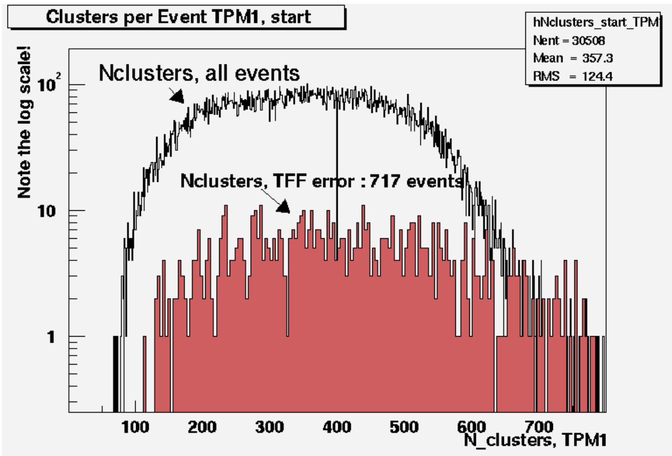

Figure 1: Distribution of events where TrackFollowFinder crashes

To test this further, we have in figure 2 plotted the relationship between $N_{clusters}$ and N_{tracks} from events reconstructed with TFF. There is a clear drop-off towards high $N_{clusters}$. Some of this will be due to tracking efficiency, but the effect is larger than what we see in our efficiency tests (see below). We therefore made a simple analysis where we used TrackStringFinder (TSF) instead of TFF on the events where TFF crashed. We also ran TSF on a selection of low–multiplicity events to determine the relationship between the raw number of tracks found by the two[1], and then scaled the number of TSF tracks by this factor and added the TSF results to the analysis where TFF crashed. The result, shown in figure 3, is much closer to the expected behavior. We still see a small drop-off at high $N_{clusters}$, but it is now of the order of our measured tracking efficiency.

We have therefore in table 3 measured the TFF crash probability as a function of the number of clusters from the entire sample of events. Note the division into $N_{clusters}$–bins of 50.

4.2 Efficiency as function of $N_{clusters}$

The tracking efficiency of TFF has been measured by Trine Tveter and Truls M. Larsen using a track insertion method, adding digitized geant tracks to real events in TPM1 at the sequence level[2]. Trine has also extracted new digitization parameters for the geant tracks to get time– and pad–profiles more closely resembling our experimental data. For a thorough presentation of this, see appendix A.

[1]It has been established earlier that TSF tends to find more tracks than TFF, though not necessarily more primary tracks.

[2]We used the module *BrTPCSequenceAdder*, which will very soon be checked into CVS

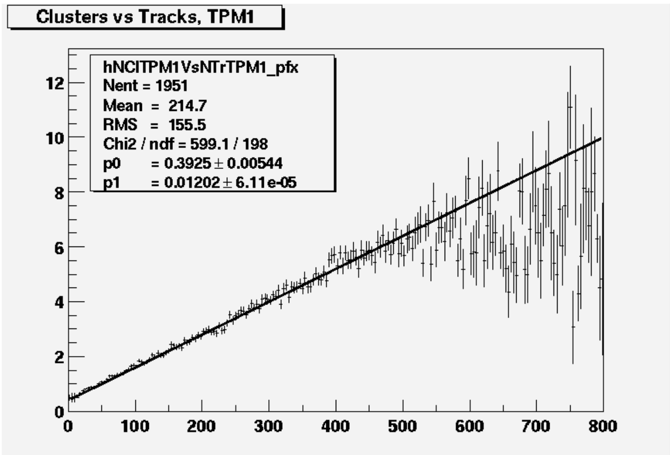

Figure 2: N_{tracks} vs $N_{clusters}$ in events where TrackFollowFinder produces a result

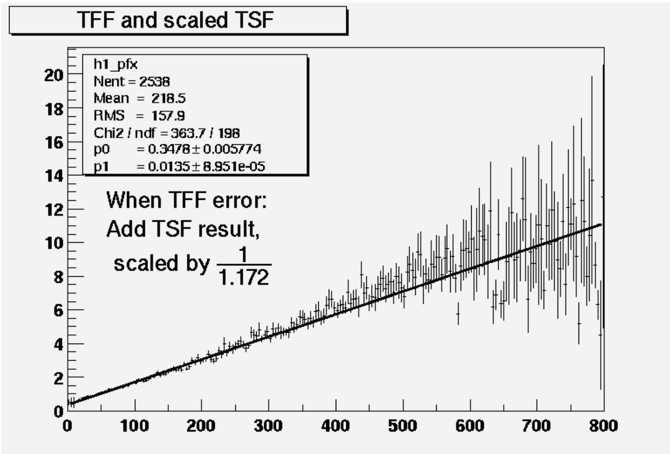

Figure 3: N_{tracks} vs $N_{clusters}$ including scaled results from TrackStringFinder where TrackFollowFinder crashes

$N_{clusters}$	0-50	50-100	100-150	150-200
$P_{crash}(N_{clusters})$	0	0	0	0.01
$N_{clusters}$	200-250	250-300	300-350	350-400
$P_{crash}(N_{clusters})$	0.02	0.02	0.03	0.04
$N_{clusters}$	400-450	450-500	500-550	550-600
$P_{crash}(N_{clusters})$	0.05	0.07	0.09	0.10
$N_{clusters}$	600-650	650-700	700-750	750-800
$P_{crash}(N_{clusters})$	0.12	0.15	0.15	0.15
$N_{clusters}$	800-850	850-900	900-950	950-1000
$P_{crash}(N_{clusters})$	0.16	0.25	0.23	0.22

Table 3: TFF crash probability as function of $N_{clusters}$

Also — why correct for efficiency based on number of clusters and not number of tracks? We attempt to illustrate this in figure 4, which shows our measured efficiency for different centrality classes as a function of tracks (left) and clusters (right). As you can see, the behavior as a function of N_{tracks} seems to be both non–linear and changing between centrality bins, whereas the behavior as a function of $N_{clusters}$ is monotonically decreasing in more or less the same way for all bins.

The latest, still preliminary result of the efficiency analysis is shown in figure 5, where the profile has been fitted with a first order polynomial. The functional dependence we get is

$$\epsilon_{TFF}(N_{clusters}) = 0.9974 - 2.2 \cdot 10^{-4} \cdot N_{clusters} \tag{1}$$

This parametrisation is used to calculate $dN/d\eta$ below.

4.3 Calculating $< N_{tracks}/N_{events} >$

We can now calculate the efficiency corrected number of tracks on a sample of events, and then divide by the number of events to get $dN/d\eta$ corrected for everything but geometry and background. To avoid any edge–effects in the TPC, we restrict the accepted tracks to those passing a fiducial area of $x = \pm 10cm$ and $y = \pm 7cm$ in the TPC. This does not change the efficiency correction procedure, only the geometry factor.

The formula is

$$N_{tracks}^{total} = \sum_{N_{tracks}} N_{tracks} \sum_{N_{clusters}} \frac{1}{\epsilon_{TFF}(N_{clusters})} N_{events}^{true}(N_{tracks}, N_{clusters}) \tag{2}$$

where

$$N_{events}^{true}(N_{tracks}, N_{clusters}) = \frac{N_{events}^{observed}(N_{tracks}, N_{clusters})}{1 - P_{crash}^{TFF}(N_{clusters})} \tag{3}$$

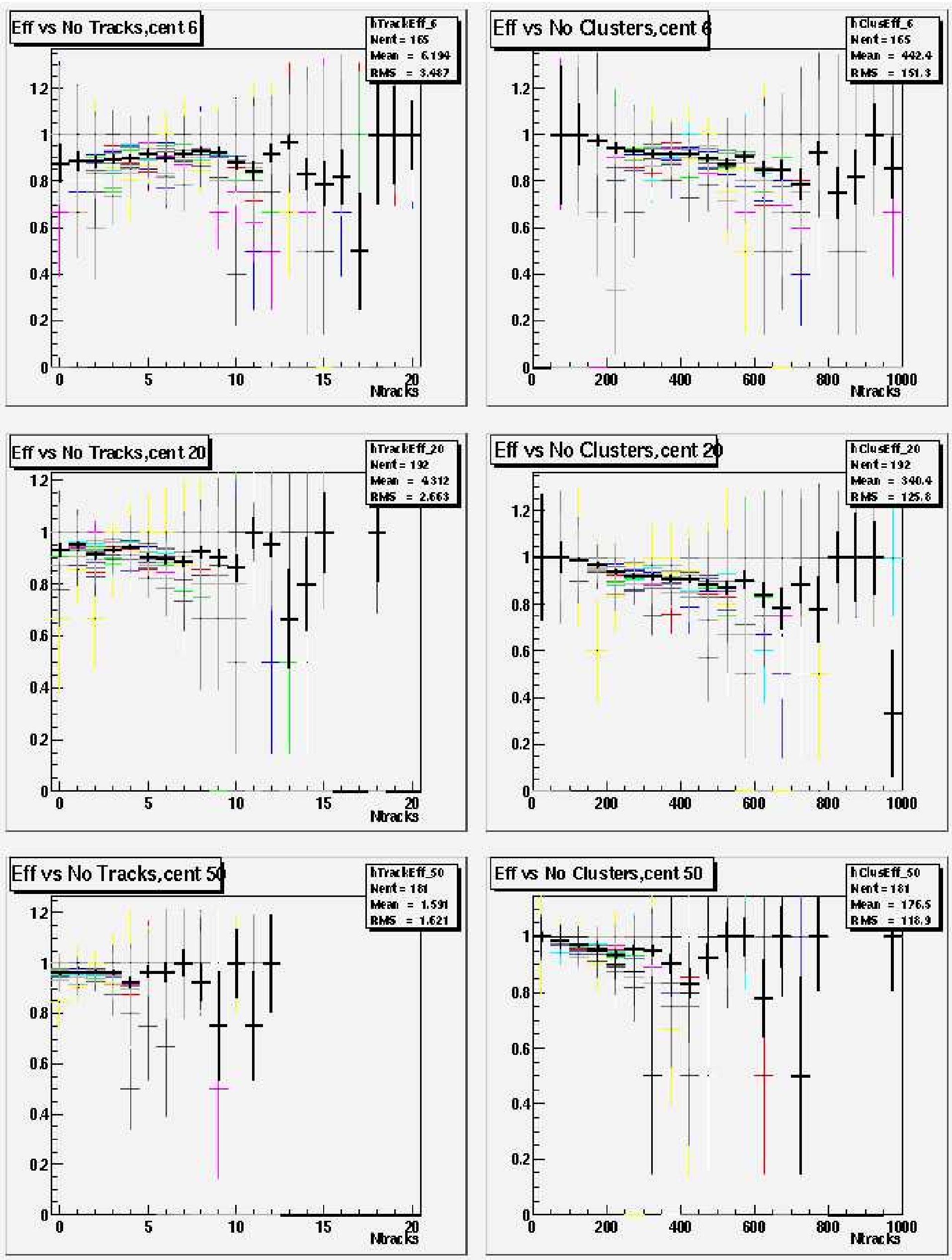

Figure 4: Efficiencies as a funcion of number of tracks and clusters for three centrality bins. The black points are the sums of the other points, which come from different data–sets. The left column shows the efficiency as function of the number of tracks, and the right as a function of the number of clusters in TPM1.

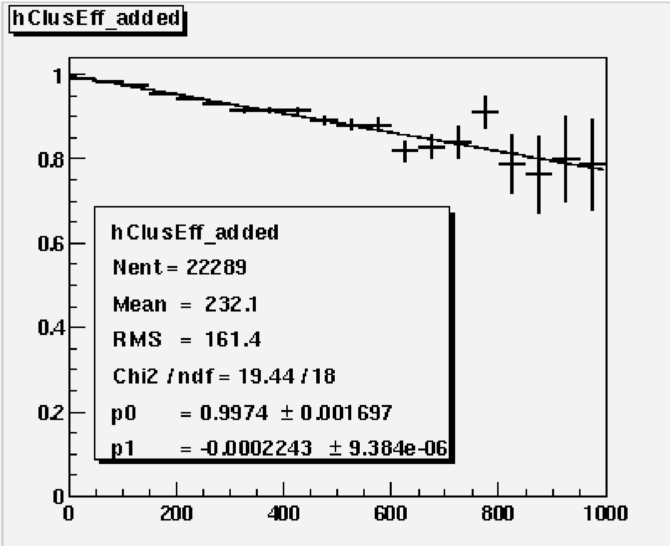

Figure 5: Our measured TrackFollowFinder efficiency as a function of $N_{clusters}$

Phew...

A brief walkthrough of the idea behind this:

1. $N^{true}_{events}(N_{tracks}, N_{clusters})$, the estimated true number of events with N_{tracks} and $N_{clusters}$, is calculated by dividing the observed number by the success probability[3] of TFF as a function of $N_{clusters}$.

2. For this exact $N_{clusters}$ we have an efficiency $\epsilon_{TFF}(N_{clusters})$. Dividing by this gives the real average number of tracks in events with N_{tracks} observed tracks.

3. Summing over the $N_{clusters}$–bins gives the total number of events with N_{tracks}, and multiplying by N_{tracks} and summing over it gives the corrected total number of tracks in TPM1.

4. Finally, dividing by the total number of events — including those where TFF failed — gives the number of tracks per event. [4]

4.4 Results

In table 4, we show the results of the analysis described above for the six centrality classes.

[3] $P_{success} = 1 - P_{crash}$

[4] For those who read my previous, first iteration of this report, this is the same as the Poisson–mean I used then except that now the efficiency correction is already added.

Class	$<\frac{N_{tracks}}{N_{events}}>$ Uncorrected	$<\frac{N_{tracks}}{N_{events}}>$ Corrected
1	4.027	4.58296
2	3.179	3.56572
3	2.569	2.82981
4	1.901	2.05834
5	1.386	1.48301
6	0.9841	1.04108

Table 4: Primary particles in TPM1, first uncorrected and then corrected for efficiency and TFF crashes. Restrained to a fiducial area of $x = \pm 10cm$ and $y = \pm 7cm$ at the midplane.

Class	$I_{DoubleGaussian}$	$I_{background}$	$\frac{I_{background}}{I_{DoubleGaussian}}$
1	6469.14	2641.95	0.40
2	2374.81	966.634	0.41
3	4401.55	1779.79	0.40
4	3037.58	1210.13	0.40
5	2117.14	855.635	0.40
6	1328.06	545.916	0.41

Table 5: Contribution of secondary tracks to our observed number of tracks in TPM1

5 Primaries to secondaries

We use the distance from nominal beam position in Y for all tracks to determine the ratio of primary to secondary particles. In figure 6, we have fitted this spectrum for all six centrality classes with the sum of two gaussians, one for the signal and one for the background. We integrate both gaussians and take the ratio of the integrals to get the background contribution to the number of tracks in TPM1. The results are shown in table 5.

It is clear that the contribution of secondary tracks to $\frac{dN}{d\eta}$ is independent of centrality class to within $\sim$1%.

6 Acceptance correction

The correction for geometric acceptance has two factors, coverage in ϕ and η-acceptance normalized to one unit of pseudorapidity.

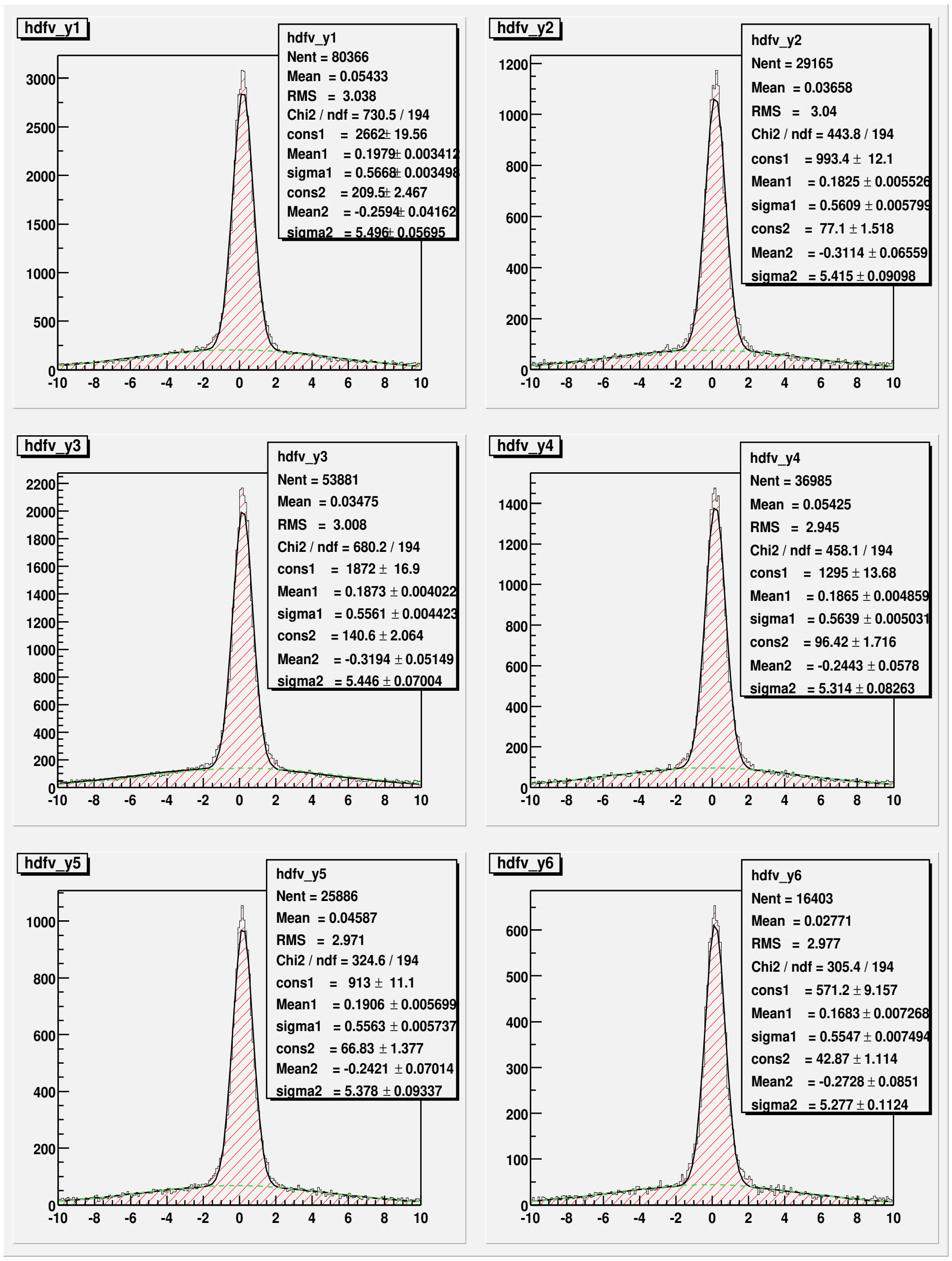

Figure 6: Distance from nominal y position for all tracks, fit with a double gaussian, for JHs centrality classes.

6.1 ϕ correction

The fiducial area in ϕ, taken at the mid-plane of TPM1 which is positioned at global x=95 cm, is ±7 cm. This gives

$$\Delta\phi = 2 \cdot tan^{-1}(\frac{7}{95}) = 8.43^o \tag{4}$$

6.2 η acceptance

The fiducial area in θ, again taken at the mid-plane of TPM1, is ±10 cm. The angles, and thus the conversion to pseudorapidity

$$\eta = -ln(tan(\frac{\theta}{2})) \tag{5}$$

are dependent on the z position of the vertex in a way symmetric about z=0, so we first calculate our η acceptance for z=0 cm and |z|=15 cm which is the vertex cut used in my analysis. This gives

$$\Delta\eta_{z=0} = 0.210 \tag{6}$$

and

$$\Delta\eta_{|z|=15} = 0.207 \tag{7}$$

i.e. almost equal. In the calculation below, we use the average of these two corrections:

$$\hat{\Delta\eta} = \sqrt{\frac{0.210^2 + 0.207^2}{2}} = 0.209 \tag{8}$$

6.3 Total geometric acceptance correction

The total geometric acceptance correction is therefore

$$f_{Geometry} = \frac{1}{\hat{\Delta\eta}} \cdot \frac{360}{\Delta\phi} = \frac{1}{0.209} \cdot \frac{360}{8.43} \approx 204 \tag{9}$$

7 Centrality dependent $\frac{dN}{d\eta}$

We are now in a position to calculate our observed charged particle multiplicity at midrapidity, including corrections for

- Crashes in TrackFollowFinder
- Tracking efficiency
- Contribution from secondaries
- Geometry

Class	$\frac{dN}{d\eta}$	$\frac{dN}{d\eta}/0.5\cdot < N_{part} >$
1	561	3.31
2	429	3.05
3	346	3.15
4	252	3.27
5	182	3.57
6	125	3.97

Table 6: Charged particle production at $\eta = 0$

The results are shown in table 6.

As you can see, the dependence of $\frac{dN}{d\eta}/0.5\cdot < N_{part} >$ with centrality is not what we'd expect. The reason for this, we believe, is that the centrality classes selected by GetCentrality are wrong — see discussion in [2]. The next section redoes the analysis with the new definition of centrality. (See also appendix C.)

8 Centrality classes and analysis — NBI style.

The NBI centrality classes ([2]) are defined as in table 7, where again $< N_{part} >$ is calculated as explained in B.

The analysis proceeds exactly as explained above, so I'll only give a summary–table of the numbers. See table 8.

Class number	% of min-bias	$< N_{part} >$ from K&N
1	0-5	344
2	5-10	288
3	10-15	241
4	15-20	199
5	20-30	154
6	30-40	102
7	40-50	63

Table 7: The centrality classes as defined by the method explained in [2]

As you can see, the $dN/d\eta$ dependence on centrality here looks much more like what we'd expect from the other experiments. Refer to figure 7 for a comparison with Quark Matter data from PHENIX and PHOBOS.

A point to note is that the background factor is here 7% lower than for the other centrality class selection, and the RMS of the peak of primary particles has dropped. The reason for this is not yet understood. We include the plots used to determine this factor for reference — see figure 8. A change here will of course alter the absolute value of the $dN/d\eta$ points, but a change of more than 10% is unlikely.

9 Conclusions and comments

Looking at figure 7, it looks like we are approaching a result very comparable to what the other experiments are presenting. We are currently a bit higher in absolute value, but this may well change as we make a bit more precise efficiency and background corrections. Also note that there are no systematic or statistical uncertainties included in the plot — doing this will almost certainly make the three measurements consistent.

It therefore looks to us like the centrality algorithm presented by Claus and Christian is closer to the mark than the one we had before. If this will change even more when we can introduce a centrality cut based on both the tiles and the strips as proposed by S. Sanders, time will tell.

Most fundamental problems connected with extracting $\frac{dN}{d\eta}/ < N_{part} >$ now seem solved, especially thanks to the new centrality determination by the authors of [2]. We believe that we can have this data set ready for publication in the planned $dN/d\eta$ paper within a very short time.

Issues currently under investigation:

- When looking at the number of tracks crossing the TPC midplane, there is a dip at one particular x value. This effect, probably due to hardware problems, must be incorporated in the efficiency calculations.
- We need to understand the background subtraction better. The difference in BG gives a roughly 7-10% difference in $dN/d\eta$ for the two centrality selections, and the reason for this must be found.
- The systematic uncertainties must be calculated.

Class number	$< \frac{N_{tracks}}{N_{events}} >$	B.g. corr.	$dN/d\eta$	$\frac{dN}{d\eta}/0.5\cdot < N_{part} >$
1	4.88395	0.34	658	3.82
2	3.76938	0.34	508	3.53
3	3.03918	0.35	403	3.34
4	2.46042	0.33	336	3.38
5	1.79014	0.32	248	3.22
6	1.13006	0.33	154	3.02
7	0.684282	0.33	94	2.98

Table 8: First results with NBI centrality. $< \frac{N_{tracks}}{N_{events}} >$ is corrected for tracking efficiency and TFF crashes.

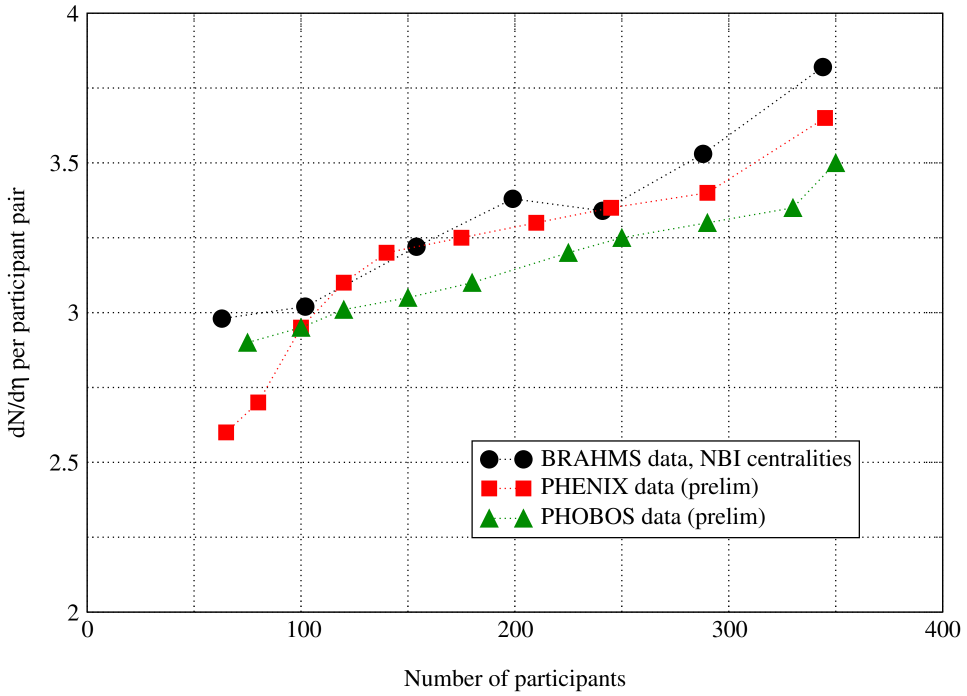

Figure 7: $dN/d\eta$ per participant pair — a comparison of the data presented here with the data presented at QM2001 by PHENIX and PHOBOS

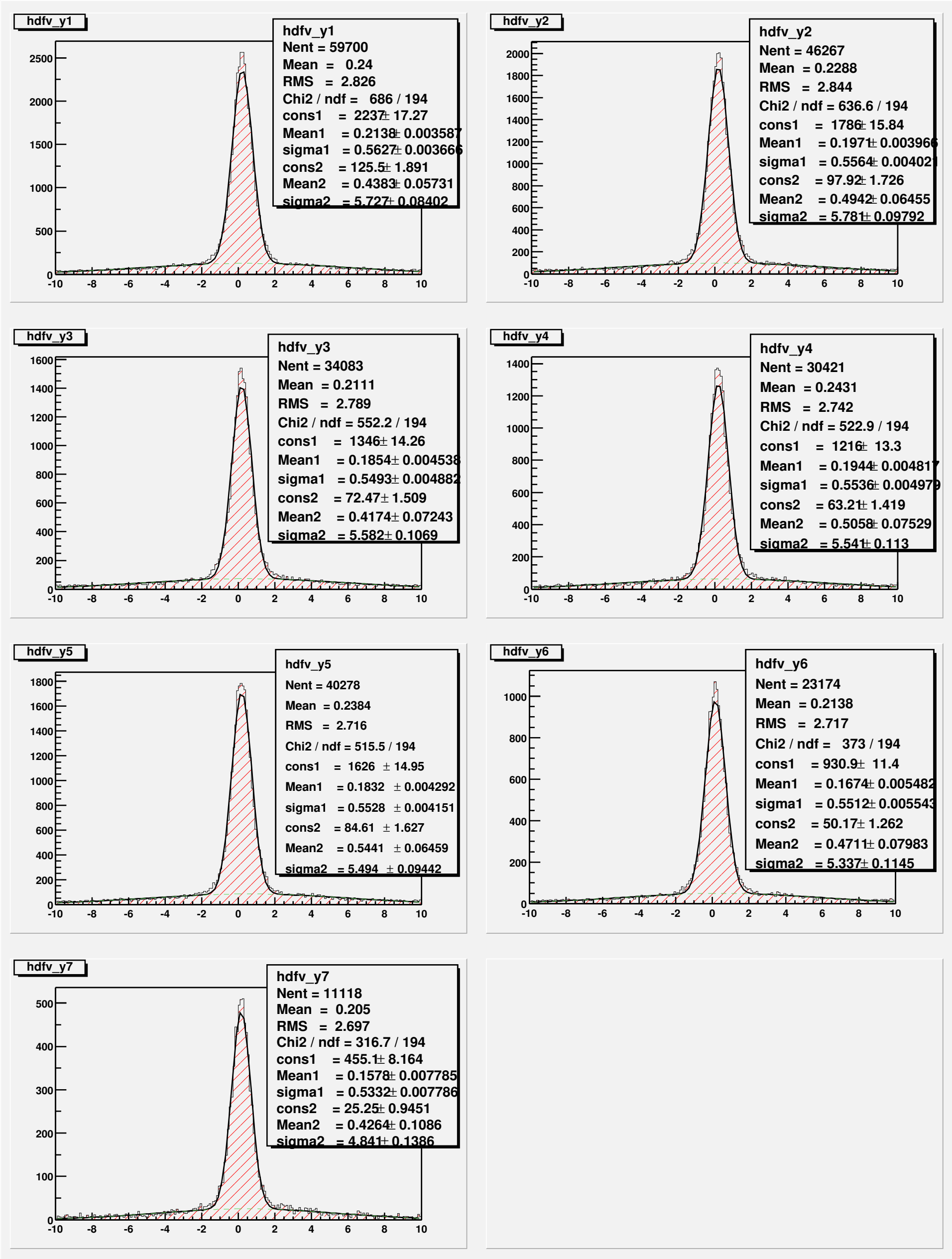

Figure 8: Distance from nominal y position for all tracks, fit with a double gaussian, for the NBI centrality classes.

- We need a better $< N_{part} >$, preferably from HIJING simulations of the tiles. Determining $< N_{part} >$ has proved to be *crucial* to the shape of the $dN/d\eta$ curve.

- When we say e.g. 5%-10% central, do we mean 5% of the *nominal* nuclear cross–section, or 5% of our measured ≈97% of the cross–section? This difference may influence the most peripheral bin by up to ≈5%.

- ... and probably many more small corrections.

A The new TPC digitize parameters

A quick determination of the digitization parameters most important for tracking efficiency have been determined by digitizing Hijing events in TPM1 and comparing histograms of appropriate quantities for simulated and real data (the first few sequences of run 2530, with Flemming's BramSelectMRS event cuts, clustering and tracking parameters). Only tracks with small slopes have been taken into account in this comparison. A vertex cut was imposed to exclude secondary tracks, in particular the confusing δ-electrons with their low charge – high residual contributions. A selection of the compared quantities are shown in Figure 9, where the experimental data are depicted in red, the simulated ones in blue[5].

The average charge per cluster is tuned by looking at distributions for cluster ADCSum averaged over each track, excluding the 30% of clusters with highest ADCsum to reduce the Landau tail. The ADCsum is divided by tracklength per cm of z-projection ($\approx 1.$) The comparison with experimental data (see panel 1 in Fig. 9) shows that the ADCgain must be increased with a factor about $3 - 3.5$.

```
Float_t adcgain = tpm1_digitize->GetDetectorParamsTPC()->GetADCGain();
Float_t amplify = 3.4;
tpm1_digitize->GetDetectorParamsTPC()->SetADCGain(adcgain*amplify);
```

Drift velocities were tuned by looking at cluster widths as functions of drift distance, or Y_0 position of the track in the TPC midplane. The transverse and longitudinal cluster width (pad and timebin direction, respectively – see panels 2 and 3 in Fig. 9.) depend approximately linearly on Y_0:

$$\sigma_T \approx \sigma_{PRF}(1 + \frac{1}{2}\frac{D_T^2}{\sigma_{PRF}} s_{drift}), \tag{10}$$

$$\sigma_L \approx \sigma_0(1 + \frac{1}{2}\frac{D_L^2}{\sigma_0} s_{drift}) \tag{11}$$

The observed widths and their drift distance dependence were well reproduced with the following parameters:

[5] Or gray and black resp. on a b&w printout

```
Float_t sigma_prf = tpm1_digitize->GetDetectorParamsTPC()->GetAnodeGap();
tpm1_digitize->GetDetectorParamsTPC()->SetAnodeGap(0.85*sigma_prf);
tpm1_digitize->GetDetectorParamsTPC()->SetDtrans(0.06);
Float_t sigma_0 = tpm1_digitize->GetDetectorParamsTPC()->GetShapingTime();
tpm1_digitize->GetDetectorParamsTPC()->SetShapingTime(1.3*sigma_0);
tpm1_digitize->GetDetectorParamsTPC()->SetDlong(0.10);
```

The default **BrDigitizeTPC::Event()** algorithm represents the time dependence in cluster shape with a Gaussian. We found that the behavior of the longitudinal cluster width (and the residuals, see below) was much more easily described when a shape composed of two exponential tails meeting in a cusp was used for the cluster width. This is also, of course, only a rough approximation to the real shape. The choice of shape has negligible influence on the tracking efficiency.

The total distribution of small slope tracks with vertex cut is shown in panel 4 of Fig. 9. For negligible slopes, the residuals in transverse and longitudinal direction can be described by:

$$\delta_T^2 \approx \delta_{T,0}^2 + \frac{D_T^2 \cdot s_{drift}}{l \cdot N_{e^-}}, \tag{12}$$

$$\delta_L^2 \approx \delta_{L,0}^2 + \frac{D_L^2 \cdot s_{drift}}{l \cdot N_{e^-}}. \tag{13}$$

where N_{e^-} of course is proportional to the cluster charge. Here it was necessary to include constant terms corresponding to $\delta_{T,0}$ and $\delta_{L,0}$ to reproduce the experimental residuals as functions of average cluster charge (see panel 5 and 6 in Fig. 9.) To this purpose, new data members **fResT0** and **fResL0** were introduced in **BrDigitizeTPC**. A high noise level, which is indeed known to be the case in TPM1, improved the correspondence for tracks with low average cluster charges, where the cluster positions are more sensitive to noise.

```
tpm1_digitize->SetResT0(0.050);
tpm1_digitize->SetResL0(0.090);
tpm1_digitize->SetNoise(7.);
```

B Calculating the number of participants

In [1], D. Kharzeev and M. Nardi present calculations based on the Glauber model of the number of participants for different centralities at RHIC energies. Their numbers for $\sqrt{s}$ = 130 GeV is shown in table 9, but note that this is the *integrated* $< N_{part} >$ for e.g. centrality 0–20, 0–45 etc.

To get the correct numbers for our centrality classes we really have to simulate the tiles and extract the corresponding $< N_{part} >$ from HIJING, but

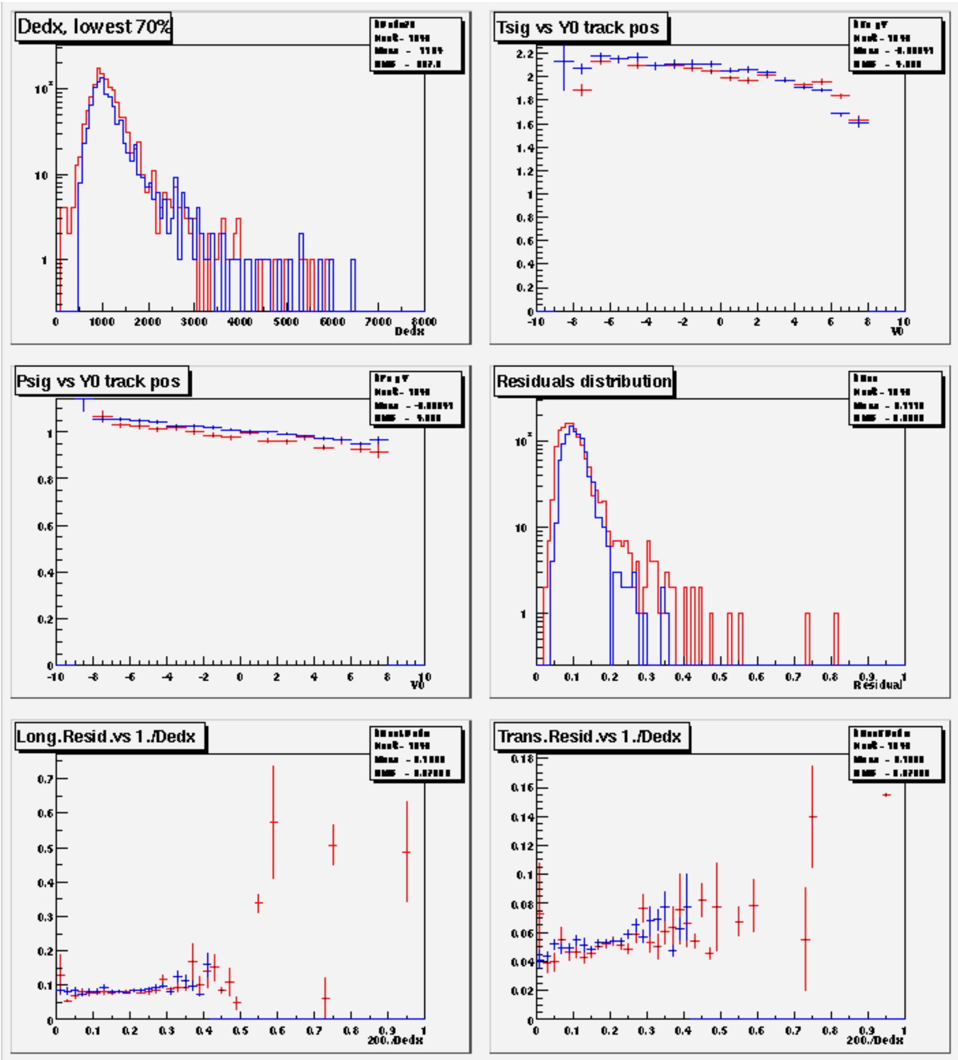

Figure 9: Comparison of experimental (red or gray) and simulated (blue or black) track avereaged cluster characteristics

as a first approximation which is probably correct to within a few percent, we have assumed that the following can be done:

$$b\cdot < N_{part} >^{b\%}_{K\&N} = (b-a)\cdot < N_{part} >^{a\%--b\%}_{Cent.Class} + a\cdot < N_{part} >^{a\%}_{K\&N} \tag{14}$$

i.e. just assume a linear dependence of $< N_{part} >$ on centrality. This works well for all bins except for the first one, and there the integrated number in [1] is the correct one anyway.

An example:

$$30\cdot < N_{part} >^{30\%}_{K\&N} = (30-20)\cdot < N_{part} >^{20\%-30\%}_{Cent.Class} + 20\cdot < N_{part} >^{20\%}_{K\&N} \tag{15}$$

$$< N_{part} >^{20\%-30\%}_{Cent.Class} = \frac{30\cdot < N_{part} >^{30\%}_{K\&N} - 20\cdot < N_{part} >^{20\%}_{K\&N}}{10} \tag{16}$$

Inserting numbers, this gives $< N_{part} > \approx 154$ for our 20%–30% centrality class.

Note again that this is an *approximation*, and that the numbers should really come from simulating the tiles.

Table 10 gives the calculated numbers for the different centrality classes.

C A change in JH's centrality classes

Looking at [2], Claus and Christian compare the GetCentrality method with their MultRdo–method. If we look at their figure 5, we can make a quick–and–dirty estimate of what centralities the GetCentrality–classes really correspond to. Doing this and then calculating the Kharzeev and Nardi $< N_{part} >$ as explained in appendix B, we get quite another dependence of $\frac{dN}{d\eta}/0.5\cdot < N_{part} >$ with $< N_{part} >$ — see table 11.

Since this calculation is *very* sketchy it should not be taken quantitatively, but it is reassuring to see that this approach indeed gives the expected rise in $dN/d\eta$ per participant pair with $< N_{part} >$.

References

[1] Dmitri Kharzeev and Marzia Nardi. Hadron production in nuclear collisions at RHIC and high density QCD. 2000.

[2] Claus Ekman Jørgensen and Christian Holm Christensen. Centrality estimates using the Scintilator Tiles in the Multiplicty Array. Technical Report 22, February 2001.

%	$< N_{part} >$
5	344
6	339
10	316
15	291
20	268
25	248
30	230
35	213
40	198
45	183
50	171
60	149
70	131
80	115
100	93

Table 9: Integrated $< N_{part} >$, as given in [1]

Centrality class	JH's GetCentrality	NBI TileRdo Centrality
1	339	344
2	281.5	288
3	220	241
4	154	199
5	102	154
6	63	102
7	—	63

Table 10: Approximate $< N_{part} >$ for our centrality classes

Centrality class	$< N_{part} >$	$\frac{dN}{d\eta}/0.5\cdot < N_{part} >$
1	335	3.35
2	278	3.08
3	229	3.02
4	174	2.89
5	131	2.78
6	96	2.60

Table 11: $dN/d\eta$ per participant pair if we take the centrality classes to correspond to figure 5 in [2] and use the calculation described in app. B to get $< N_{part} >$.