Nuclei wagon for MPDRoot

V. Kireyeu

21.12.23

Introduction

The new "wagon" for the light nuclei (d, He) analysis must be implemented within the MpdRoot train-like analysis chain. The nuclei wagon should:

- Be **highly configurable** to avoid the unnecessary source code recompiling.
- Be **highly automated** for the same reasons.
- Be **well documented**.
- Provide the **phase-space distributions** for the particles of interest.
- Provide the **TPC, ToF, PID, DCA efficiencies and PID contamination** within the same phase-space bins as the particles distributions **for the final results** corrections.

The first version of the "nuclei" MpdRoot wagon will be presented in this talk.

Wagon description

Wagon structure and logic

- Initialization: settings a read from the JSON-file, histograms are booked for each:
  - particle
  - centrality bin
  - PID method (MC PDG, evPID wagon):
  - PID mode for the evPID case:
    - "All you can take"
    - "Competitive"
- Event processing:
  - Events are checked for the ”event quality”.
  - Tracks are checked for the ”track quality”.
  - Centrality bin is selected.
  - Particles are identified (MC, evPID).
  - Histograms for the identified histograms are filled.
- Helper subroutines for the configuration reading, ”quality” checks, centrality check, particles selection etc.

Configuration: global, event cuts

```
"Verbose": "1",
"N_MPD_PID_Particles": "8",
"make_MC": "1",
"make_PID": "1",
"make_Efficiency": "1",
"competitive_pid": "0",
"use_pt_corrections": "0",
"pt_corrections_file": "pt_corrections.root",
"Events": {
    "PrimaryVertexZ": "100",
    "Centrality": [[0, 10], [10, 20], [20, 30], [30, 40]]
},
```

Configuration: track quality, PID

```
"Tracks": {
    "NHits": "20",
    "NSigmaDCAx": "2",
    "NSigmaDCAy": "2",
    "NSigmaDCAz": "2",
    "LowPtCut": "0.05"
},
"PID": {
    "TPCSigma": "2",
    "TOFSigma": "2",
    "TOFDphiSigma": "3",
    "TOFDzSigma": "3"
},
```

Configuration: particles of interest

```
"Particles": {
    "p": {
        "PDG": "2212",
        "Mass": "0.938",
        "Charge": "1",
        "Enum": "3",
        "pt_bins": [60, 0.0, 6.0],
        "rapidity_bins": [60, -3.0, 3.0]
    },
```

Configuration: particles of interest

```
"d": {
    "PDG": "1000010020",
    "Mass": "1.876",
    "Charge": "1",
    "Enum": "4",
    "pt_bins": [30, 0.0, 6.0],
    "rapidity_bins": [60, -3.0, 3.0]
},
```

Configuration: particles of interest

```
"pim": {
    "PDG": "-211",
    "Mass": "0.1395",
    "Charge": "1",
    "Enum": "1",
    "pt_bins": [60, 0.0, 6.0],
    "rapidity_bins": [60, -3.0, 3.0]
},
```

Usage

Dependencies:

- Branches: MCTrack, TpcKalmanTrack, ZdcDigi, Vertex, MPDEvent, TOFMatching.
- Wagons: evCentrality, evPID.

Usage (add these lines to your "train" macro (e.g. 'RunAnalyses.C')):

```
MpdNuclei taskNuclei("taskNuclei","taskNuclei","NucleiAna.json");
man.AddTask(&taskNuclei);
```

Histograms naming scheme

Example: **hv__eff_pdg_primary_nhits_dca_tof**

- **hv** – histograms vector
- **eff** – "efficiency" histograms
- **pdg** – PID by MC
- **primary** – primary by MC
- **nhits** – with nhits cut
- **dca** – with dca cut
- **tof** – has ToF matchging

Each single histogram in this vector:
h__eff_pdg_primary_nhits_dca_tof_%s_centrality%d

- **h** – single histogram
- **%s** – particle name from the JSON configuration file ("p", "d", etc)
- **%d** – centrality bin number (0, 1, etc)

Histograms naming scheme

Example: **hv__pty_evpid**

- **hv** – histograms vector
- **pteta** – phase-space histograms "p_T *vs* y"
- **evpid** – PID by evPID wagon

Each single histogram in this vector: **h__pty_%s_evpid_centrality%d**

- **h** – single histogram
- **%s** – particle name from the JSON configuration file ("p", "d", etc)
- **%d** – centrality bin number (0, 1, etc)

Postprocessing

For the test purpose one can run the MpdRoot analysis train on the NICA cluster – in this case it would be a good idea to run tasks in parallel, e.g. 1000 parallel jobs, each job process 20000 events.

As the output one will have 1000 files with efficiency and phase-space histograms.

These histograms must be concatenated into the single file with the "hadd" program:

```
$ hadd final_file.root /some/directory/*.root
```

Postprocessing

Now, the final single file can be processed.

```
TH2D *hResult   = (TH2D*) inFile -> Get(Form("h__pteta_%s_evpid_centrality%d", pname, c_bin)) -> Clone("hResult");
TH2D *hEfficiency = nullptr;
TH2D *hNumerator = nullptr;
TH2D *hDenominator = nullptr;

hNumerator   = (TH2D*) inFile -> Get(Form("h__eff_pdg_primary_nhits_dca_%s_centrality%d", pname, c_bin)) -> Clone("hNumerator");
hDenominator = (TH2D*) inFile -> Get(Form("h__eff_pdg_primary_%s_centrality%d", pname, c_bin))           -> Clone("hDenominator");
hEfficiency = (TH2D*) hNumerator -> Clone("hEfficiency");
hEfficiency -> Divide(hNumerator, hDenominator);
hResult -> Divide(hEfficiency); // TPC efficiency

hNumerator   = (TH2D*) inFile -> Get(Form("h__eff_pdg_primary_nhits_dca_tof_%s_centrality%d", pname, c_bin)) -> Clone("hNumerator");
hDenominator = (TH2D*) inFile -> Get(Form("h__eff_pdg_primary_nhits_dca_%s_centrality%d", pname, c_bin))     -> Clone("hDenominator");
hEfficiency -> Divide(hNumerator, hDenominator);
hResult -> Divide(hEfficiency); // ToF efficiency

hNumerator   = (TH2D*) inFile -> Get(Form("h__eff_pdg_nhits_dca_tof_pid_%s_centrality%d", pname, c_bin)) -> Clone("hNumerator");
hDenominator = (TH2D*) inFile -> Get(Form("h__eff_pdg_nhits_dca_tof_%s_centrality%d", pname, c_bin))     -> Clone("hDenominator");
hEfficiency -> Divide(hNumerator, hDenominator);
hResult -> Divide(hEfficiency); // PID efficiency

hNumerator   = (TH2D*) inFile -> Get(Form("h__eff_primary_nhits_dca_tof_pid_%s_centrality%d", pname, c_bin)) -> Clone("hNumerator");
hDenominator = (TH2D*) inFile -> Get(Form("h__eff_nhits_dca_tof_pid_%s_centrality%d", pname, c_bin))         -> Clone("hDenominator");
hEfficiency -> Divide(hNumerator, hDenominator);
hResult -> Divide(hEfficiency); // DCA efficiency
```

PID

PID

The MpdRoot "trains"PID is based on the evPID wagon methods:

- float GetTPCNSigma(particle_type)
- float GetTOFNSigma(particle_type)

Both subroutines return the SD from the expected value in the number of σ.

In this wagon, particle can be identified by one of the two PID modes: **"All you can take"** and **"Competitive"**.

The PID mode can be defined in the configuration file – no need to recompile!

PID modes

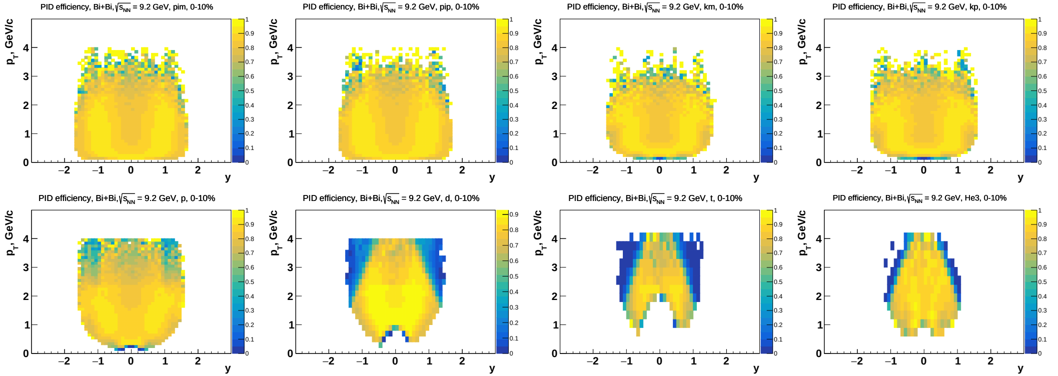
All you can take mode:

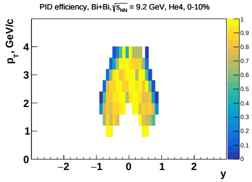
- GetTPCNSigma(particle_type) $< 2\sigma$.
- GetTOFNSigma(particle_type) $< 2\sigma$.
- The particle_type for TPC and TOF is the same.
- Several particles can be identified for the single track.

Competitive mode:

- GetTPCNSigma(particle_type) $< 2\sigma$.
- GetTOFNSigma(particle_type) $< 2\sigma$.
- **Both GetTPCNSigma and GetTOFNSigma are the least within all possible particles.**
- The particle_type for TPC and TOF is the same.
- Only one particle can be assigned to the track.

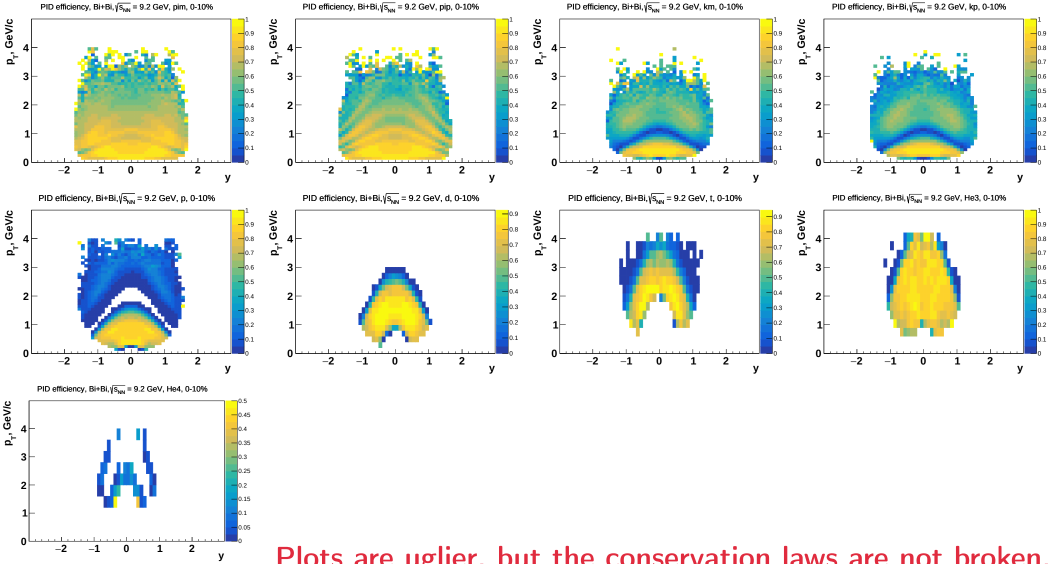
PID: All you can take





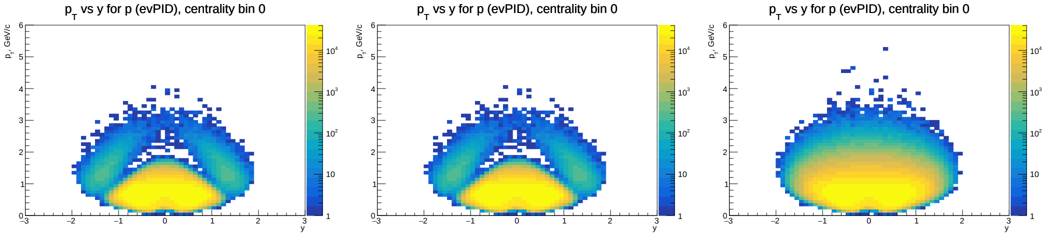
Nice plots, but is it really a good idea?

PID: Competitive



Plots are uglier, but the conservation laws are not broken.

PID: Competitive

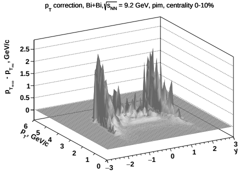


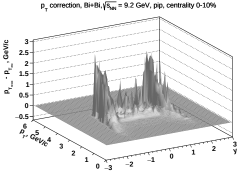
Protons phase-space vs particles included in the analysis, left to right:

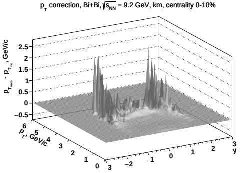
- π^-, π^+, K^-, K^+, p
- π^-, π^+, K^-, K^+, p, d, t, ^3He, ^4He
- p, d, t, ^3He, ^4He

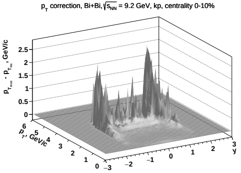
Corrections

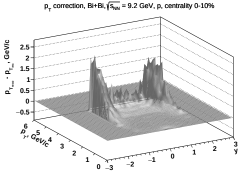
$p_{T_{reco}} - p_{T_{mc}}$

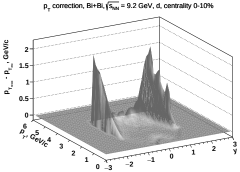

p_T correction, Bi+Bi, $\sqrt{s_{NN}}$ = 9.2 GeV, pim, centrality 0-10%
$p_{T_{reco}} - p_{T_{mc}}$, GeV/c
p_T, GeV/c
y

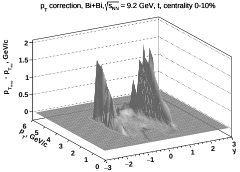

p_T correction, Bi+Bi, $\sqrt{s_{NN}}$ = 9.2 GeV, pip, centrality 0-10%
$p_{T_{reco}} - p_{T_{mc}}$, GeV/c
p_T, GeV/c
y

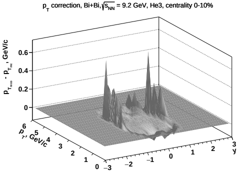

p_T correction, Bi+Bi, $\sqrt{s_{NN}}$ = 9.2 GeV, km, centrality 0-10%
$p_{T_{reco}} - p_{T_{mc}}$, GeV/c
p_T, GeV/c
y

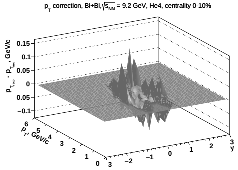

p_T correction, Bi+Bi, $\sqrt{s_{NN}}$ = 9.2 GeV, kp, centrality 0-10%
$p_{T_{reco}} - p_{T_{mc}}$, GeV/c
p_T, GeV/c
y


p_T correction, Bi+Bi, $\sqrt{s_{NN}}$ = 9.2 GeV, p, centrality 0-10%
$p_{T_{reco}} - p_{T_{mc}}$, GeV/c
p_T, GeV/c
y

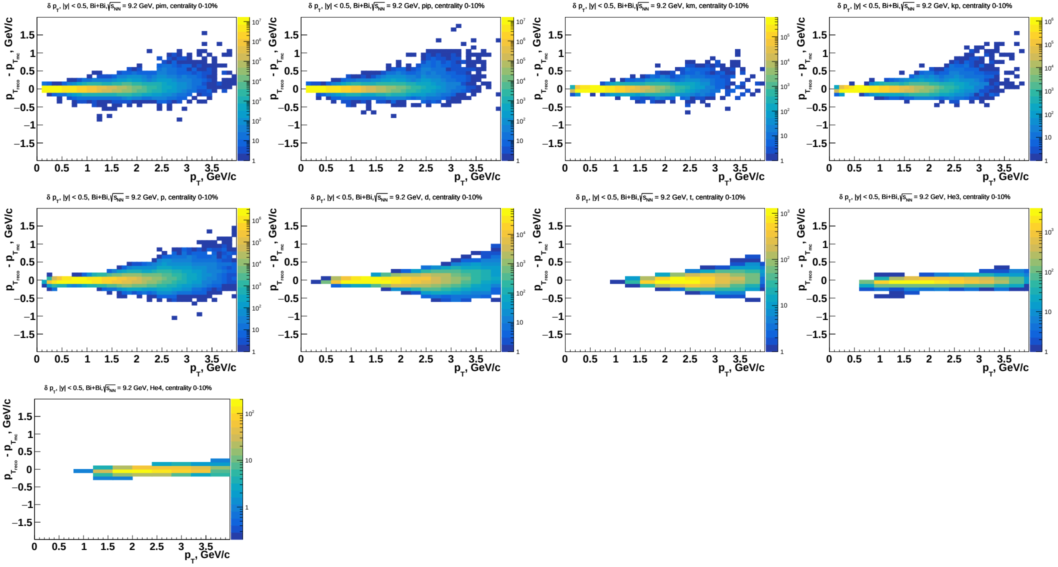

p_T correction, Bi+Bi, $\sqrt{s_{NN}}$ = 9.2 GeV, d, centrality 0-10%
$p_{T_{reco}} - p_{T_{mc}}$, GeV/c
p_T, GeV/c
y


p_T correction, Bi+Bi, $\sqrt{s_{NN}}$ = 9.2 GeV, t, centrality 0-10%
$p_{T_{reco}} - p_{T_{mc}}$, GeV/c
p_T, GeV/c
y

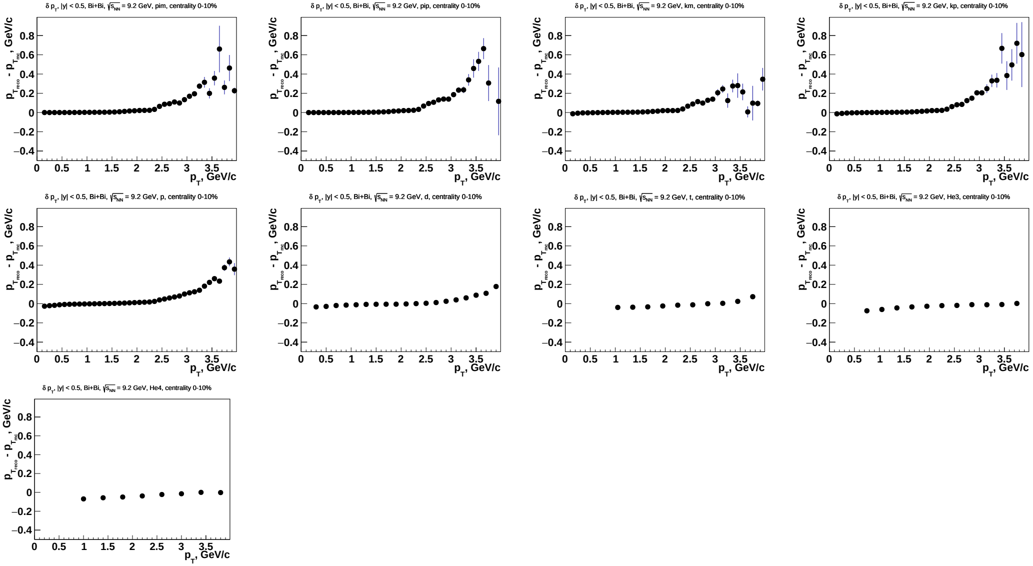

p_T correction, Bi+Bi, $\sqrt{s_{NN}}$ = 9.2 GeV, He3, centrality 0-10%
$p_{T_{reco}} - p_{T_{mc}}$, GeV/c
p_T, GeV/c
y


p_T correction, Bi+Bi, $\sqrt{s_{NN}}$ = 9.2 GeV, He4, centrality 0-10%
$p_{T_{reco}} - p_{T_{mc}}$, GeV/c
p_T, GeV/c
y

$p_{T_{reco}} - p_{T_{mc}}$ within $|y| < 0.5$



$p_{T_{reco}} - p_{T_{mc}}$ within $|y| < 0.5$



Reconstructed p_T correction

The reconstructed p_T is taken after the nhits and dca cuts if the track has the ToF matching and PID by wagon is equal to the PID by MC.

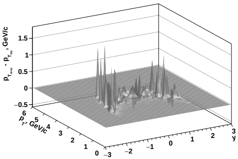
Correction is extracted from the TProfile2D:
`correction[particle] = p__corr_pt[particle] -> Interpolate(y, pt)`
The bilinear interpolation is based on the four nearest bin centers.

Corrected p_T:
`pt -= correction[particle]`

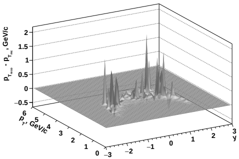
Rapidity is recalculated with the corrected p_T.

No need to make parametrizations for each particle or in case some tracking/GEANT/etc update.

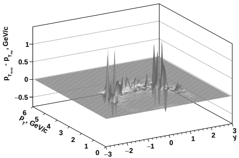
$p_{T_{reco}} - p_{T_{mc}}$ after correction



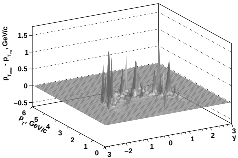
p_T correction, Bi+Bi, $\sqrt{s_{NN}}$ = 9.2 GeV, pim, centrality 0-10%



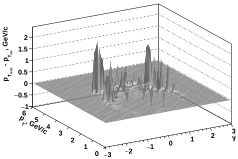
p_T correction, Bi+Bi, $\sqrt{s_{NN}}$ = 9.2 GeV, pip, centrality 0-10%



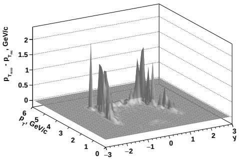
p_T correction, Bi+Bi, $\sqrt{s_{NN}}$ = 9.2 GeV, km, centrality 0-10%



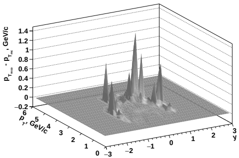
p_T correction, Bi+Bi, $\sqrt{s_{NN}}$ = 9.2 GeV, kp, centrality 0-10%



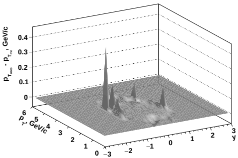
p_T correction, Bi+Bi, $\sqrt{s_{NN}}$ = 9.2 GeV, p, centrality 0-10%



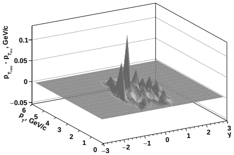
p_T correction, Bi+Bi, $\sqrt{s_{NN}}$ = 9.2 GeV, d, centrality 0-10%



p_T correction, Bi+Bi, $\sqrt{s_{NN}}$ = 9.2 GeV, t, centrality 0-10%

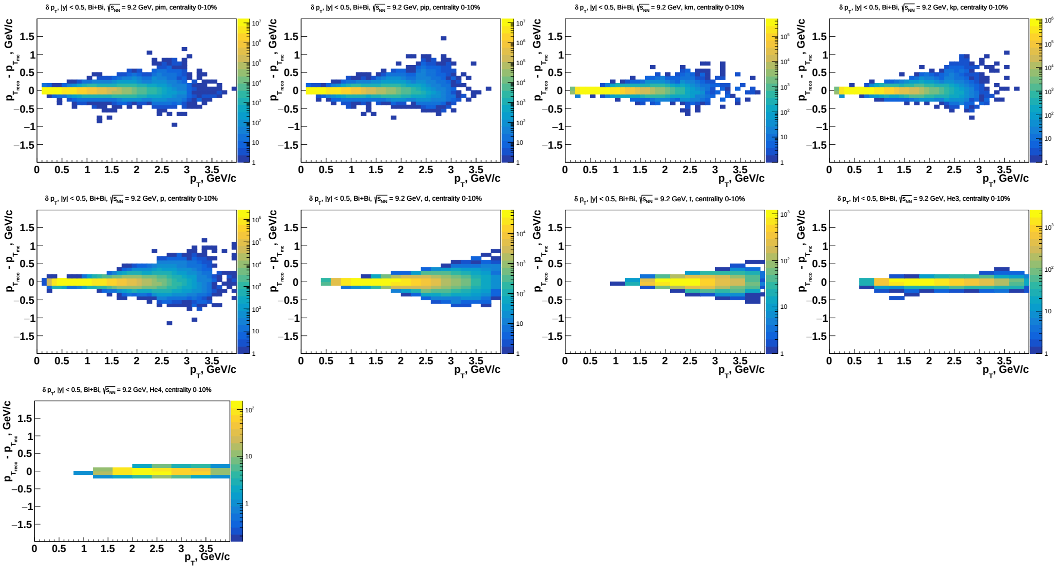


p_T correction, Bi+Bi, $\sqrt{s_{NN}}$ = 9.2 GeV, He3, centrality 0-10%

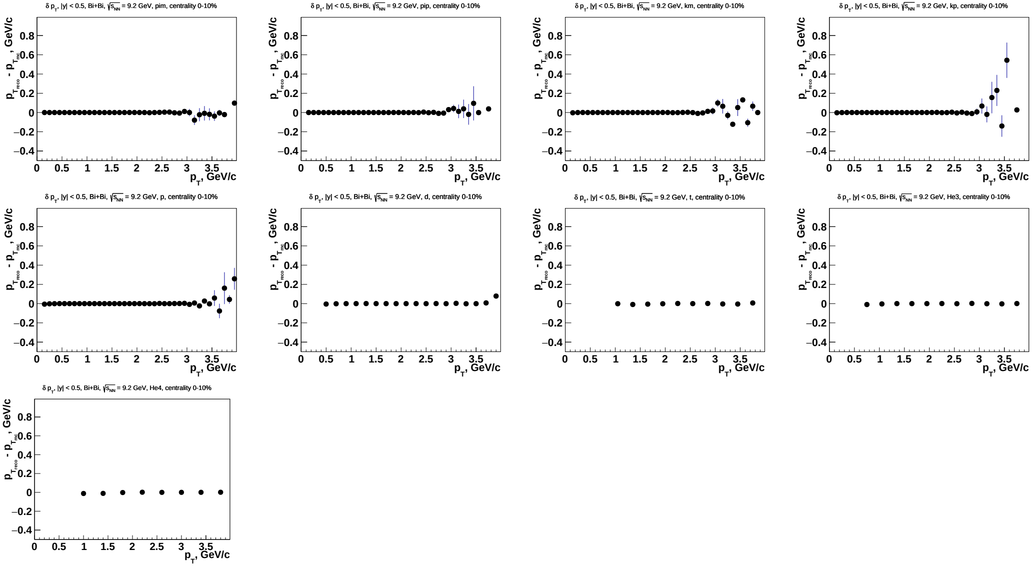


p_T correction, Bi+Bi, $\sqrt{s_{NN}}$ = 9.2 GeV, He4, centrality 0-10%

$p_{T_{reco}} - p_{T_{mc}}$ within $|y| < 0.5$ after correction

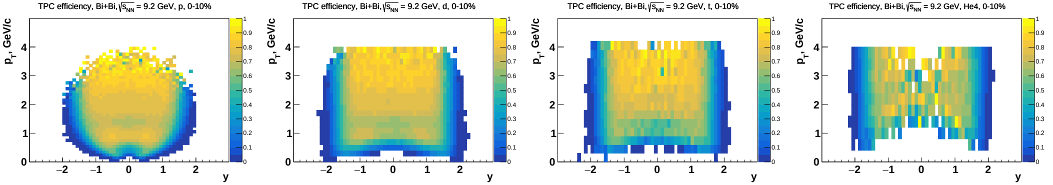


$p_{T_{reco}} - p_{T_{mc}}$ within $|y| < 0.5$ after correction



Efficiencies

TPC efficiency



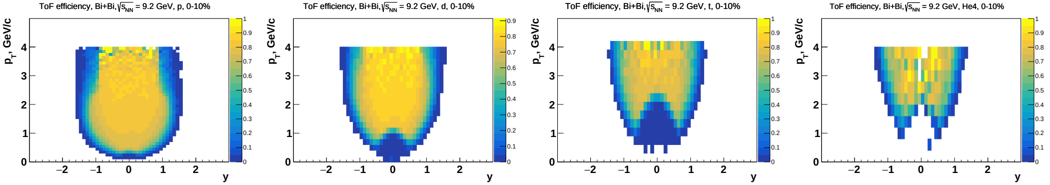
$$TPC\ efficiency = \frac{hv\_\_eff\_pdg\_primary\_nhits\_dca}{hv\_\_eff\_pdg\_primary}$$

hv__eff_pdg_primary_nhits_dca – PID by MC, primary by MC, with nhits cut, with dca cut. The low p_T cut is also here. Reco loop, associated MC tracks info.

hv__eff_pdg_primary – PID by MC, primary by MC. There is no low p_T cut. MC loop.

Courtesy to A. Mudrokh

ToF efficiency



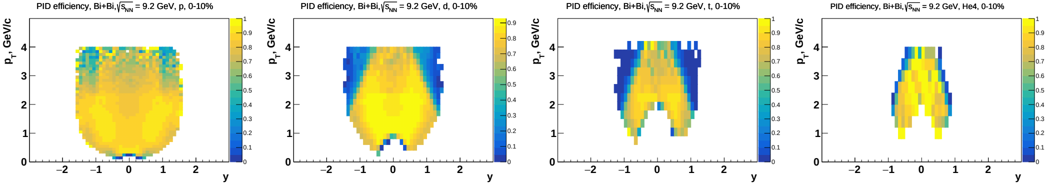
$$ToF\ efficiency = \frac{hv\_\_eff\_pdg\_primary\_nhits\_dca\_tof}{hv\_\_eff\_pdg\_primary\_nhits\_dca}$$

hv__eff_pdg_primary_nhits_dca_tof – PID by MC, primary by MC, with nhits cut, with dca cut, has ToF matching. Reco loop, associated MC tracks info.

hv__eff_pdg_primary_nhits_dca – PID by MC, primary by MC, with nhits cut, with dca cut. Reco loop, associated MC tracks info.

Courtesy to A. Mudrokh

PID efficiency

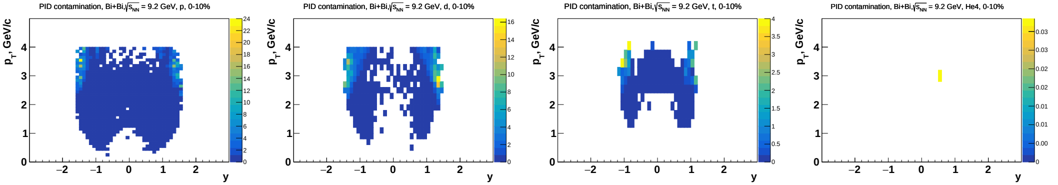


$$PID\ efficiency = \frac{hv\_\_eff\_pdg\_nhits\_dca\_tof\_pid}{hv\_\_eff\_pdg\_nhits\_dca\_tof}$$

hv__eff_pdg_nhits_dca_tof_pid – with nhits cut, with dca cut, has ToF matching, PID by wagon = PID by MC. Reco loop, associated MC tracks info.
hv__eff_pdg_nhits_dca_tof – PID by MC, with nhits cut, with dca cut, has ToF matching. Reco loop, associated MC tracks info.

Courtesy to A. Mudrokh

PID contamination

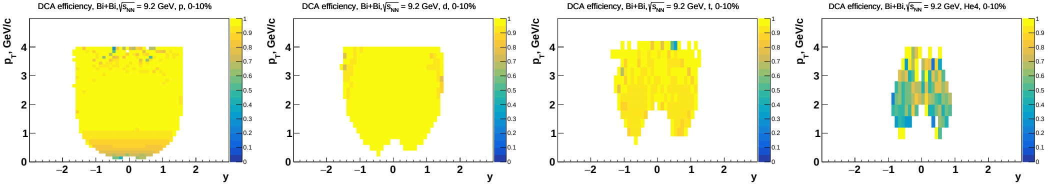


$$PID\ contamination = \frac{hv\_\_eff\_pdg\_nhits\_dca\_tof\_wpid}{hv\_\_eff\_pdg\_nhits\_dca\_tof\_pid}$$

hv__eff_pdg_nhits_dca_tof_wpid – PID by MC, with nhits cut, with dca cut, has ToF matching, PID by wagon $\neq$ PID by MC. Reco loop, associated MC tracks info.
hv__eff_pdg_nhits_dca_tof_pid – PID by MC, with nhits cut, with dca cut, has ToF matching, PID by wagon = PID by MC. Reco loop, associated MC tracks info.

Courtesy to A. Mudrokh

DCA efficiency



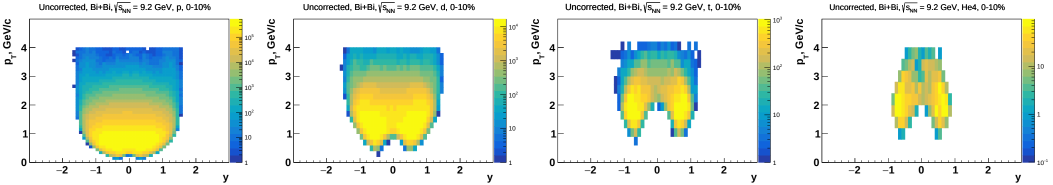
$$DCA\ efficiency = \frac{hv\_\_eff\_primary\_nhits\_dca\_tof\_pid}{hv\_\_eff\_nhits\_dca\_tof\_pid}$$

hv__eff_primary_nhits_dca_tof_pid – PID by wagon, primary by MC, with nhits cut, with dca cut, has ToF matching. Reco loop, associated MC tracks info.
hv__eff_nhits_dca_tof_pid – PID by wagon, with nhits cut, with dca cut, has ToF matching. Reco loop.

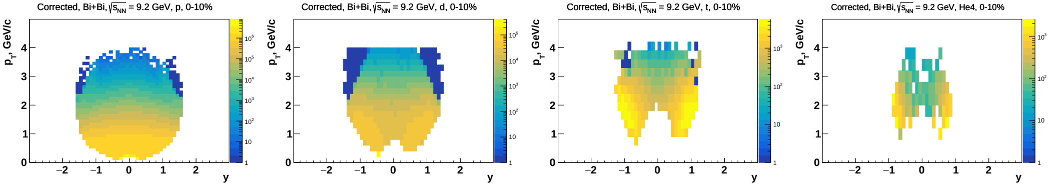
Courtesy to A. Mudrokh

Results

Uncorrected results

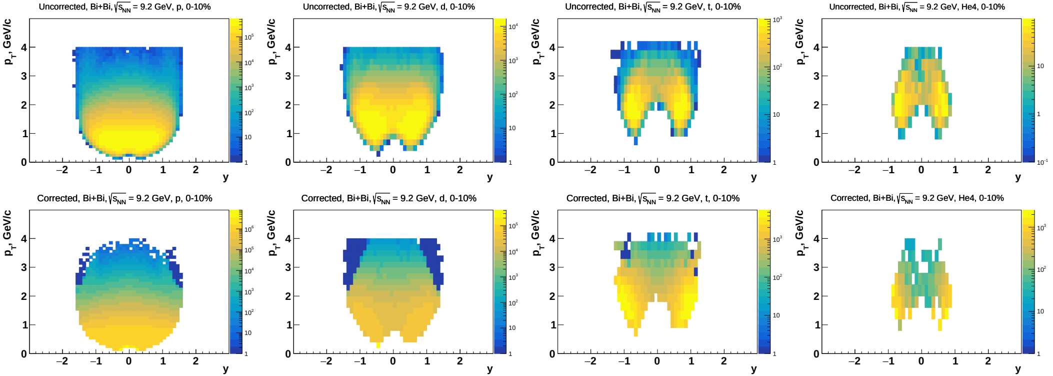


Corrected results

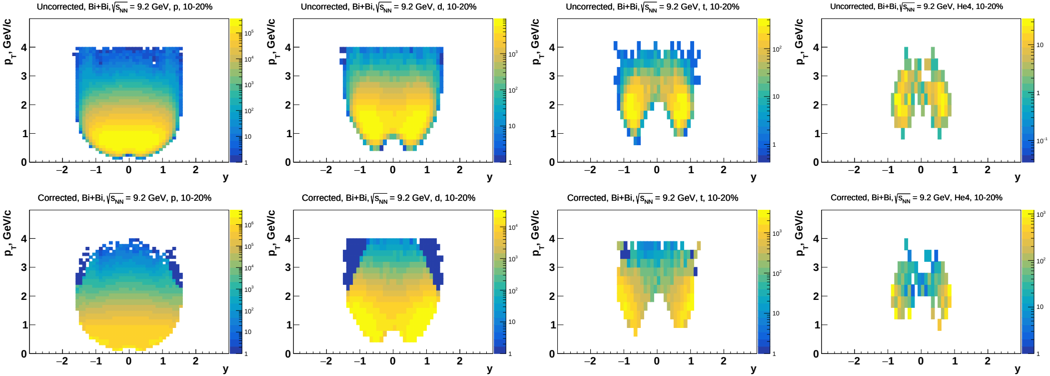


$$Result = \frac{Uncorrected \cdot (1 - PID\ \ contamination)}{TPC\ \ efficiency \cdot ToF\ \ efficiency \cdot PID\ \ efficiency \cdot DCA\ \ efficiency}$$

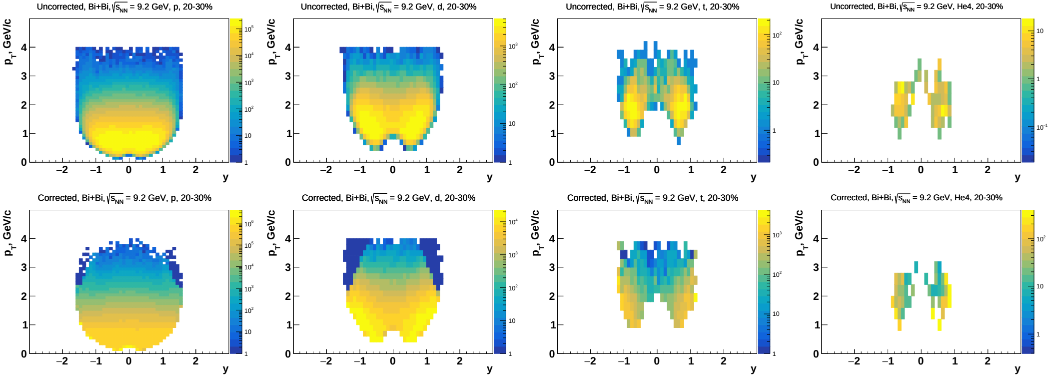
Uncorrected and corrected results: 0-10%



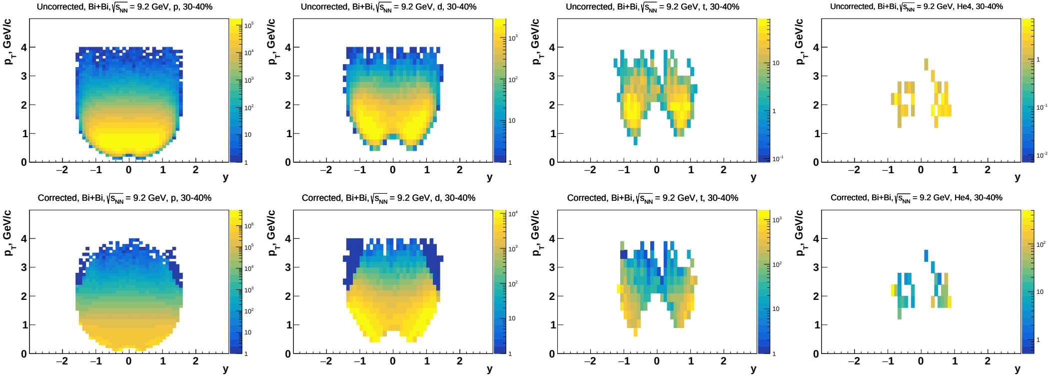
Uncorrected and corrected results: 10-20%



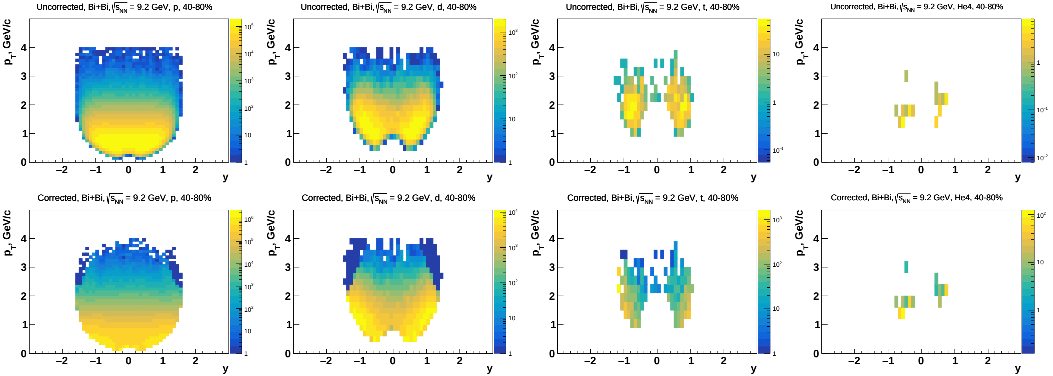
Uncorrected and corrected results: 20-30%



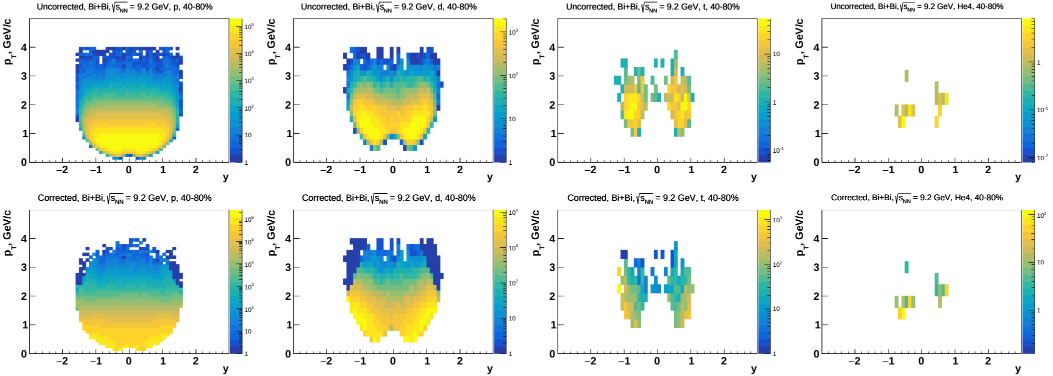
Uncorrected and corrected results: 30-40%



Uncorrected and corrected results: 40-80%



Uncorrected and corrected results: 40-80%



Documentation

The new MpdNuclei ”wagon” is using the specially-formatted comments within the code processed by the documentation generator Doxygen.
Current documentation status:

- All output histograms are described.
- All possible settings are documented.
- All subroutines are described.
- A "Readme"with the common wagon description is provided with the source code.
- The post-processing macro and its subroutines are documented. All plots within this presentation are made with this single post-processing macro.

```
/** This subroutine performs the track quality checks.

    Return values are:
    - true (1) if checks are not passed ("bad" track -- skip);
    - false (0) if checks are passed ("good" track -- analyze);

  The track can be skipped if:
    - The number of hits is less than MpdNuclei::s__tr_NHits
    - The n-sigma for DCAx is larger than MpdNuclei::s__tr_NSigmaDCAx
    - The n-sigma for DCAy is larger than MpdNuclei::s__tr_NSigmaDCAy
    - The n-sigma for DCAz is larger than MpdNuclei::s__tr_NSigmaDCAz
    - The transverse momentum is less than MpdNuclei::s__tr_LowPtCut
  \param track MpdTrack to analyse
*/
bool MpdNuclei::bad_track(MpdTrack* track){
  if (track->GetNofHits() <= s__tr_NHits)              return true; // Number of hits cut
  if (fabs(track->GetNSigmaDCAx()) > s__tr_NSigmaDCAx) return true; // |DCAx| cut
  if (fabs(track->GetNSigmaDCAy()) > s__tr_NSigmaDCAy) return true; // |DCAy| cut
  if (fabs(track->GetNSigmaDCAz()) > s__tr_NSigmaDCAz) return true; // |DCAz| cut
  if (fabs(track->GetPt()) <= s__tr_LowPtCut)          return true; // Low transverse momentum cut
  return false;
}
```

◆ bad_track()

bool MpdNuclei::bad_track (MpdTrack * track) private

This subroutine performs the track quality checks.

Return values are:

- true (1) if checks are not passed ("bad" track – skip);
- false (0) if checks are passed ("good" track – analyze);

The track can be skipped if:

- The number of hits is less than **MpdNuclei::s__tr_NHits**
- The n-sigma for DCAx is larger than **MpdNuclei::s__tr_NSigmaDCAx**
- The n-sigma for DCAy is larger than **MpdNuclei::s__tr_NSigmaDCAy**
- The n-sigma for DCAz is larger than **MpdNuclei::s__tr_NSigmaDCAz**
- The transverse momentum is less than **MpdNuclei::s__tr_LowPtCut**

Parameters

track MpdTrack to analyse

Definition at line **395** of file **MpdNuclei.cxx**.

Summary

The first version of the "Nuclei" wagon is presented:

- The wagon uses the JSON-formatted input file to handle all possible settings and automatically create histograms for the defined particles.
- Only phase-space histograms (p_T *vs* y) are included.
- The reconstructed p_T correction procedure is implemented.
- Different efficiencies are calculated within same phase-space bins:
  - TPC efficiency
  - ToF efficiency
  - PID efficiency
  - DCA efficiency
  - PID contamination
- TPC, ToF, PID, DCA efficiencies and PID contamination are used for the final results corrections.
- The Doxygen-style documentation for the wagon is provided.

Current **proposals**:

- Remove dE/dx histograms.
- Push the "nucleï" wagon into the "dev" version of the MpdRoot.
- Revise the definitions of efficiencies.
- ~~Move from the "p_T/η" phase-space to the "p_T/y" for the final results.~~
- Merge the bulk spectra and nuclei wagons into one?

Thank you for your attention!

This presentation was prepared using LaTeX with the Beamer package on Overleaf.