Updates of a new analysis framework on SHOE

Giacomo Ubaldi
Roberto Zarrella

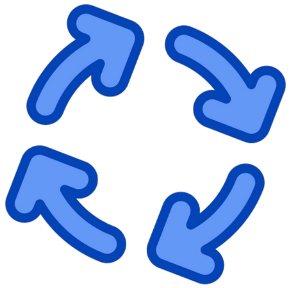
XVII FOOT Collaboration Meeting

17/12/2024

Analysis workflow

- **Action** based structure
- Every class has methods to be applied *before*, *during* and *after* the event loop.

Event Loop( )



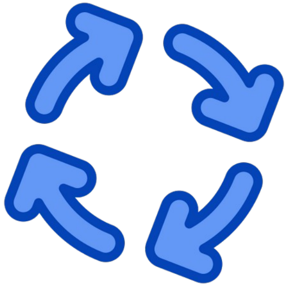
1

Create a container of **selection cuts**
for events and global tracks (**reco** level)

Analysis workflow

- **Action** based structure
- Every class has methods to be applied *before*, *during* and *after* the event loop.

Event Loop( )



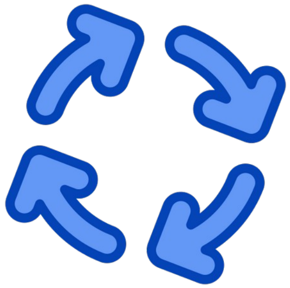
1 Create a container of **selection cuts** for events and global tracks (**reco** level)

2 Create a container of **physical quantities counts** needed for cross section (**reco** level)

Analysis workflow

- **Action** based structure
- Every class has methods to be applied *before*, *during* and *after* the event loop.

Event Loop( )



1 Create a container of **selection cuts** for events and global tracks (**reco** level)

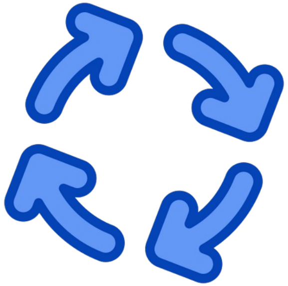
2 Create a container of **physical quantities counts** needed for cross section (**reco** level)

3 Create a container of **selection cuts** and **quantities counts** (**MC truth** level)

Analysis workflow

- **Action** based structure
- Every class has methods to be applied *before*, *during* and *after* the event loop.

Event Loop( )

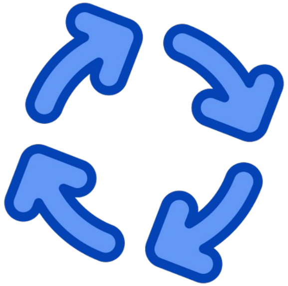


1. Create a container of **selection cuts** for events and global tracks (**reco** level)
2. Create a container of **physical quantities counts** needed for cross section (**reco** level)
3. Create a container of **selection cuts** and **quantities counts** (**MC truth** level)
4. Fill **Flat Trees** with the previous containers for every event / global track

Analysis workflow

- **Action** based structure
- Every class has methods to be applied *before*, *during* and *after* the event loop.

Event Loop( )

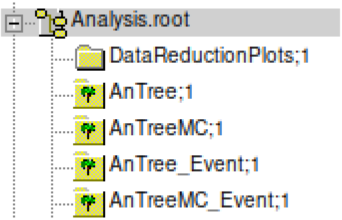


1 Create a container of **selection cuts** for events and global tracks (**reco** level)

2 Create a container of **physical quantities counts** needed for cross section (**reco** level)

3 Create a container of **selection cuts** and **quantities counts** (**MC truth** level)

4 Fill **Flat Trees** with the previous containers for every event / global track



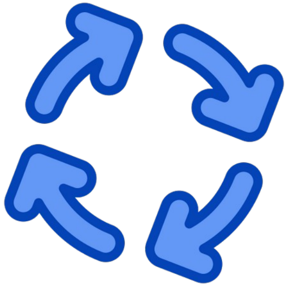
- For **every event** (MC and Reco):
  - event selection cuts
- For **every global track** (MC and Reco):
  - track selection cuts
  - physical quantities: **charge, angle, velocity**

for more technical details see Software Meeting 18/06/2024
https://agenda.infn.it/event/42238/

Analysis workflow

- **Action** based structure
- Every class has methods to be applied *before*, *during* and *after* the event loop.

Event Loop( )



After Loop( )

1. Create a container of **selection cuts** for events and global tracks (**reco** level)
2. Create a container of **physical quantities counts** needed for cross section (**reco** level)
3. Create a container of **selection cuts** and **quantities counts** (**MC truth** level)
4. Fill **Flat Trees** with the previous containers for every event / global track
5. Filter, process data to obtain a final **cross section**

Parameter Configuration

```
// -+-+-+-+-+--+-+-+-+-+--+-+-+-+-+--+-+-+-+-+--+-+-+-+-+--+-+-+-+-+--+-+-+-+-+-
// Parameters for Analysis
// -+-+-+-+-+--+-+-+-+-+--+-+-+-+-+--+-+-+-+-+--+-+-+-+-+--+-+-+-+-+--+-+-+-+-+-
MassReso:          0

PtReso:            0

DataReduction:     1    1 2 3 4

CrossSection:      1    5
```

CrossSection ***after*** *DataReduction*

- With **DataReduction** on: trees with *variables* and *cuts*
- With **CrossSection** on: plots of XS
- **Two different outputs**

```
// -+-+-+-+-+--+-+-+-+-+--+-+-+-+-+--+-+-+-+-+--+-+-+-+-+--+-+-+-+-+--+-+-+-+-+-
// Parameters for Analysis
// -+-+-+-+-+--+-+-+-+-+--+-+-+-+-+--+-+-+-+-+--+-+-+-+-+--+-+-+-+-+--+-+-+-+-+-
MassReso:          0

PtReso:            0

DataReduction:     0

CrossSection:      1    5
```

CrossSection ***stand alone***

- With **CrossSection** on: plots of XS
- **DataReduction as input; XS plots as output**

RDataFrae

5 TANAactCrossSection

***After Loop( )*:**

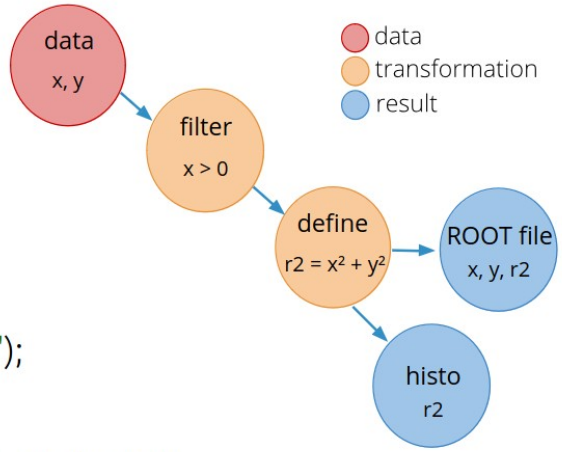
- Root TTree handled by the class **ROOT::RDataFrame**

```
//----------------------------------- Load ROOT DataFrame
ROOT::RDataFrame d(*fAnTree);
ROOT::RDataFrame d_MC(*fAnTreeMC);
ROOT::RDataFrame d_Ev(*fAnTree_Event);
ROOT::RDataFrame d_MC_Ev(*fAnTreeMC_Event);
```

- used to manipulate **HEP data**
- consistent interfaces in **Python** and **C++**
- already implemented and tested methods for **data analysis** (filters, selections...)

```
ROOT::RDataFrame df(dataset);
auto df2 = df.Filter("x > 0")
             .Define("r2", "x*x + y*y");
auto rHist = df2.Histo1D("r2");
df2.Snapshot("newtree", "newfile.root");
```

Write datasets to disk, also in parallel.



Cross Section steps

5

TANAactCrossSection

***After Loop( )*:**

$$\frac{d\sigma}{d\theta}(Z,\theta) = \frac{Y(Z,\theta)}{N_{beam}\, N_{target}\, \Omega_\theta\, \epsilon(Z,\theta)}$$

- Applying **selection cuts**

```
//--- Selection Cuts
aEventCutsMap["BMcut"] = 1;               // only 1 track in BM
aEventCutsMap["VTXposCut"] = 1;           // no VTX points pileup
aTrackCutsMap["HasTwPoint"] = 1;          // track has TW point
aTrackCutsMap["TrackQuality"] = 1;        // inside Chi2 values
aTrackCutsMap["VTXposCut"] = 1;           // VTX point inside TG geometry
                                          // VT track matches with BM track
```

Cross Section steps

5 **TANAactCrossSection**

$$\frac{d\sigma}{d\theta}(Z,\theta) = \frac{Y(Z,\theta)}{N_{beam}\ N_{target}\ \Omega_\theta\ \epsilon(Z,\theta)}$$

After Loop( ):

- Applying **selection cuts**

```
//--- Selection Cuts
aEventCutsMap["BMcut"] = 1;                  // only 1 track in BM
aEventCutsMap["VTXposCut"] = 1;              // no VTX points pileup
aTrackCutsMap["HasTwPoint"] = 1;             // track has TW point
aTrackCutsMap["TrackQuality"] = 1;           // inside Chi2 values
aTrackCutsMap["VTXposCut"] = 1;              // VTX point inside TG geometry
                                             // VT track matches with BM track
```

- Calculate the **yields** (filtering the previous selection cuts)

```
//--- Yield
aVariablesList.push_back("Charge");
aVariablesList.push_back("Theta");
FillYield(d, aEventCutsMap, aTrackCutsMap, aVariablesList, "1_Reco_Charge");
```

Cross Section steps

5 **TANAactCrossSection**

***After Loop( )*:**

$$\frac{d\sigma}{d\theta}(Z,\theta) = \frac{Y(Z,\theta)}{N_{beam}\, N_{target}\, \Omega_\theta\, \epsilon(Z,\theta)}$$

- Applying **selection cuts**

```
//--- Selection Cuts
aEventCutsMap["BMcut"] = 1;              // only 1 track in BM
aEventCutsMap["VTXposCut"] = 1;          // no VTX points pileup
aTrackCutsMap["HasTwPoint"] = 1;         // track has TW point
aTrackCutsMap["TrackQuality"] = 1;       // inside Chi2 values
aTrackCutsMap["VTXposCut"] = 1;          // VTX point inside TG geometry
                                         // VT track matches with BM track
```

- Calculate the **yields** (filtering the previous selection cuts)

```
//--- Yield
aVariablesList.push_back("Charge");
aVariablesList.push_back("Theta");
FillYield(d, aEventCutsMap, aTrackCutsMap, aVariablesList, "1_Reco_Charge");
```

- Calculate the **efficiency**

```
//--- Efficiency
aEffPathList.push_back("1_Reco_MC_Theta");
aEffPathList.push_back("0_MC_Ref_Theta");
ComputeEfficiencies(aEffPathList, aVariablesList);
```

Cross Section steps

5 **TANAactCrossSection**

***After Loop( )*:**

$$\frac{d\sigma}{d\theta}(Z,\theta) = \frac{Y(Z,\theta)}{N_{beam}\ N_{target}\ \Omega_\theta\ \epsilon(Z,\theta)}$$

- Applying **selection cuts**

```
//--- Selection Cuts
aEventCutsMap["BMcut"] = 1;                  // only 1 track in BM
aEventCutsMap["VTXposCut"] = 1;              // no VTX points pileup
aTrackCutsMap["HasTwPoint"] = 1;             // track has TW point
aTrackCutsMap["TrackQuality"] = 1;           // inside Chi2 values
aTrackCutsMap["VTXposCut"] = 1;              // VTX point inside TG geometry
                                             // VT track matches with BM track
```

- Calculate the **yields** (filtering the previous selection cuts)

```
//--- Yield
aVariablesList.push_back("Charge");
aVariablesList.push_back("Theta");
FillYield(d, aEventCutsMap, aTrackCutsMap, aVariablesList, "1_Reco_Charge");
```

- Calculate the **efficiency**

```
//--- Efficiency
aEffPathList.push_back("1_Reco_MC_Theta");
aEffPathList.push_back("0_MC_Ref_Theta");
ComputeEfficiencies(aEffPathList, aVariablesList);
```

- Calculate the **luminosity**

```
//--- Luminosity
ComputeLuminosity(d_Ev, aEventCutsMap, "1_Reco_Charge");
```

Cross Section steps

5

TANAactCrossSection

***After Loop( )*:**

$$\frac{d\sigma}{d\theta}(Z,\theta) = \frac{Y(Z,\theta)}{N_{beam}\, N_{target}\, \Omega_\theta\, \epsilon(Z,\theta)}$$

- Applying **selection cuts**

```
//--- Selection Cuts
aEventCutsMap["BMcut"] = 1;              // only 1 track in BM
aEventCutsMap["VTXposCut"] = 1;          // no VTX points pileup
aTrackCutsMap["HasTwPoint"] = 1;         // track has TW point
aTrackCutsMap["TrackQuality"] = 1;       // inside Chi2 values
aTrackCutsMap["VTXposCut"] = 1;          // VTX point inside TG geometry
                                         // VT track matches with BM track
```

- Calculate the **yields** (filtering the previous selection cuts)

```
//--- Yield
aVariablesList.push_back("Charge");
aVariablesList.push_back("Theta");
FillYield(d, aEventCutsMap, aTrackCutsMap, aVariablesList, "1_Reco_Charge");
```

- Calculate the **efficiency**

```
//--- Efficiency
aEffPathList.push_back("1_Reco_MC_Theta");
aEffPathList.push_back("0_MC_Ref_Theta");
ComputeEfficiencies(aEffPathList, aVariablesList);
```

- Calculate the **luminosity**

```
//--- Luminosity
ComputeLuminosity(d_Ev, aEventCutsMap, "1_Reco_Charge");
```

- Calculate the **Cross Section**

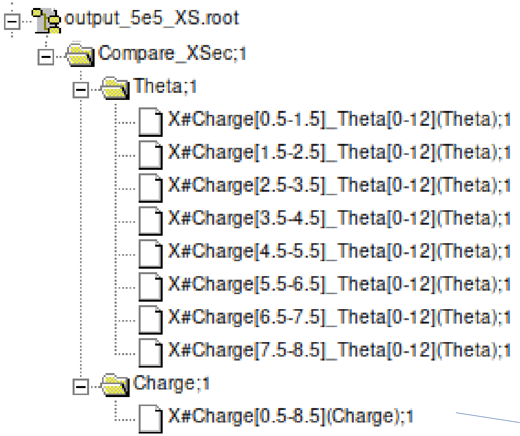
```
//--- Cross Section
ComputeXSec(aVariablesList,      "1_Reco_Charge",       // yield
                                 "1_Reco_MC_Charge",    // efficiency
                                 "",                    // purity
                                 "1_Reco_Charge");      // luminosity
```

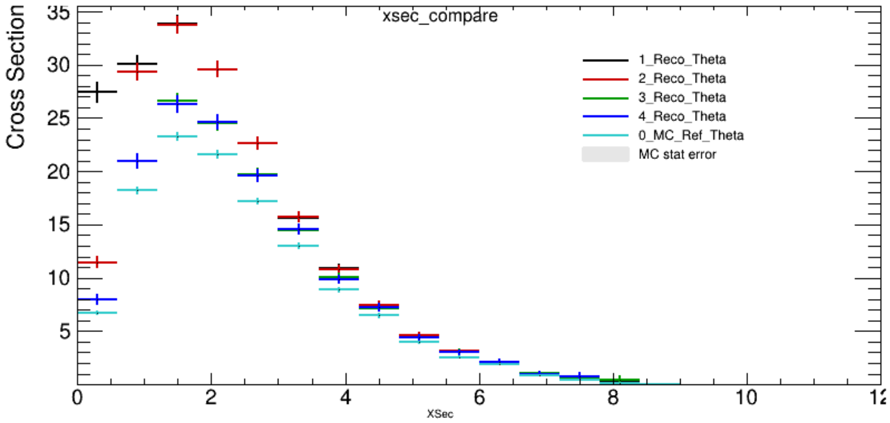
Results

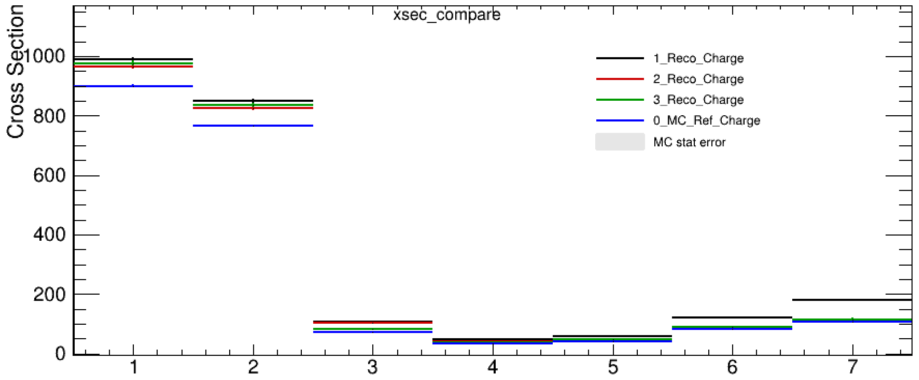
5 TANAactCrossSection

***After Loop( )*:**

- Save plots in **subfolders** according to the variable:







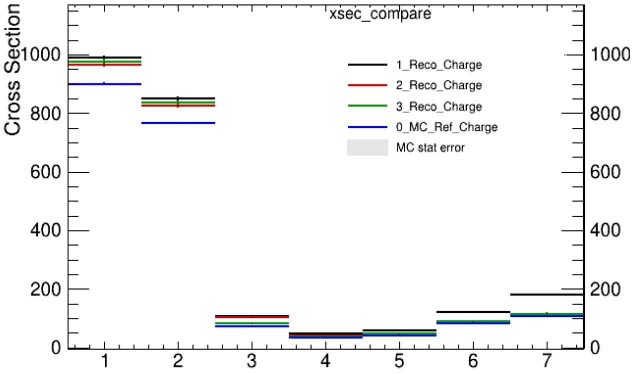
Analysis Comparisons

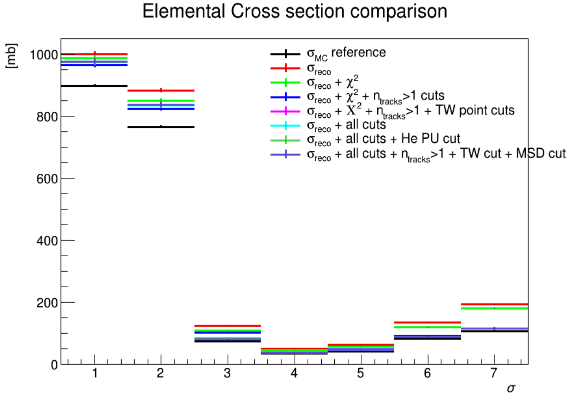
example: *GSI2021_PS_MC dataset*

Analysis (2024)

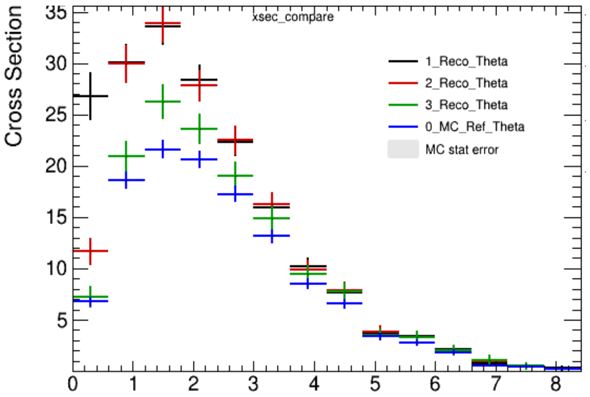
Analysis (2023)

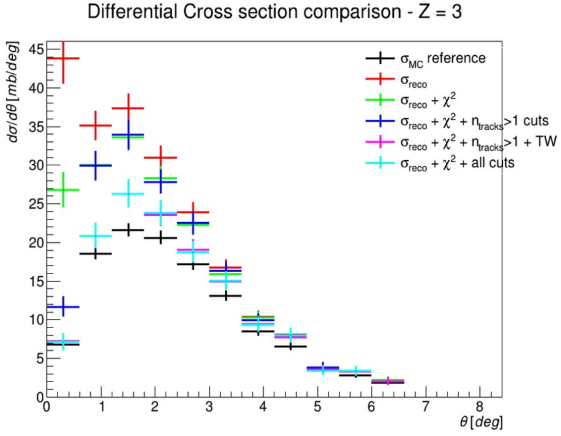
elemental XS





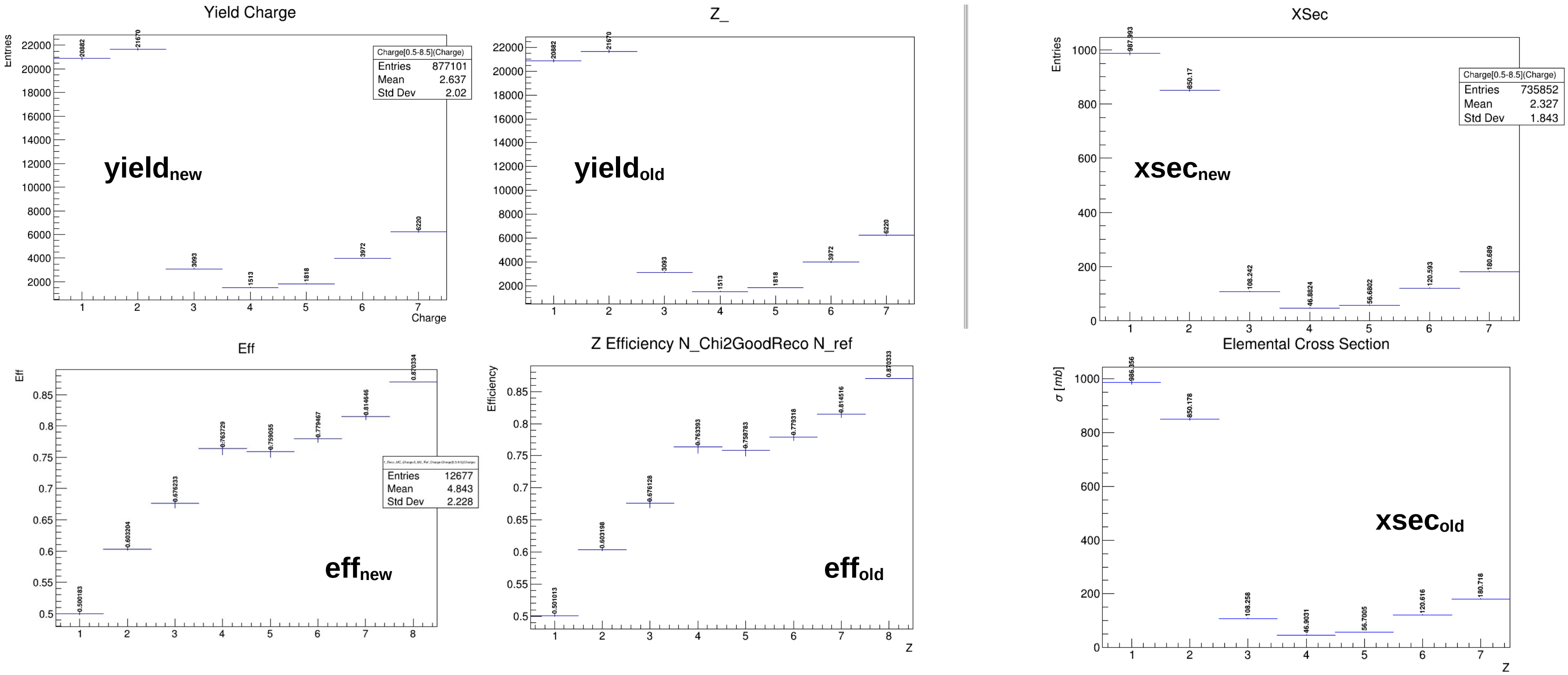
angular differential XS of Z=3





Results

- New Analysis (2024) vs "old" one (2023)
  example: *elemental cross section (Z) GSI2021_PS_MC*



Conclusions and future perspectives

- All the **steps of analysis** cross section measurements introduced

- **New analysis** organized in classes and actions: 1 big class vs more specific classes
- Much **less time consuming:** ~ dozen of hours vs ~ few hours
- CONDOR script to be launched by TIER1
- Everything is **done in SHOE** (before SHOE + external machinery)

- **Closure test** comparison with the previous analysis framework **done succesfully**

- Refine details like paths and parameter files (f.e. parameters binning, “standard” cuts…)

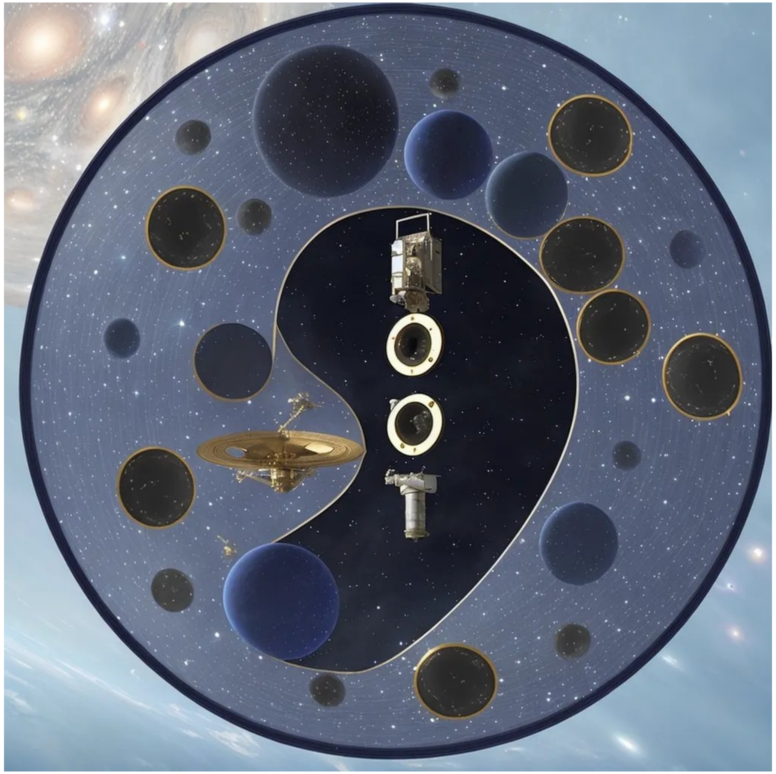
- Compare GSI21_MC (GM Trento 2023) vs GSI21_PS_MC

- When everything is fixed, the framework could be easily applied to **all the data takings**
  → apply **closure tests** to all the campains
  (HIT22, CNAO23, CNAO24 ...)

Thank you for the attention!

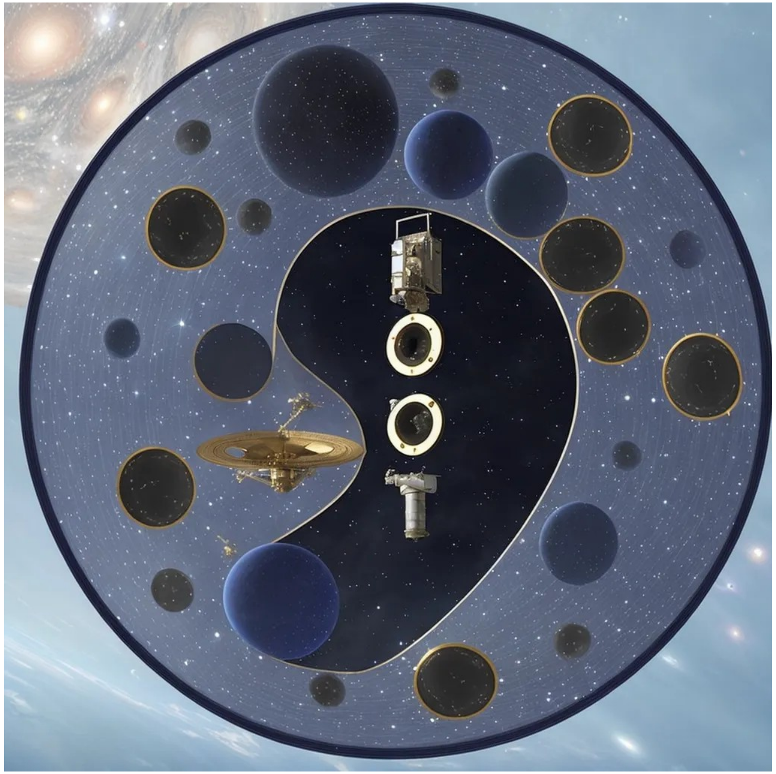
https://huggingface.co/spaces/AP123/IllusionDiffusion

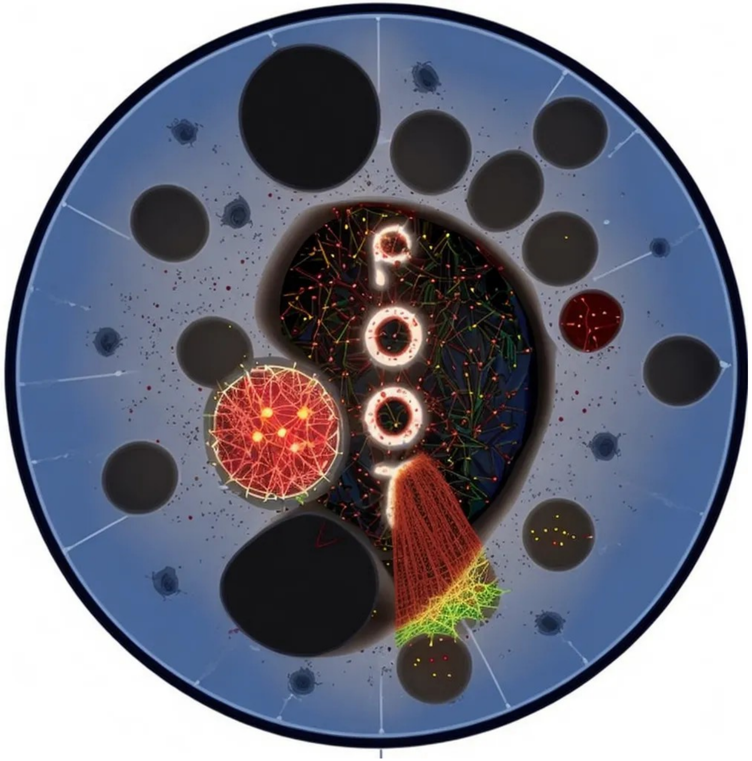
Thank you for the attention!



https://huggingface.co/spaces/AP123/IllusionDiffusion

Thank you for the attention!





https://huggingface.co/spaces/AP123/IllusionDiffusion

Back up slides

Selection Cuts

1

TANAactNtuSelectionCuts

In Loop( ):

- For every reconstructed track, two **cuts containers** are associated, for *events* and for *tracks*
- According to every cut, a value of 1,0 or exceptions is associated

```
 Loaded Event: 1
Event cuts
[BMcut] = 1;
[NTracksCut] = 0;
[SCcut] = -99;
[TWnum] = 1;

Track cuts
Element 0
[TWclone] = 0;
[TrackQuality] = 1;
[VTXposCut] = 1;
```

key	**description**	**values**
`SCcut`	There is NOT pileup in the SC and the energy release is higher than the one of a primary (> .005 GeV)	**1**: the condition is hold **0** : the condition is not verified **-99** : some errors expect
`BMcut`	Only one track crosses the BM detector	**1**: the condition is hold **0** : the condition is not verified **-99** : some errors expect
`NTracksCut`	The number of reconstructed tracks should be higher than 1	**1**: the condition is hold **0** : the condition is not verified
`TWnum`	The number of reconstructed tracks is the same of the reconstructed TW points	**1**: the condition is hold **0** : the condition is not verified **-99** : some errors expect

Selection Cuts

1 **TANAactNtuSelectionCuts**

In Loop( ):

- For every reconstructed track, two **cuts containers** are associated, for *events* and for *tracks*
- According to every cut, a value of 1,0 or exceptions is associated

```
 Loaded Event: 1
Event cuts
[BMcut] = 1;
[NTracksCut] = 0;
[SCcut] = -99;
[TWnum] = 1;

Track cuts
Element 0
[TWclone] = 0;
[TrackQuality] = 1;
[VTXposCut] = 1;
```

key	description	values
`TWclone`	The specific track has the same TWpoint of *at least* another track	**1**: the condition is hold **0** : the condition is not verified **-99** : the track has **not** TW point
`TrackQuality`	The specific track has a residual < 0.01 and a p-value > 0.01	**1**: the condition is hold **0** : the condition is not verified
`VTXposCut`	The reconstructed VTX point is positioned within the target dimensions	**1**: the condition is hold **0** : the condition is not verified **-99** : some errors expect

Reco quantities

2

TANAactNtuGlbTrackCounts

In Loop( ):

For every track, the main quantities are recorded in containers,both via *reconstruction* and via *Monte Carlo*

- The present variables are now: ***charge***, ***angle***, ***beta***

```
MC tracks MAP:
Element 0
[Beta_true]
0.000000;
[Charge_true]
8.000000;
[Theta_true]
0.000000;
```

```
Event: 95
reco tracks MAP:
Element 0
[Beta]
0.685597;
[Charge]
2.000000;
[Theta]
3.245622;
Element 1
[Beta]
-1.313688;
[Charge]
0.000000;
[Theta]
3.473851;
Element 2
[Beta]
0.639217;
[Charge]
1.000000;
[Theta]
7.711094;
Element 3
[Beta]
0.634058;
[Charge]
1.000000;
```

MC reference Cuts & quantities

3

TANAactNtuMCref

In Loop( ):

For every TAMCParticle, three containers are associated:

- a container fo **element**-wise MC cuts

```
MC tracks MAP:
Element 0
[Beta_true]
0.000000;
[Charge_true]
8.000000;
[Theta_true]
0.000000;
```

key	description	values
`MCgoodEvent` `MCgoodEventID0`	The MC particle crosses the TG (so no fragmentation before it). If the ID ==0 it is also a primary particle.	**1**: the condition is hold **0** : the condition is not verified

MC reference Cuts & quantities

3

TANAactNtuMCref

In Loop( ):

For every TAMCParticle, three containers are associated:

- a container fo **element**-wise MC cuts
- a container for **track**-wise MC cuts

```
MC tracks MAP:
Element 0
[Beta_true]
0.000000;
[Charge_true]
8.000000;
[Theta_true]
0.000000;
```

key	description	values
`GoodParticle`	The particle is a primary or a fragment generated in the TG which `crosses the two planes of the TW and go beyond (so no secondary fragmentation)`	**1**: the condition is hold **0** : the condition is not verified

MC reference Cuts & quantities

3

TANAactNtuMCref

In Loop( ):

For every TAMCParticle, three containers are associated:

- a container fo **element**-wise MC cuts
- a container for **track**-wise MC cuts
- a container of **true quantities** of variables
  - The present variables are now: ***charge***, ***angle***, ***beta***

```
MC tracks MAP:
Element 0
[Beta_true]
0.000000;
[Charge_true]
8.000000;
[Theta_true]
0.000000;
```

Toward Cross Section Measurements, 1

4

TANAactCrossSection

In Loop( ):

Two **TTree** are filled with the retrieved quantities for **all the tracks**:

- *aTree* for **reco**-wise tracks
- *aTreeMC* for **MC_truth**-wise MC cuts

4 **TANAactCrossSection**

In Loop( ):

Two **TTree** are filled with the retrieved quantities for **all the tracks**:

- *aTree* for **reco**-wise tracks
- *aTreeMC* for **MC_truth**-wise MC cuts

```
// Define branches in the tree for the TTree element
fAnTree->Branch("Event_ID", &aTree.event_id);
fAnTree->Branch("Track_ID", &aTree.track_id);
fAnTree->Branch("Parameters", &aTree.parameters);
fAnTree->Branch("Parameters_truth", &aTree.parameters_truth);
fAnTree->Branch("RecoEvCuts", &aTree.RecoEvCuts);
fAnTree->Branch("RecoTrackCuts", &aTree.RecoTrackCuts);

// Define branches in the tree for the TTree element MC
fAnTreeMC->Branch("Event_ID", &aTreeMC.event_id);
fAnTreeMC->Branch("Track_ID", &aTreeMC.track_id);
fAnTreeMC->Branch("Parameters_truth", &aTreeMC.parameters_truth);
fAnTreeMC->Branch("EvCuts", &aTreeMC.RecoEvCuts);
fAnTreeMC->Branch("TRackCuts", &aTreeMC.RecoTrackCuts);
```

Toward Cross Section Measurements, 1

4 **TANAactCrossSection**

In Loop( ):

Two **TTree** are filled with the retrieved quantities for **all the tracks**:

- *aTree* for **reco**-wise tracks
- *aTreeMC* for **MC_truth**-wise MC cuts

```
// Define branches in the tree for the TTree element
fAnTree->Branch("Event_ID", &aTree.event_id);
fAnTree->Branch("Track_ID", &aTree.track_id);
fAnTree->Branch("Parameters", &aTree.parameters);
fAnTree->Branch("Parameters_truth", &aTree.parameters_truth);
fAnTree->Branch("RecoEvCuts", &aTree.RecoEvCuts);
fAnTree->Branch("RecoTrackCuts", &aTree.RecoTrackCuts);

// Define branches in the tree for the TTree element MC
fAnTreeMC->Branch("Event_ID", &aTreeMC.event_id);
fAnTreeMC->Branch("Track_ID", &aTreeMC.track_id);
fAnTreeMC->Branch("Parameters_truth", &aTreeMC.parameters_truth);
fAnTreeMC->Branch("EvCuts", &aTreeMC.RecoEvCuts);
fAnTreeMC->Branch("TRackCuts", &aTreeMC.RecoTrackCuts);
```

Maps from

1 TANAactNtuSelectionCuts

2 TANAactNtuGlbTrackCounts

3 TANAactNtuMCref

ROOT Declarative Analysis: RDataFrame

Customisation point, public interface!

Goals:

➔ Be the **fastest** way to manipulate HEP data
➔ Be the **go-to ROOT analysis interface** from laptop to cluster
➔ Consistent interfaces in **Python and C++**
➔ Top notch [documentation and examples]

https://indico.cern.ch/event/759388/contributions/3356304/attachments/1815599/2968077/RDF__HSF_JeffersonLab.pdf

Toward Cross Section Measurements, 3

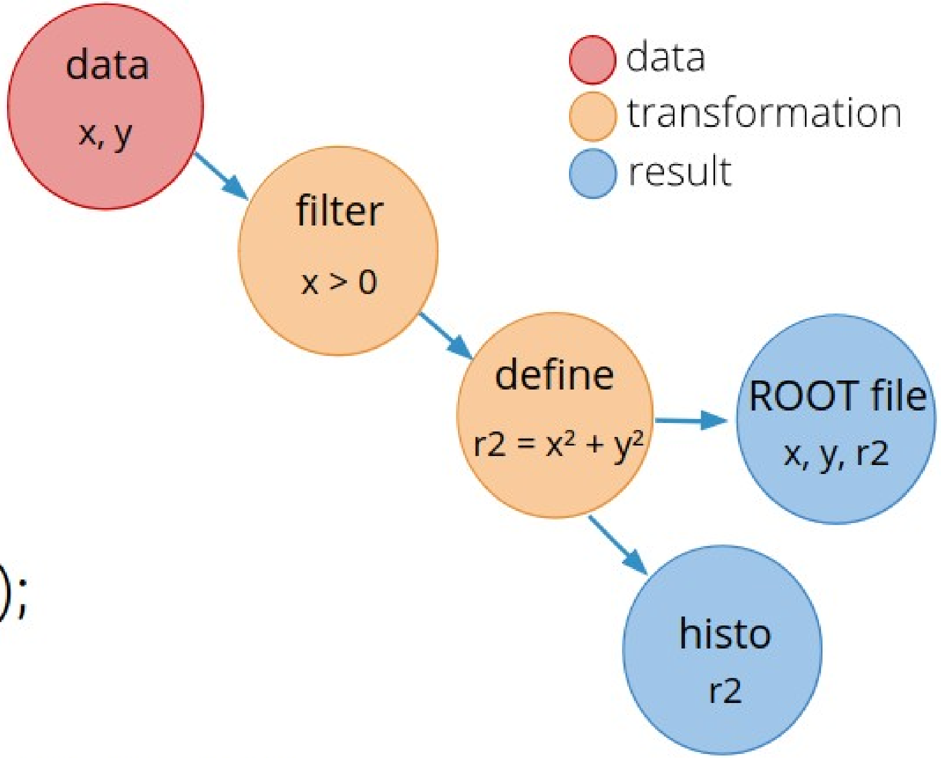
4

TANAactCrossSection

// Run a parallel analysis

```
ROOT::EnableImplicitMT();
ROOT::RDataFrame df(dataset);
auto df2 = df.Filter("x > 0")
             .Define("r2", "x*x + y*y");
auto rHist = df2.Histo1D("r2");
df2.Snapshot("newtree", "newfile.root");
```

Write datasets to disk, also in parallel.



Selection Cuts, 1

1 **TANAactNtuSelectionCuts**

In Loop( ):
For every reconstructed track, two cut maps are associated:

- *fEventCutsMap* for element-wise cuts
- *fTrackCutsMap* for track-wise cuts

- for every cut, a key of the map is generated
- an int value of **0,1** (or others if exception) is associated to each key

```
// event cuts
SCpileUpCut(); // add "SCcut" in event map
BMCut();  // add "BMcut" in event map
TwClonesCut(); // add "TWclone" in track map and "TWnum" cut in event map
NTracksCut(); // add "NTracksCut" in event map

// track cuts
for (int it = 0; it < nt; it++)
{
fGlbTrack = fNtuGlbTrack->GetTrack(it);
VtxPositionCut(it, fGlbTrack); // add "VTXposCut" cut in track map
TrackQualityCut(it,fGlbTrack); // add "TrackQuality" cut in track map

if (isMC){  // MC cuts
MC_VTMatch(it,fGlbTrack); // add "MC_VTMatch" cut in track map
MC_MSDMatch(it,fGlbTrack);  // add "MC_MSDMatch" cut in track map
MC_TwParticleOrigin(it,fGlbTrack); // add "MC_TwParticleOrigin" cut in track map

MC_isGoodReco(it,fGlbTrack); // add "MC_isGoodReco" cut in track map
```

Event cuts:
// Check if there is pile up in the SC, triggering an event
// Check if there is only one track in BM
// Check events with N° of tracks == N° of TW points
Check the tracks with the same TW point
// Check if N° of tracks for every event is > 1

Track cuts:
// Cuts about vtx position with the target dimensio
// Cuts about quality chi2 and residual of a track

// Compare the track with the MC_ID to infer if it is a good reco track

Reco quantities, 1

2

TANAactNtuGlbTrackCounts

In Loop( ):

The idea is to create a map for every track, in which the main variables (the one for cross sections) are inserted

- *fTrackGlbCounts_reco* for **reconstructed** values of variables
- *fTrackGlbCounts_MC* for **true** values of variables

```
for (int it = 0; it < nt; it++)
{
  fGlbTrack = fNtuGlbTrack->GetTrack(it);
  Float_t Z_reco = fGlbTrack->GetTwChargeZ();
  Float_t Th_reco = fGlbTrack->GetTgtThetaBm()* TMath::RadToDeg();
  Float_t Tof_meas = fGlbTrack->GetTwTof() - fPrimary_tof;
  Float_t Beta_reco = fGlbTrack->GetLength() / Tof_meas / TAGgeoTrafo::GetLightVelocity();

  fTrackGlbCounts_reco[it]["Charge"] = Z_reco;
  fTrackGlbCounts_reco[it]["Theta"] = Th_reco;
  fTrackGlbCounts_reco[it]["Beta"] = Beta_reco;
```

// Charge
// Theta
// Beta

```
mcNtuPart = (TAMCntuPart *)fpNtuMcTrk->Object();
TAMCpart *particle = mcNtuPart->GetTrack(TrkIdMC);
Float_t Z_true = particle->GetCharge();
Float_t Th_true = 0;    //TO BE MODIFIED
Float_t Beta_true = 0; //TO BE MODIFIED

fTrackGlbCounts_MC[it]["Charge_true"] = Z_true;
fTrackGlbCounts_MC[it]["Theta_true"] = Th_true;
fTrackGlbCounts_MC[it]["Beta_true"] = Beta_true;
```

// Charge
// Theta
// Beta

Analysis SHOE structure

- Based on the Analysis classes developed by Chris (see Analysis Meeting – 27/02/12 [1])
- final developments in SHOE branch *Ubaldi_temp*

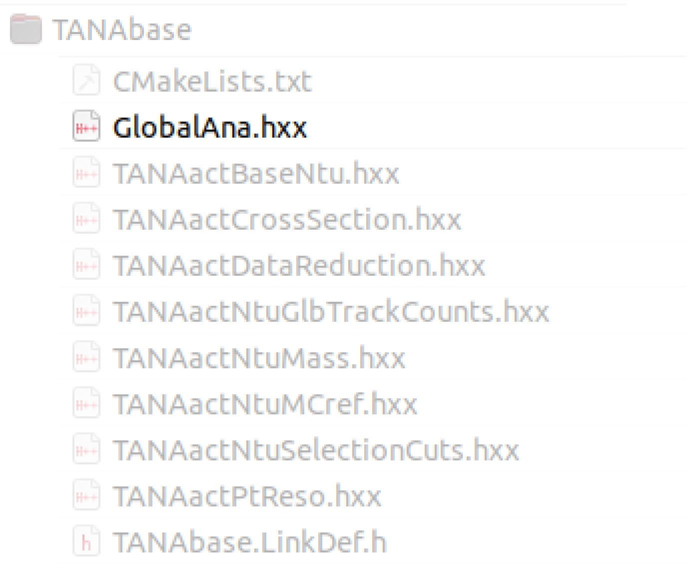
TANAbase

- CMakeLists.txt
- GlobalAna.hxx
- TANAactBaseNtu.hxx
- TANAactCrossSection.hxx
- TANAactDataReduction.hxx
- TANAactNtuGlbTrackCounts.hxx
- TANAactNtuMass.hxx
- TANAactNtuMCref.hxx
- TANAactNtuSelectionCuts.hxx
- TANAactPtReso.hxx
- TANAbase.LinkDef.h

[1] https://agenda.infn.it/event/40231/

Analysis SHOE structure

- Based on the Analysis classes developed by Chris (see Analysis Meeting – 27/02/12 [1])
- final developments in SHOE branch *Ubaldi_temp*

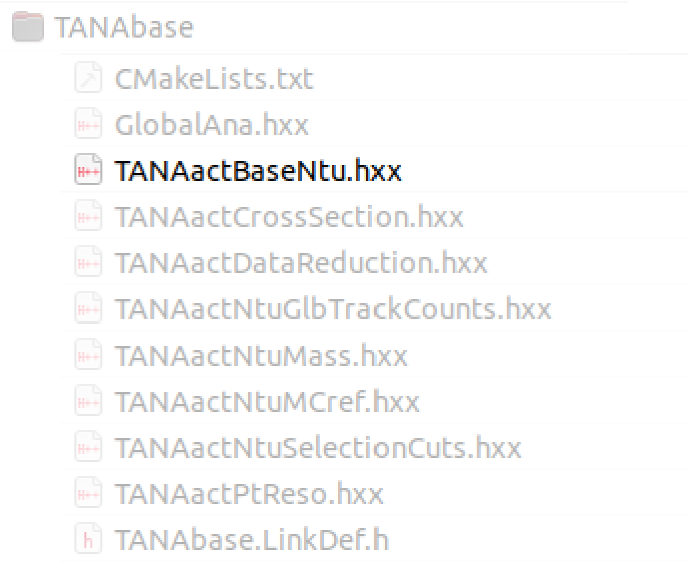


- based on the structure of BaseReco class
- Read all geomaps/config files for all included detectors
- Read all containers for the included detectors
- Create and require the dedicated class analysis

[1] ***https://agenda.infn.it/event/40231/***

Analysis SHOE structure

- Based on the Analysis classes developed by Chris (see Analysis Meeting – 27/02/12 [1])
- final developments in SHOE branch *Ubaldi_temp*

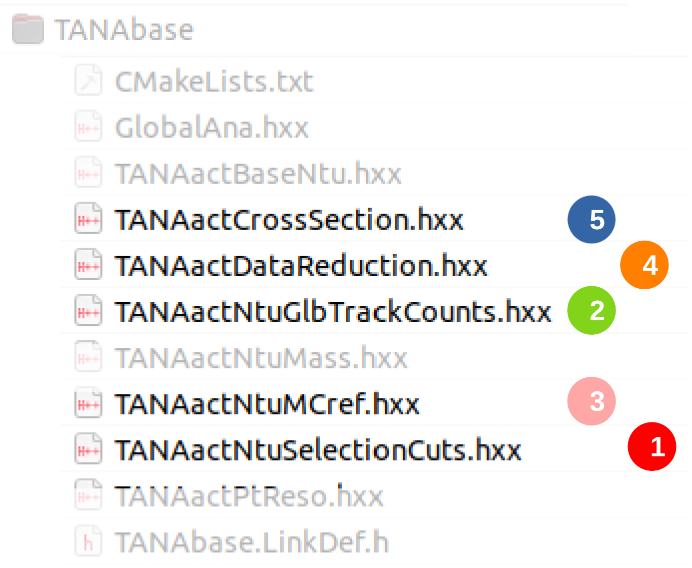


- contains the global track container, target/beam and FOOT geometry
- implementation of plotting methods

[1] https://agenda.infn.it/event/40231/

Analysis SHOE structure

- Based on the Analysis classes developed by Chris (see Analysis Meeting – 27/02/12 [1])
- final developments in SHOE branch *Ubaldi_temp*



Improved Interfaces

what we write

```
TTreeReader reader(data);
TTreeReaderValue<A> x(reader,"x");
TTreeReaderValue<B> y(reader,"y");
TTreeReaderValue<C> z(reader,"z");
while (reader.Next()) {
   if (IsGoodEntry(*x, *y, *z))
      h->Fill(*x);
}
```

what we *mean*

- full control over the event loop
- requires some boilerplate
- users implement common tasks again and again
- parallelisation is not trivial

RDataFrame: declarative analyses

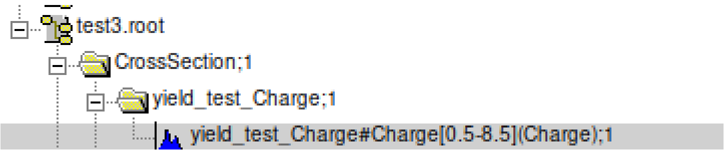
```cpp
RDataFrame d(data);
auto h = d.Filter(IsGoodEntry, {"x","y","z"})
          .Histo1D("x");
```

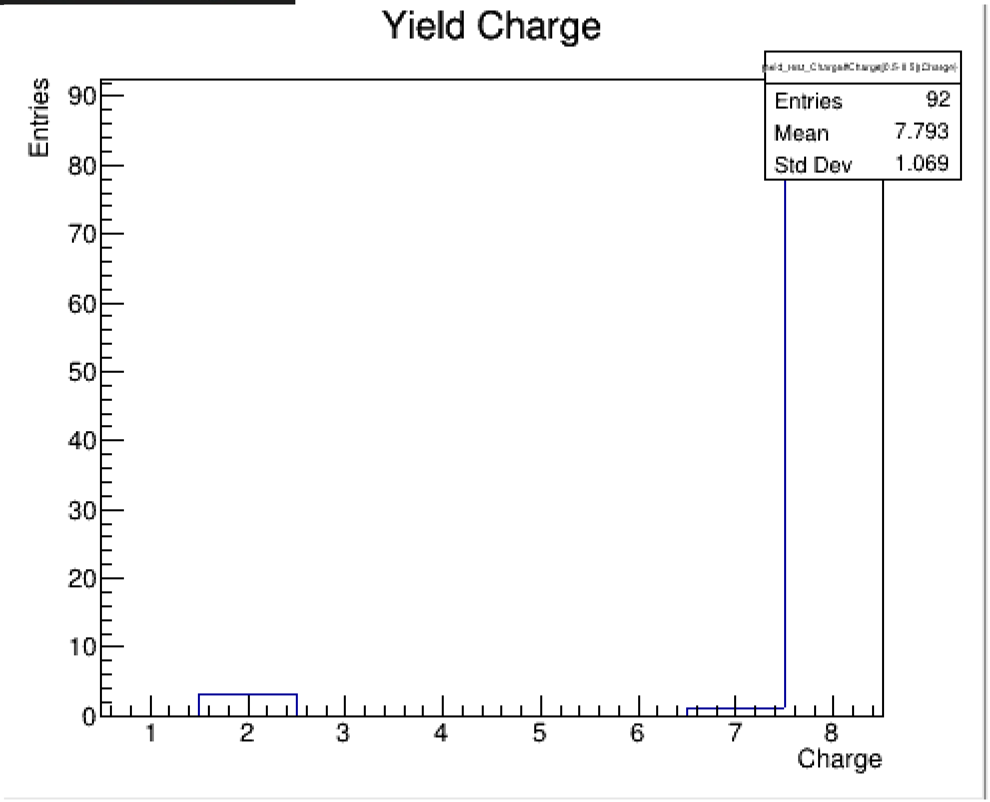
- full control over *the analysis*
- no boilerplate
- common tasks are already implemented

? parallelization is not trivial?

Charge yield

```
aEventCutsMap["TWnum"] = 1;
aTrackCutsMap["TrackQuality"] = 1;
aVariablesList.push_back("Charge");
FillYield(d, aEventCutsMap, aTrackCutsMap, aVariablesList, "yield_test_Charge");
```

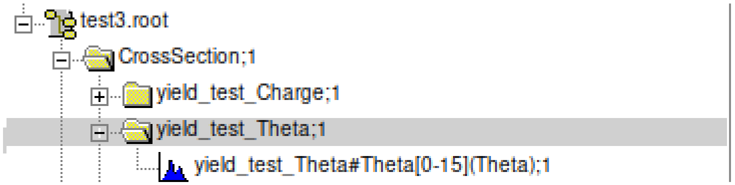


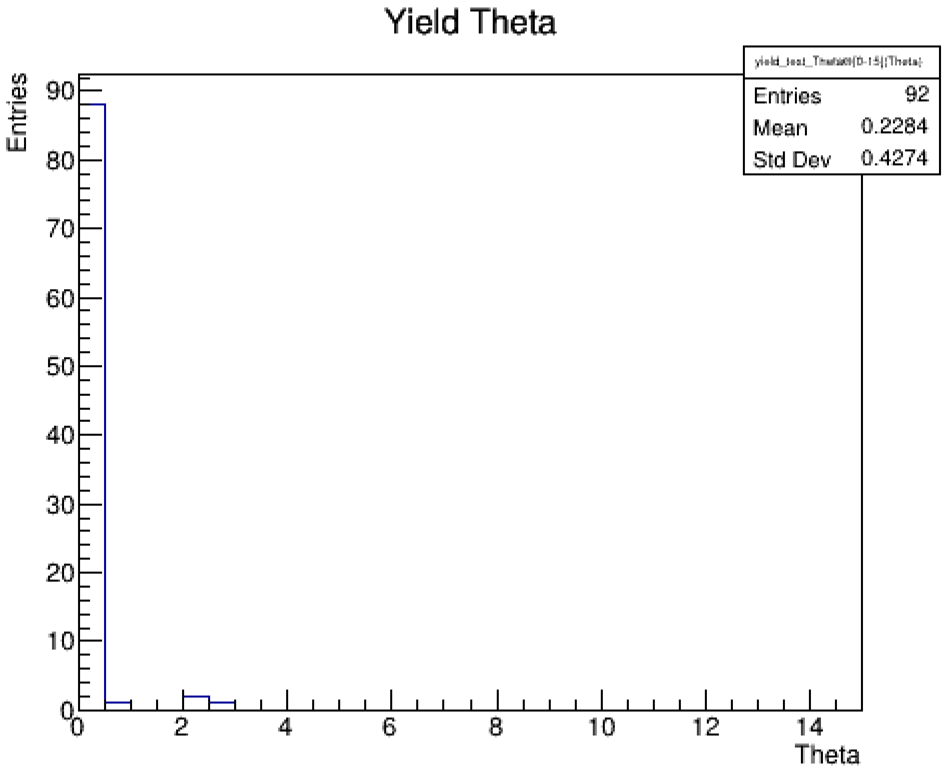


Examples

Theta yield

```
aEventCutsMap["TWnum"] = 1;
aTrackCutsMap["TrackQuality"] = 1;
aVariablesList.push_back("Theta");
FillYield(d, aEventCutsMap, aTrackCutsMap, aVariablesList, "yield_test_Theta");
```





Yields via RDataFrame, 1

4 **TANAactCrossSection**

After Loop( ):

Create histograms of (differential) **yields** of a specific variable, according to cuts

```
auto filter = d.Filter([aEventCutsMap](std::map<string, Int_t> EvCut) { ...
                        {"RecoEvCuts"})
                   .Filter([aTrackCutsMap](map<string, Int_t> TrackCut) {
                           {"RecoTrackCuts"})
                   .Filter([aVariablesList, values](map<string, Float_t> parameter)
                           {"Parameters"});
```

1 TANAactNtuSelectionCuts

1 TANAactNtuSelectionCuts

2 TANAactNtuGlbTrackCounts

```
TH1D hist = (TH1D)filter.Define("LastParam", last_param.Data())
                .Histo1D<float>({...}, "LastParam")
                .GetValue();
```

→ **Create histogram as output**

Yields via RDataFrame, 2

IMPORTANT

4 **TANAactCrossSection**

After Loop( ):

Create histograms of (differential) **yields** of a specific variable, according to cuts

same keys from 1 TANAactNtuSelectionCuts

```
// Add the elements of cuts I want (maybe from a parameter file?)
aEventCutsMap["TWnum"] = 1;
aTrackCutsMap["TrackQuality"] = 1;
// Variables of my Yields
aVariablesList.push_back("Charge");
FillYield(d, aEventCutsMap, aTrackCutsMap, aVariablesList, "yield_test_Charge");
aEventCutsMap.clear();
aVariablesList.clear();
aTrackCutsMap.clear();
```

same keys from 2 TANAactNtuGlbTrackCounts

Examples

Analysis comparison, reco level

new analysis

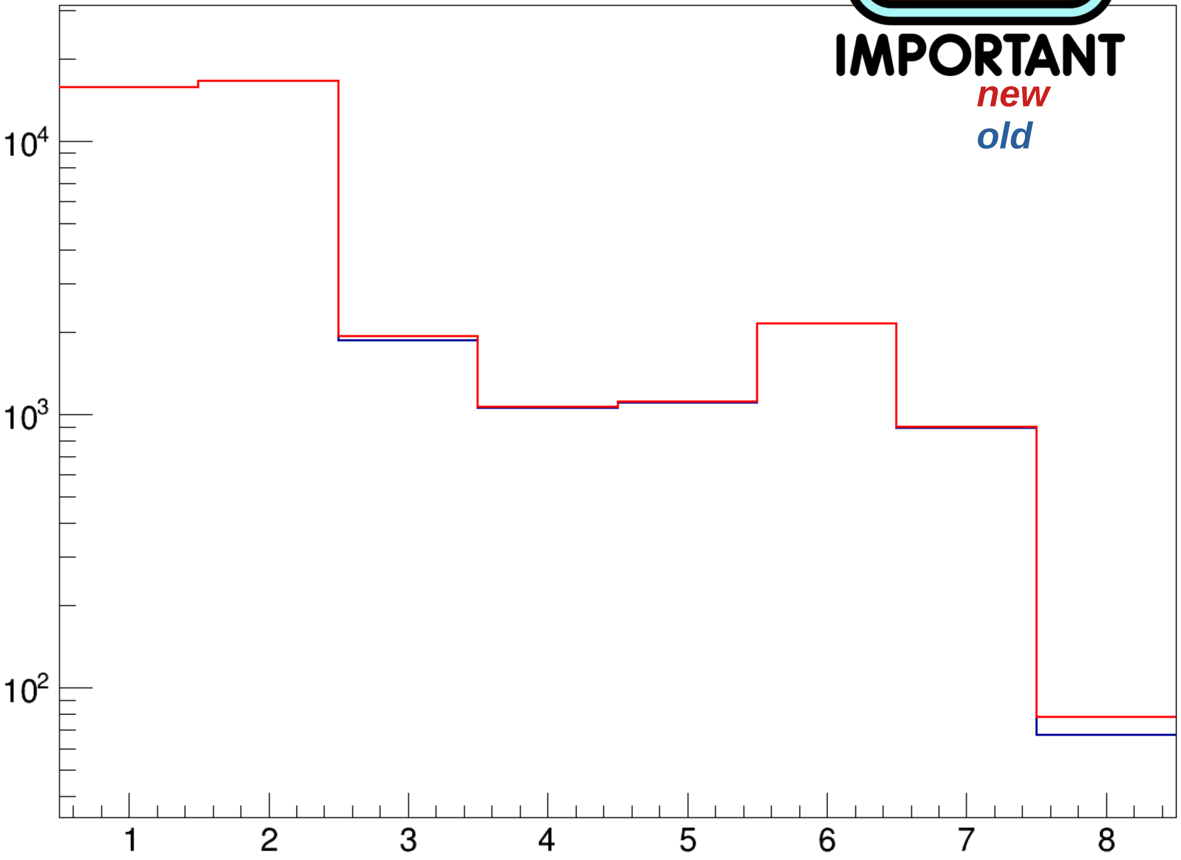
```
aEventCutsMap["BMcut"] = 1;
aEventCutsMap["NTracksCut"] = 1;
aEventCutsMap["TWnum"] = 1;


aTrackCutsMap["TWclone"] = 0;
aTrackCutsMap["TrackQuality"] = 1;
aTrackCutsMap["VTXposCut"] = 1;


// Variables of my Yields
aVariablesList.push_back("Charge");
FillYield(d, aEventCutsMap, aTrackCutsMap, aVariablesList, "yield
```

old analysis

```
// // =================== Chi2 cuts + multitrack
if (VTok && chi2Cut &&
residualCut && nt > 1 &&
hasSameTwPoint.at(it) == false &&
nt_TW == myTWNtuPt->GetPointsN()){
  MyReco("MChi2TWTngt");
}
```



Examples

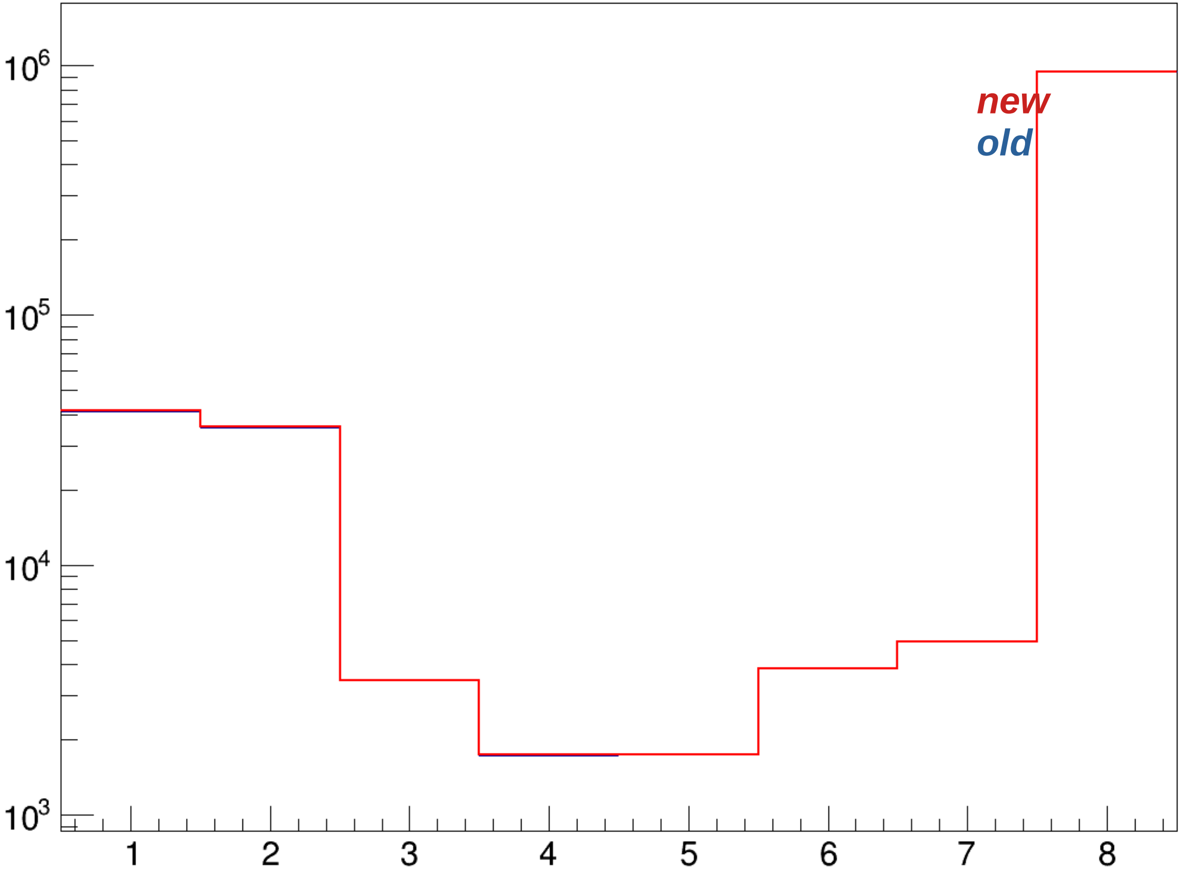
Analysis comparison, MC level

new analysis

```
aEventCutsMap["MCgoodEvent"] = 1;
aTrackCutsMap["GoodParticle"] = 1;
aVariablesList.push_back("Charge_true"
FillYield(d_MC, aEventCutsMap, aTrackC
```

old analysis

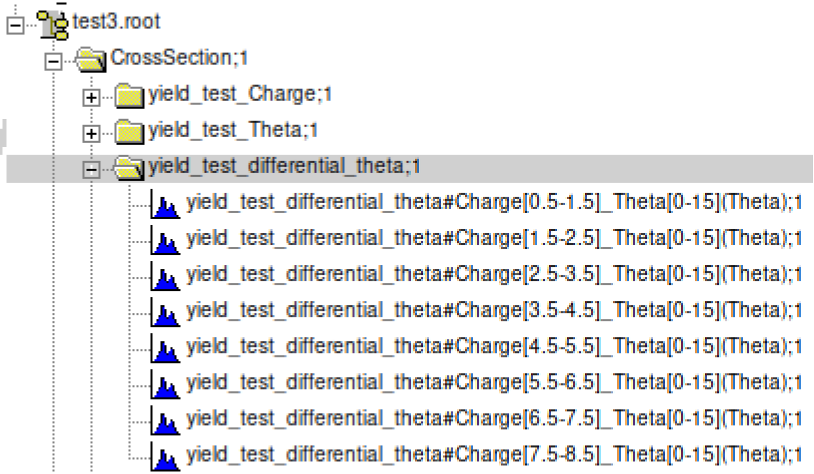
```
// MCParticleStudies();
//***** loop on every TAMCparticle:
FillMCPartYields(); // N_ref
```

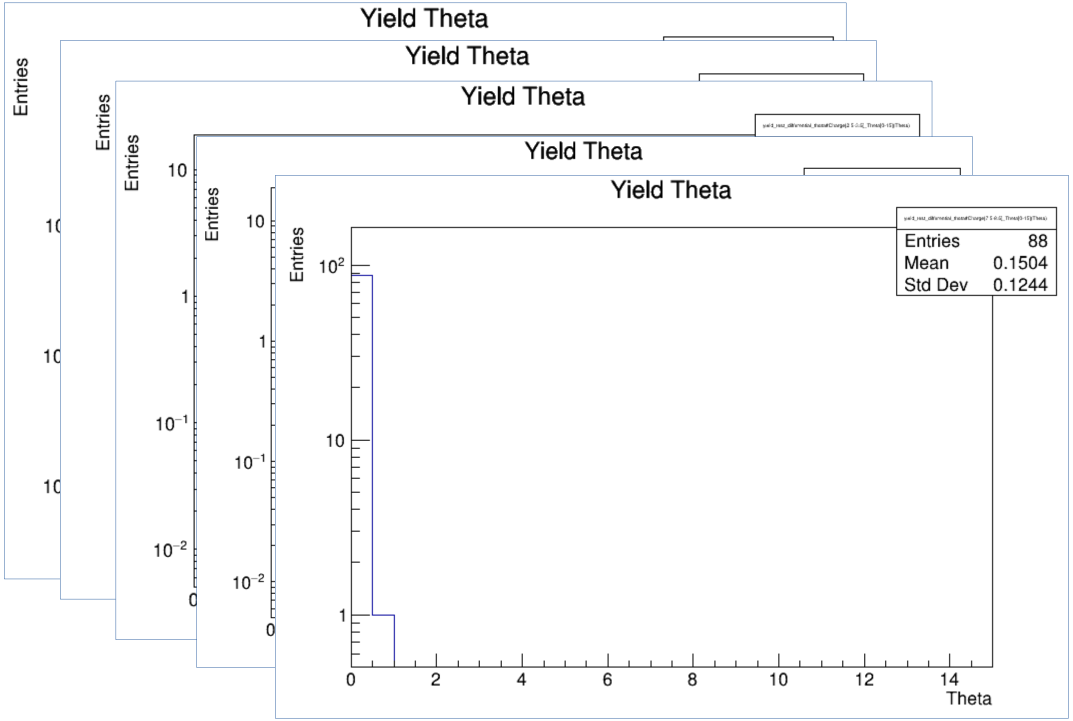


Examples

Theta yield, differential in charge

```
aEventCutsMap["TWnum"] = 1;
aTrackCutsMap["TrackQuality"] = 1;
aVariablesList.push_back("Charge");
aVariablesList.push_back("Theta");
FillYield(d, aEventCutsMap, aTrackCutsMap, aVariablesList, "yield_test_differential_theta");
```





Analysis strategy

In the analysis, I am considering the following levels:

MC DATASET

MC TRUTH LEVEL

RECO LEVEL

N_Reference (MC)

all TAMCParticles

- primary beams
- primary fragments generated in the TG

- which cross the beginning of TW

(all the particle inside the geometrical acceptance of the setup without secondary fragmentation beneath the detectors + check about gamma decay)

N_RECO

I take the **MC_ID** of the track

(most frequent id)

N_Good Reco

(Tracks with MC_ID which satisfies MC GENERATION requirements)

Analysis strategy

To compute angular differential cross section:

$$\frac{d\sigma}{d\theta}(Z,\theta) = \frac{Y(Z,\theta)}{N_{beam}\ N_{target}\ \Omega_\theta\ \epsilon(Z,\theta)}$$

where:

Y:	fragment counts	**N_RECO**
N_{beam}:	n° of primary events	
N_{target}:	n° of scattering centers per unit area	
ε:	efficiency	**N_Good Reco / N_Reference (MC)**
Ω_ϑ:	angular phase space	