Messing around with “different” centrality selections

W.J. Llope
bulkcorr PWG meeting
8/17/2011

Standard centrality definitions are based on refmult cuts (Ntracks \(|\eta|<0.5\))

“higher moments” analyses make a centrality selection then proceed to calculate the moments of the distributions of the E-by-E multiplicity of specific particles or multiplicity differences or ratios.

Note: these moments are not “central moments” so K,S,… moments are sensitive to the mean values of the multiplicities.

Autocorrelation between the centrality selection and the K,S moments values?

e.g. Nihar & Lizhu define “refmult2” (0.5<|\eta|<1.0) for centrality and then calculate the moments in the range |\eta|<0.5.
We have other variables that should carry centrality information
FTPC Ntracks, BBC SumADC, ZDC SumADC, VPD SumADC or SumToT, etc…
But not all of these variables are non-zero for all root-s values….

Would still be interesting to compare the moments values when using different centrality definitions (at least at the root-s values where these different definitions are possible)….
Au+Au, 200 GeV, Run-10
ZDC ADC sum vs BBC ADC sum

BBC ADC sum vs refmult

ZDC ADC sum vs refmult

Au+Au, 62.4 GeV, Run-10

BBC gain change in the middle of the data-taking... ...to correct:
if (runid<11089075) bbcsumadc *= 2.1
Au+Au, 39 GeV, Run-10
Algorithm:

1. determine “ridge curve” of each distribution graphically
2. find \( x_{\text{max}} \) and \( y_{\text{max}} \) from the curve, and do everything in \((x/x_{\text{max}}, y/y_{\text{max}})\) space
3. for every point on the curve, calculate the running path-length along the curve
4. for every point in the scaled space, locate closest point on the curve and assign this point in the space to this point on the curve
5. for every point on the curve, add up the number of events from every point in the space assigned to this point on the curve
6. “\( dN_{\text{evt}}/dS \)” is then integrated to produce percentages (0-5%, 5-10%, etc) which defines S-value cuts (in similarity to refmult cut windows)
7. fill a 2D of the original space where the Z-value is the centrality bin (1-16)
8. use this 2D in another pass through the data to look up centrality bin number for any given value of \((x, y)\)

then use three different “Centrality Styles”
1 … 1D space is just refmult
2 … 2D space is ZDC sum ADC vs BBC sum ADC
3 … 2D space is BBC sum ADC vs refmult

and look at the refmult distributions for styles 2 and 3….
Au+Au 200 Run10

ZDC ∑ADC vs BBC ∑ADC

ridge curve
BBC $\sum$ADC vs RefMult

Au+Au, 200 GeV, Run-10

Entries 997577

75-80%

0-5%

5-10%

10-15%

15-20%

20-25%
refmult distributions....... Au+Au, 200 GeV, Run-10
refmult distributions...... Au+Au, 62.4 GeV, Run-10

refmult cuts

ZDC vs BBC cuts

BBC vs refmult cuts
refmult distributions...... Au+Au, 39 GeV, Run-10

refmult cuts

ZDC vs BBC cuts

BBC vs refmult cuts
ZDC is inefficient for
19.6, 11.5, & 7.7 GeV

BBC vs refmult @ these energies:
if you want to try this in your code, on RCF get three root files:
/star/u/llope/cent/cent_16.root        Au+Au, 200 GeV, Run-10
/star/u/llope/cent/cent_17.root        Au+Au, 62.4 GeV, Run-10
/star/u/llope/cent/cent_18.root        Au+Au, 39 GeV, Run-10

before event loop:
kCentralityStyle=2        TH2D* hcent = (TH2D*)file->Get("hcenztddcbbc_use");
kCentralityStyle=3        TH2D* hcent = (TH2D*)file->Get("hcenbbcrefmult_use");
                        then, hcent->SetName("hcent");

inside event loop:
if (root-s==62.4){ if (runid<11089075){ adjustedbbcadc = 2.1*bbcsadmc } }

then:
if kCentralityStyle=2        int ibxy = hcent->FindBin(adjustedbbcadcsum,zdadcsum)
if kCentralityStyle=3        int ibxy = hcent->FindBin(refmult,bbcadcsum)

then
int kCentrality = hcent->GetBinContent(ibxy) – 4;       // bin s start at 95-100%

kCentrality<1 -> reject
kCentrality = 16 means 0-5%, & kCentrality = 1 means 75-80%