Di-hadron tutorial

Zhiyong Lu
NBI & CIAE

Why do we need di-hadron correlation

In the small collision systems, such as pp, p–Pb, OO and Ne–Ne collisions, are dominated by so-called non-flow contamination.

Therefore, it is useful to construct standard two-particle correlations which provides additional information about the shape of the correlation function.

$$C(\Delta\eta, \Delta\phi)$$

$\Delta\phi$: Fourier coefficients are extracted based on $\Delta\phi$ distribution
$\Delta\eta$: Useful for selecting the long/short range correlation

How to get the correlation function

The correlation function C(Δη, Δϕ) is a distribution of particle pairs as a function of difference between their pseudorapidities Δη and azimuthal angles Δϕ.

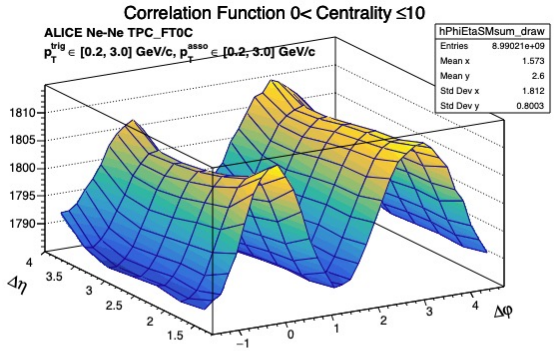
Every pair consists of a leading particle, referred to as the trigger particle, and an associated particle. The shape of C(Δη, Δϕ) depends on pT of both trigger and associated particles and the centrality/multiplicity class of the event. Moreover, the shape is significantly affected by the acceptance of the detector. For this reason, an event mixing technique is commonly used.

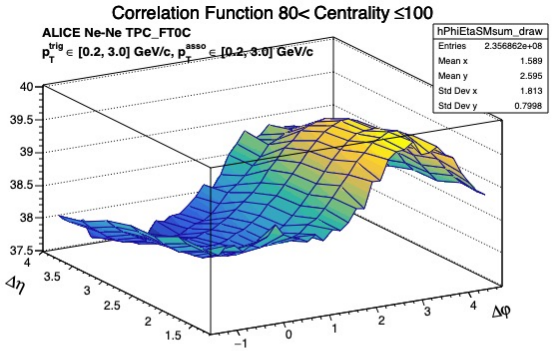
1. SE: in the same event from all the pairs of different trigger and associated particles.
2. ME: in different events, but with similar proprieties, such as the position within the detector (primary vertex) or the collision centrality. The trigger particles originate from the studied event, while the associated particles are reconstructed in different events.

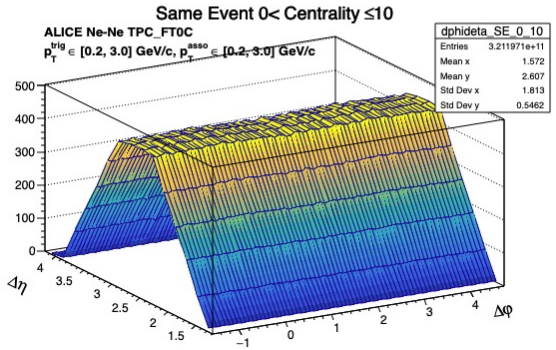
$$C(\Delta\eta,\Delta\varphi) = \frac{1}{N_{\text{trig}}} \sum_{\text{PV}_z} \frac{SE(\Delta\eta,\Delta\varphi)}{\alpha\, ME(\Delta\eta,\Delta\varphi)},$$

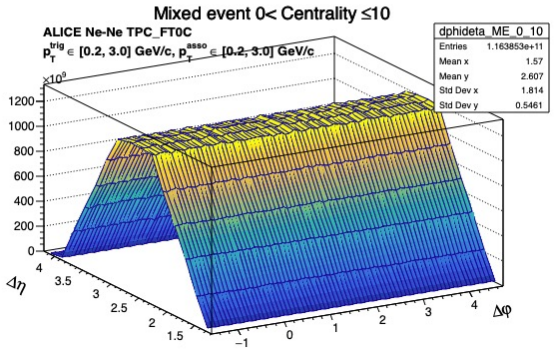
How to get the correlation function

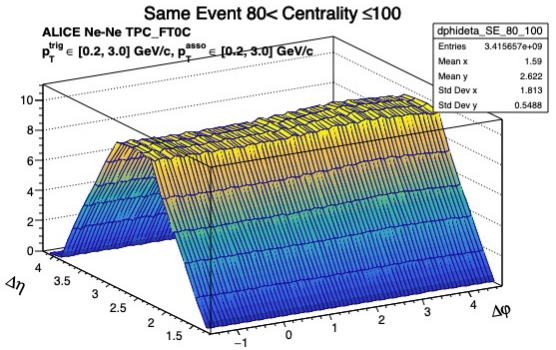
$$C(\Delta\eta, \Delta\varphi) = \frac{1}{N_{\text{trig}}} \sum_{\text{PV}_z} \frac{SE(\Delta\eta, \Delta\varphi)}{\alpha\, ME(\Delta\eta, \Delta\varphi)},$$

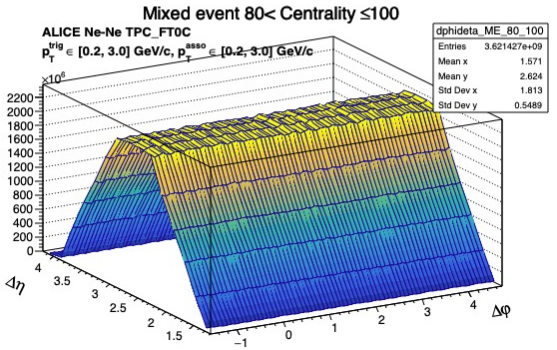












How to get the correlation function

$$C(\Delta\eta, \Delta\varphi) = \frac{1}{N_{\text{trig}}} \sum_{\text{PV}_z} \frac{SE(\Delta\eta, \Delta\varphi)}{\alpha\, ME(\Delta\eta, \Delta\varphi)},$$

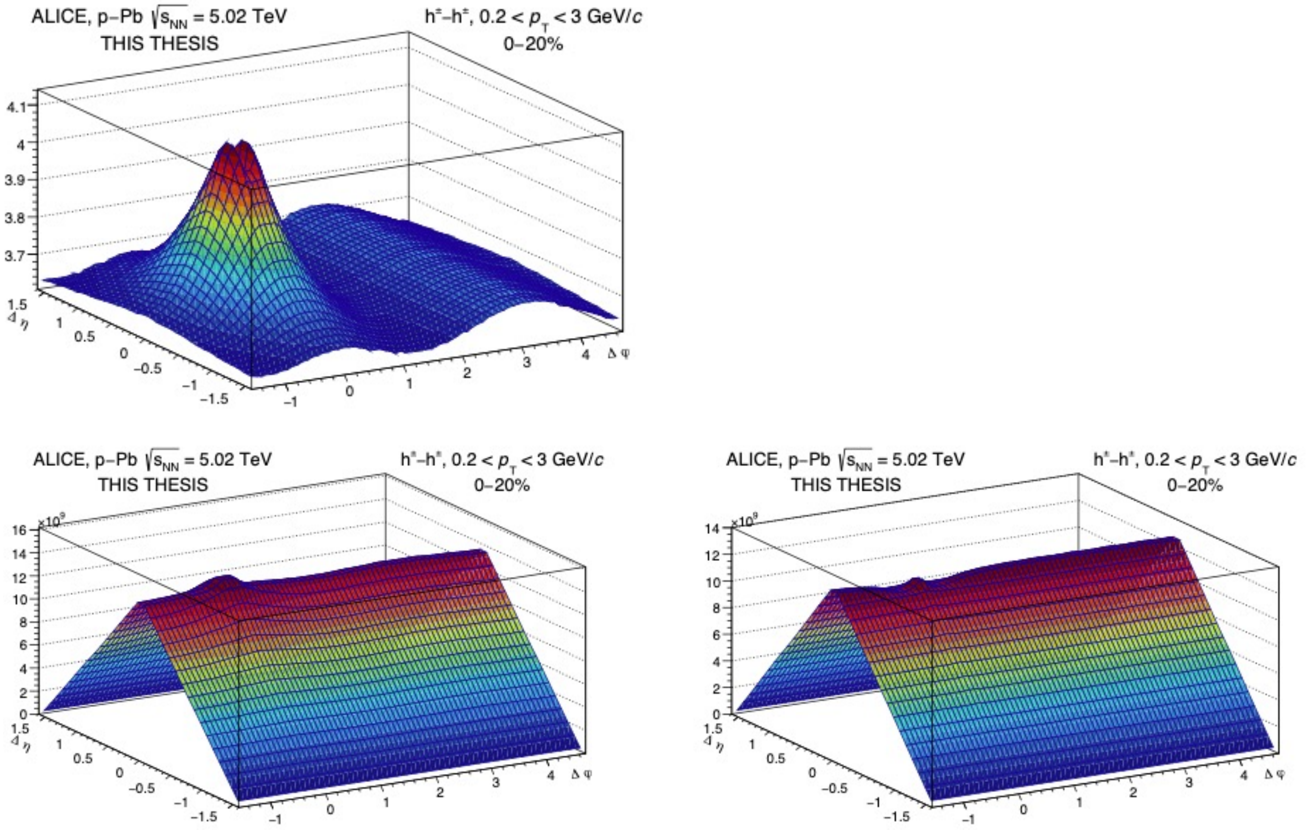


FIGURE 3.2: Distribution of $SE(\Delta\eta, \Delta\varphi)$ (left) and $ME(\Delta\eta, \Delta\varphi)$ (right)

How to get VnΔ

the two-dimensional correlation function C(Δη, Δϕ) is projected into the Δϕ axis. The projected distribution Y(Δϕ) can be described by a Fourier decomposition as

$$Y(\Delta\varphi) = a_0 + 2\sum_{n=1}^{\infty} a_n \cos(n\Delta\varphi).$$

$$V_{n\Delta} = \frac{a_n}{a_0}$$

$$v_n\{2\mathrm{PC}\} = \sqrt{V_{n\Delta}}.$$

$$v_n\{2\mathrm{PC}\}(p_\mathrm{T}) = V_{n\Delta}(p_\mathrm{T})/\sqrt{V_{n\Delta}}.$$

$$v_n[2\mathrm{PC}](p_\mathrm{T}) = \sqrt{V_{n\Delta}(p_\mathrm{T})},$$

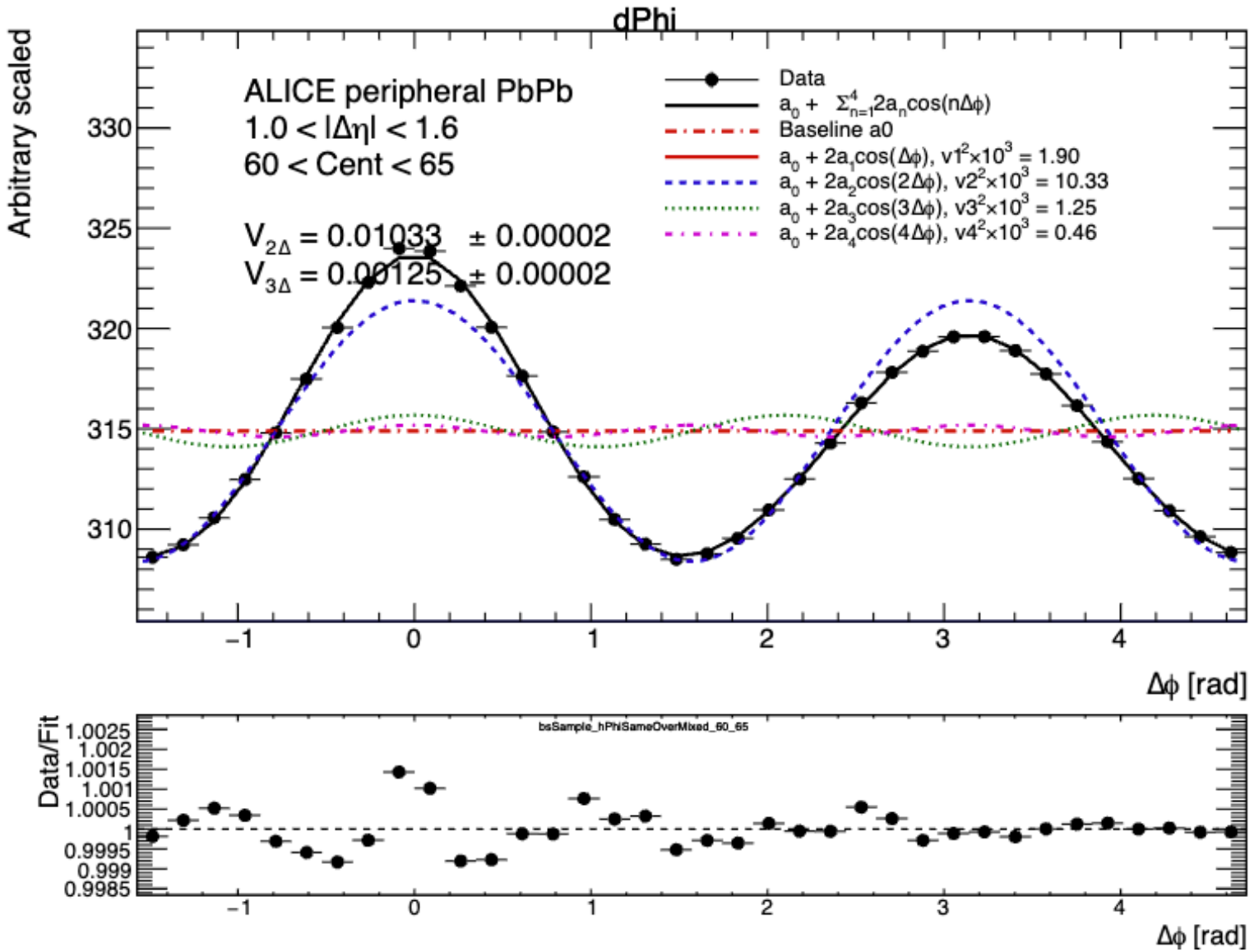
This is called Fourier fit method

(pT) means you need to do this procedure for every narrow pT bin

How to get VnΔ

$$Y(\Delta\varphi) = a_0 + 2\sum_{n=1}^{\infty} a_n \cos(n\Delta\varphi).$$

$$V_{n\Delta} = \frac{a_n}{a_0}$$



Template fit method

In the low multiplicity (peripheral) subtraction method, it is assumed that there is weak to no flow signal.

The correlations from high multiplicity (HM) collisions are assumed to be a superposition of scaled correlation from low multiplicity (LM) collisions and a cosine modulation that represents the flow contribution,

$$Y(\Delta\varphi)^{\mathrm{HM}} = FY(\Delta\varphi)^{\mathrm{LM}} + G^{\mathrm{tmp}}\left[1 + 2\sum_{n=2}^{\infty} V_{n\Delta}^{\mathrm{tmp}}\cos(n\Delta\varphi)\right],$$

F, G are scaling factors

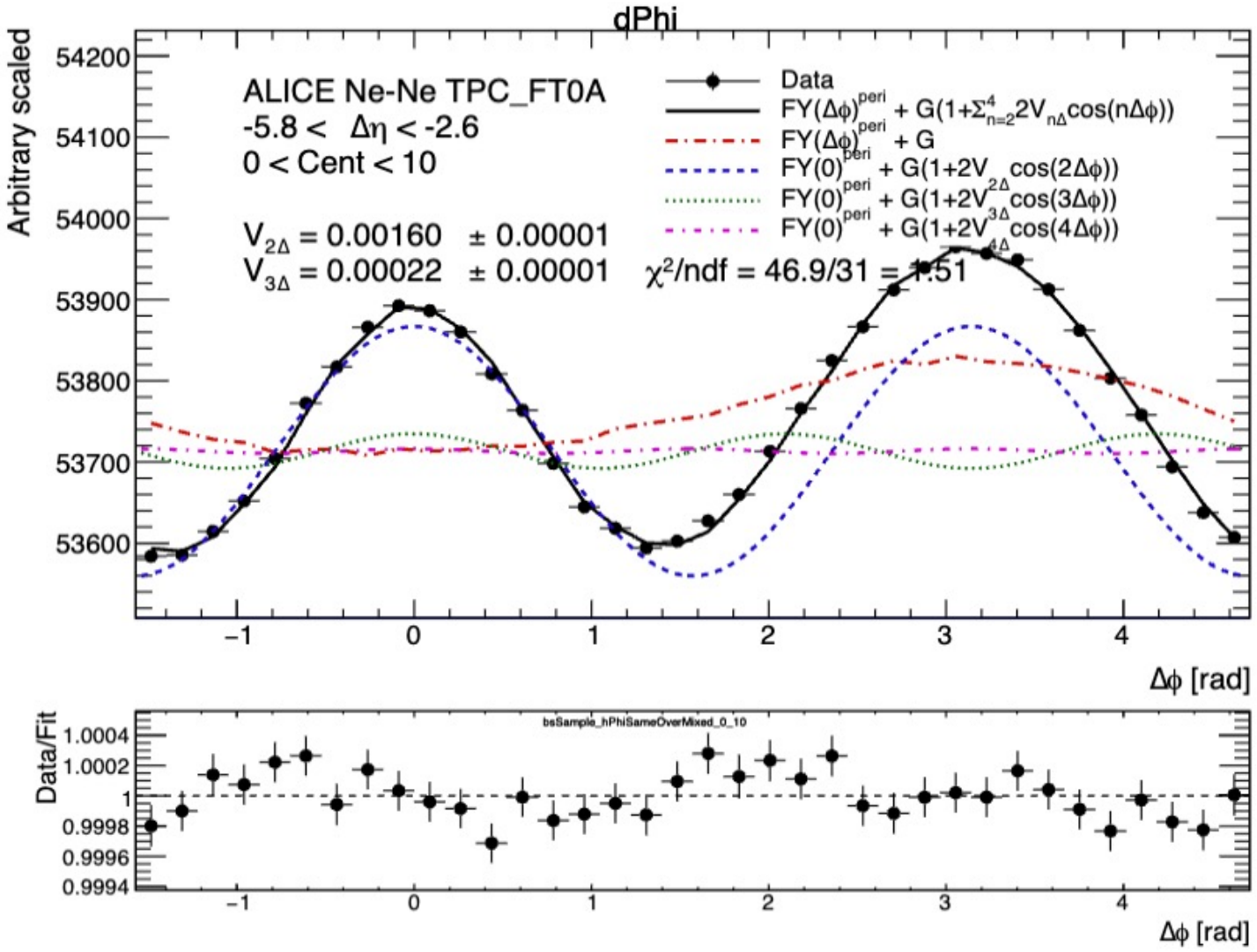
$$v_n\{2\mathrm{PC}\} = \sqrt{V_{n\Delta}}.$$

$$v_n\{2\mathrm{PC}\}(p_{\mathrm{T}}) = V_{n\Delta}(p_{\mathrm{T}})/\sqrt{V_{n\Delta}}.$$

$$v_n[2\mathrm{PC}](p_{\mathrm{T}}) = \sqrt{V_{n\Delta}(p_{\mathrm{T}})},$$

VnΔ in TPC-FT0A

$$Y(\Delta\varphi)^{\mathrm{HM}} = FY(\Delta\varphi)^{\mathrm{LM}} + G^{\mathrm{tmp}}\left[1 + 2\sum_{n=2}^{\infty} V_{n\Delta}^{\mathrm{tmp}} \cos(n\Delta\varphi)\right],$$

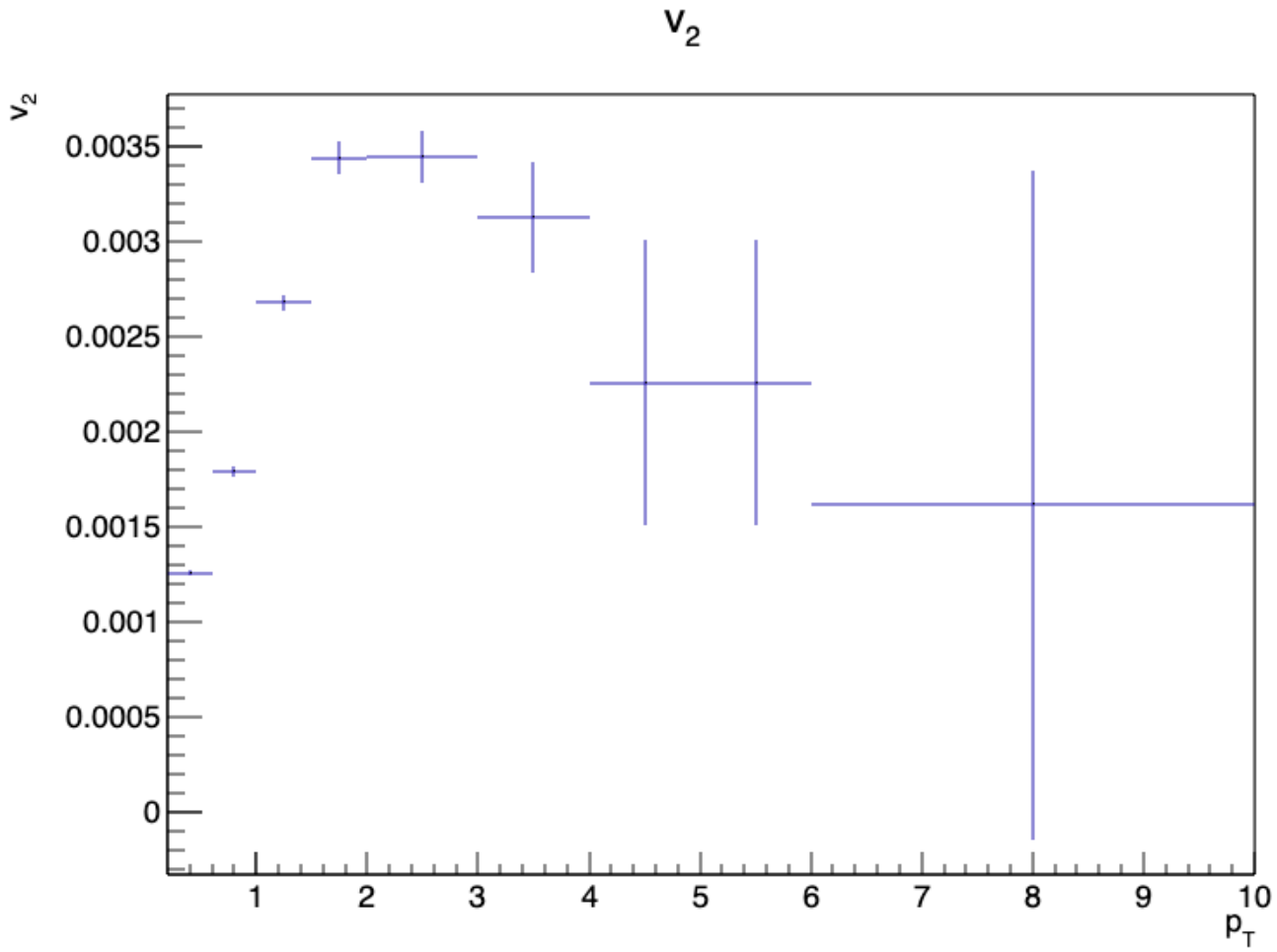


$$v_n\{2\mathrm{PC}\} = \sqrt{V_{n\Delta}}.$$

$$v_n\{2\mathrm{PC}\}(p_{\mathrm{T}}) = V_{n\Delta}(p_{\mathrm{T}}) / \sqrt{V_{n\Delta}}.$$

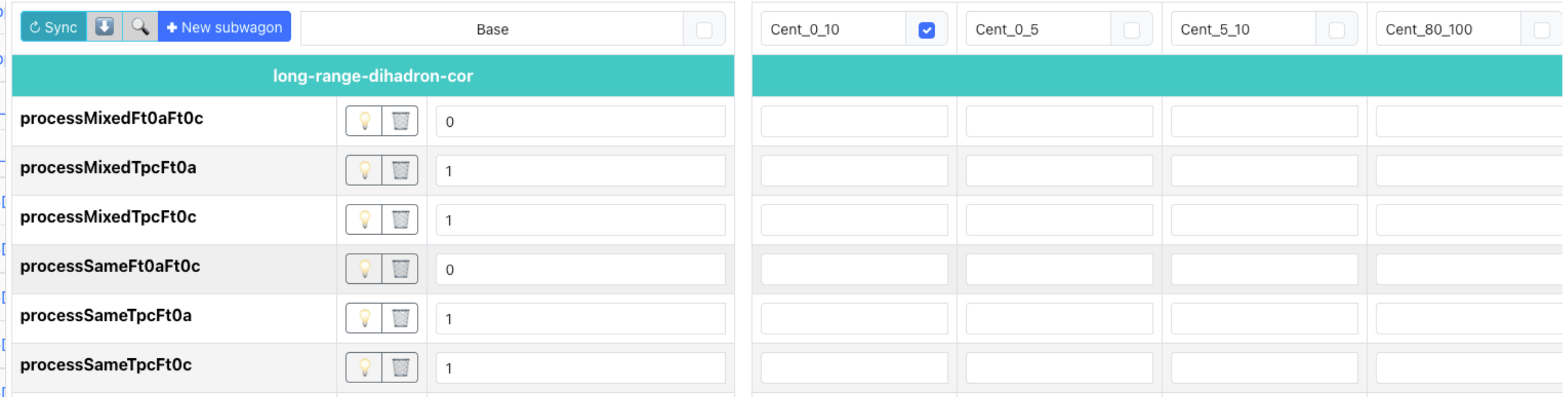
$$v_n[2\mathrm{PC}](p_{\mathrm{T}}) = \sqrt{V_{n\Delta}(p_{\mathrm{T}})},$$

VnΔ in TPC-FT0A



Usage of hyperloop

Use this train for reference: https://alimonitor.cern.ch/hyperloop/train-run/532067/general
You can clone this train and modify it for your own need



Subwagon naming (required because this is used in post-processing code)
Centrality dependence: Cent_*_*
Multiplicity dependence: Mult_*_* (you need to set **cfgSelCollByNch** to 1, and set **cfgCutMultMin/Max** correspondingly if you want to switch to mult dependence)

Usage of post-processing code

Long range version is still under construction, please waiting…
Basic usage:
change these in Process_dPhidEta.cxx:

```
inputList.push_back(InputUnit("LHC25af_pass1_532067", kTPCFT0A, kCent, kPtDiffOff, 0, 10));
inputList.push_back(InputUnit("LHC25af_pass1_532068", kTPCFT0A, kCent, kPtDiffOff, 80, 100));
inputList.push_back(InputUnit("LHC25af_pass1_532067", kTPCFT0C, kCent, kPtDiffOff, 0, 10));
inputList.push_back(InputUnit("LHC25af_pass1_532068", kTPCFT0C, kCent, kPtDiffOff, 80, 100));
```

- “LHC25af_pass1_532068”: Superfix of your train output file, fullname is “AnalysisResults_ LHC25af_pass1_532068.root”
- kTPCFT0A: enum for the code to know which correlation you want, change to kTPCFT0C if you want
- kPtDiffOff: switch of using pT-diff results. (!!!: you need to run kPtDiffOff first before kPtDiffOn because reference is based on kPtDiffOff)
- 0, 10: the Cent/Mult range, should be corresponding to your subwagon naming

Usage of post-processing code

Process_dPhidEta.cxx: produce C(Δη, Δϕ) and Y(Δϕ)

Process_CreatBootstrapSample.cxx: produce bootstrap samples.

Process_TemplateFit.cxx: perform template fit to get your VnΔ

Their input structure are written in a very similar way. Once you know how to use the first one then the other could be easily mimic.