WAYNE STATE UNIVERSITY

ALICE

# – Correlations analysis framework –

(Jan Fiete's seminal `AliAnalysisTaskPhiCorrelations`)

# – hands-on –

**Víctor González**

**O2 Analysis tutorial, Third edition, November 9 2023**

# Your first O2Physics correlations task

## Working layout

- **One working directory**
  - For holding test datafiles, configuration, results
  - For running the task
  - Enter the `O2Physics` environment
- **One production directory**
  - Usually `alice`
  - For producing `O2Physics` with `aliBuild`
- **Your preferred text editor**
  - For editing `firstcfcorrelations.cxx`

# First things first

- **Where is our code**

  `O2Physics/Tutorials/PWGCF/TwoParticleCorrelations/src/firstcfcorrelations.cxx`

- **Locate a "good" test file. We have selected**
  - Period `LHC17l`
  - Run 277907
  - Reconstruction pass `pass2`
- **In the documentation and in the Run3_Conversion train**

  https://twiki.cern.ch/twiki/bin/viewauth/ALICE/AliceO2WP14AF

  https://alimonitor.cern.ch/trains/train.jsp?train_id=132

**check that**

- Train run 316
- Train child 8

**Go to the `AOD` output and download the `AO2D` 46**

# Bonus in case we have enough time

## The data we will be using

- In order to showcase the use of derived data and current Pb-Pb 2023 datataking:
- Derived files based on Pb-Pb reconstruction in cpass1 have been generated for you!
  - Small file (**~5MB**): dropbox cernbox
  - Medium file (**~115MB**): dropbox cernbox
  - Large file (**~367MB**, equivalent to **6x10$^7$ events**): dropbox cernbox
    - Pretty cool: 60M events is 4x larger than all Pb-Pb data taken in 2010!
    - Thanks to derived data use, you will be able to analyse it in minutes in this session if you follow all instructions
- Do not worry about how this was generated for now. You'll learn more about this later!
  - But if you are anyway curious, the code that generated this is also in the Tutorials directory [1]
- My suggestion: start developing with the small file, but download the large file now (while you code)!
  - This way, once your work is done, you can run the final code over the large file
  - Later on, I'll also show you how to use multiple cores, a key functionality of O2, to analyse!
  - In O2 parlance, multi-core processing is achieved via what is called 'pipelining' (more on that later!)

# Let's start doing things

# Checking our input file

- **Modify the default `process` function**

```
void process(aod::Collisions const& collisions, aod::Tracks const&)
{
  LOGF(info, "Received %d collisions", collisions.size());
}
```

- **Produce `O2Physics`**
- **Run your code**

```
o2-analysis-cf-firstcfcorr -b | o2-analysis-collision-converter -b |
  o2-analysis-tracks-extra-converter -b | o2-analysis-zdc-converter -b |
  o2-analysis-bc-converter -b --aod-file AO2D_316_20220630-0042_child_8_AOD_046.root > execution.log
```

- **Check the log output**

# Configuring your task

```
static constexpr float cfgPairCutDefaults[1][5] = {{-1, -1, -1, -1, -1}};

struct firstcorrelations {
  Configurable<float> cfgZVtxCut = {"zvtxcut", 7.0, "Vertex z cut. Default 7 cm"};
  Configurable<float> cfgPtCutMin = {"minpt", 0.2, "Minimum accepted track pT. Default 0.2 GeV"};
  Configurable<float> cfgPtCutMax = {"maxpt", 5.0, "Maximum accepted track pT. Default 5.0 GeV"};
  Configurable<float> cfgEtaCut = {"etacut", 0.8, "Eta cut. Default 0.8"};

  Configurable<LabeledArray<float>> cfgPairCut{"cfgPairCut",
                                               {cfgPairCutDefaults[0],
                                                5,
                                                {"Photon", "K0", "Lambda", "Phi", "Rho"}},
                                                "Pair cuts on various particles"};

  ConfigurableAxis axisVertex{"axisVertex", {7, -7, 7}, "vertex axis for histograms"};
  ConfigurableAxis axisDeltaPhi{"axisDeltaPhi", {72, -constants::math::PIHalf, constants::math::PIHalf * 3},
                                "delta phi axis for histograms"};
  ConfigurableAxis axisDeltaEta{"axisDeltaEta", {40, -2, 2}, "delta eta axis for histograms"};
  ConfigurableAxis axisPtTrigger{"axisPtTrigger", {VARIABLE_WIDTH, 0.5, 1.0, 1.5, 2.0, 3.0, 4.0, 6.0, 10.0},
                                 "pt trigger axis for histograms"};
  ConfigurableAxis axisPtAssoc{"axisPtAssoc", {VARIABLE_WIDTH, 0.5, 1.0, 1.5, 2.0, 3.0, 4.0, 6.0},
                               "pt associated axis for histograms"};
  ConfigurableAxis axisMultiplicity{"axisMultiplicity", {VARIABLE_WIDTH, 0, 5, 10, 20, 30, 40, 50, 100.1},
                                    "multiplicity / centrality axis for histograms"};
  ConfigurableAxis axisVertexEfficiency{"axisVertexEfficiency", {10, -10, 10}, "vertex axis for efficiency histograms"};
  ConfigurableAxis axisEtaEfficiency{"axisEtaEfficiency", {20, -1.0, 1.0}, "eta axis for efficiency histograms"};
  ConfigurableAxis axisPtEfficiency{"axisPtEfficiency",
                                    {VARIABLE_WIDTH, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0, 1.25, 1.5, 1.75, 2.0, 2.25,
                                    2.5, 2.75, 3.0, 3.25, 3.5, 3.75, 4.0, 4.5, 5.0, 6.0, 7.0, 8.0},
                                    "pt axis for efficiency histograms"};
  void init(InitContext&)
```

# Track your headers and usings

## ... and move forward

```
#include "Framework/runDataProcessing.h"
#include "Framework/AnalysisTask.h"
#include "Framework/AnalysisDataModel.h"
#include "Framework/ASoAHelpers.h"
#include "Framework/HistogramRegistry.h"
#include "Framework/RunningWorkflowInfo.h"
#include "CommonConstants/MathConstants.h"
#include "Common/DataModel/EventSelection.h"
#include "Common/DataModel/TrackSelectionTables.h"
#include "Common/DataModel/Centrality.h"
#include "PWGCF/Core/CorrelationContainer.h"
#include "PWGCF/Core/PairCuts.h"

using namespace o2;
using namespace o2::framework;
using namespace o2::framework::expressions;
```

- **Compile, run, execute and have a look at the ‘dpl-config.json‘**

```
o2-analysis-cf-firstcfcorr -b | o2-analysis-collision-converter -b |
  o2-analysis-tracks-extra-converter -b | o2-analysis-zdc-converter -b |
  o2-analysis-bc-converter -b --aod-file AO2D_316_20220630-0042_child_8_AOD_046.root > execution.log
```

# The configuration file

**Use as source `dpl-config.json` and rename it, now or you can do it later**

```
{
    ..........................
    "firstcorrelations": {
        "zvtxcut": "7",
        "minpt": "0.200000003",
        "maxpt": "5",
        "etacut": "0.800000012",
        "cfgPairCut": {
            "labels_rows": "",
            "labels_cols": [
                "Photon",
                "K0",
                "Lambda",
                "Phi",
                "Rho"
            ],
            "values": [
                [
                    "-1",
                    "-1",
                    "-1",
                    "-1",
                    "-1"
                ]
            ]
        },
    ..........................
}
```

# **Task outputs and objects**, build and register them

```
OutputObj<CorrelationContainer> same{"sameEvent"};
OutputObj<CorrelationContainer> mixed{"mixedEvent"};
HistogramRegistry registry{"registry"};
PairCuts mPairCuts;
bool doPairCuts = false;

void init(InitContext&)
{
  registry.add("yields", "centrality vs pT vs eta", {HistType::kTH3F, {{100, 0, 100, "centrality"},
                                                                       {40, 0, 20, "p_{T}"}, {100, -2, 2, "#eta"}}});
  registry.add("etaphi", "centrality vs eta vs phi", {HistType::kTH3F, {{100, 0, 100, "centrality"},
                                                                        {100, -2, 2, "#eta"}, {200, 0, 2 * M_PI, "#varphi"}}});
  const int maxMixBin = axisMultiplicity->size() * axisVertex->size();
  registry.add("eventcount", "bin", {HistType::kTH1F, {{maxMixBin + 2, -2.5, -0.5 + maxMixBin, "bin"}}});
  mPairCuts.SetHistogramRegistry(&registry);
  if (cfgPairCut->get("Photon") > 0 || cfgPairCut->get("K0") > 0 || cfgPairCut->get("Lambda") > 0 ||
      cfgPairCut->get("Phi") > 0 || cfgPairCut->get("Rho") > 0) {
    mPairCuts.SetPairCut(PairCuts::Photon, cfgPairCut->get("Photon"));
    mPairCuts.SetPairCut(PairCuts::K0, cfgPairCut->get("K0"));
    mPairCuts.SetPairCut(PairCuts::Lambda, cfgPairCut->get("Lambda"));
    mPairCuts.SetPairCut(PairCuts::Phi, cfgPairCut->get("Phi"));
    mPairCuts.SetPairCut(PairCuts::Rho, cfgPairCut->get("Rho"));
    doPairCuts = true;
  }
  std::vector<AxisSpec> corrAxis = {{axisDeltaEta, "#Delta#eta"}, {axisPtAssoc, "p_{T} (GeV/c)"}, {axisPtTrigger, "p_{T} (GeV/c)"},
                                    {axisMultiplicity, "multiplicity / centrality"}, {axisDeltaPhi, "#Delta#varphi (rad)"},
                                    {axisVertex, "z-vtx (cm)"}};
  std::vector<AxisSpec> effAxis = {{axisEtaEfficiency, "#eta"}, {axisEtaEfficiency, "#eta"}, {axisPtEfficiency, "p_{T} (GeV/c)"},
                                   {axisVertexEfficiency, "z-vtx (cm)"}};
  same.setObject(new CorrelationContainer("sameEvent", "sameEvent", corrAxis, effAxis, {}));
  mixed.setObject(new CorrelationContainer("mixedEvent", "mixedEvent", corrAxis, effAxis, {}));
}
```

# Checking the results so far

- **Modify the default `process` function**

```
void process(aod::Collision const& collision, aod::Tracks const& tracks)
{
  float centrality = 50.0;
  for (auto& track : tracks) {
    registry.fill(HIST("yields"), centrality, track.pt(), track.eta());
    registry.fill(HIST("etaphi"), centrality, track.eta(), track.phi());
  }
}
```

- **Produce `O2Physics`**
- **Run your code**

```
o2-analysis-cf-firstcfcorr -b | o2-analysis-collision-converter -b |
  o2-analysis-tracks-extra-converter -b | o2-analysis-zdc-converter -b |
  o2-analysis-bc-converter -b --aod-file AO2D_316_20220630-0042_child_8_AOD_046.root > execution.log
```

- **Check the log output and your results file**

# Let's incorporate event selection and centrality

# Some helping processing

```
template <typename TCollision, typename TTracks>
void fillQA(TCollision collision, float centrality, TTracks tracks)
{
  for (auto& track : tracks) {
    registry.fill(HIST("yields"), centrality, track.pt(), track.eta());
    registry.fill(HIST("etaphi"), centrality, track.eta(), track.phi());
  }
}

template <typename TTarget, typename TCollision>
bool fillCollision(TTarget target, TCollision collision, float centrality)
{
  target->fillEvent(centrality, CorrelationContainer::kCFStepAll);

  if (!collision.alias_bit(kINT7) || !collision.sel7()) {
    return false;
  }

  target->fillEvent(centrality, CorrelationContainer::kCFStepReconstructed);

  return true;
}
```

# Incorporate event selection and centrality

- **Modify the default `process` function**

```
void process(soa::Join<aod::Collisions, aod::EvSels, aod::CentRun2V0Ms>>::iterator const& collision,
             aod::Tracks const& tracks)
{
  const auto centrality = collision.centRun2V0M();

  if (fillCollision(same, collision, centrality) == false) {
    return;
  }
  registry.fill(HIST("eventcount"), -2);
  fillQA(collision, centrality, tracks);
}
```

- **Produce `O2Physics`**

- **Run your code**

```
o2-analysis-cf-firstcfcorr -b --aod-file AO2D_316_20220630-0042_child_8_AOD_046.root |
  o2-analysis-collision-converter -b | o2-analysis-tracks-extra-converter -b |
  o2-analysis-zdc-converter -b | o2-analysis-bc-converter -b > execution.log
```

- **What is the problem?**

# Fixing missing input information

```
[33556:internal-dpl-aod-reader]: [20:33:22][ERROR] Exception caught:
 Couldn't get TTree "DF_2779070046299684250/O2centrun2v0m" from "AO2D_316_20220630-0042_child_8_AOD_046.root".
```

Subscribed to `centrality` table whose producer task was not invoked

```
o2-analysis-cf-firstcfcorr -b --aod-file AO2D_316_20220630-0042_child_8_AOD_046.root |
  o2-analysis-collision-converter -b | o2-analysis-centrality-table -b > execution.log
[34349:internal-dpl-aod-reader]: [20:42:53][ERROR] Exception caught:
 Couldn't get TTree "DF_2779070046299684250/O2mult" from "AO2D_316_20220630-0042_child_8_AOD_046.root".
```

`centrality` task needs `multiplicity` table whose producer was not invoked

```
o2-analysis-cf-firstcfcorr -b --aod-file AO2D_316_20220630-0042_child_8_AOD_046.root |
  o2-analysis-collision-converter -b | o2-analysis-centrality-table -b |
  o2-analysis-multiplicity-table -b > execution.log
[34524:internal-dpl-aod-reader]: [20:45:35][ERROR] Exception caught:
 Couldn't get TTree "DF_2779070046299684250/O2timestamps" from "AO2D_316_20220630-0042_child_8_AOD_046.root".
```

`multiplicity` task needs `timestamp` table whose producer was not invoked

```
o2-analysis-cf-firstcfcorr -b --aod-file AO2D_316_20220630-0042_child_8_AOD_046.root |
  o2-analysis-collision-converter -b | o2-analysis-centrality-table -b | > execution.log
  o2-analysis-multiplicity-table -b | o2-analysis-timestamp -b > execution.log
[35200:internal-dpl-aod-reader]: [20:57:57][ERROR] Exception caught:
 Couldn't get TTree "DF_2779070046299684250/O2evsel" from "AO2D_316_20220630-0042_child_8_AOD_046.root".
```

Subscribed to `event-selection` table whose producer task was not invoked

```
o2-analysis-cf-firstcfcorr -b --aod-file AO2D_316_20220630-0042_child_8_AOD_046.root |
  o2-analysis-collision-converter -b | o2-analysis-centrality-table -b | > execution.log
  o2-analysis-multiplicity-table -b | o2-analysis-timestamp -b | o2-analysis-event-selection -b > execution.log
```

# Get your configuration file copying from `dpl-config.json`

# Put in place collisions classification

```
std::vector<double> vtxBinsEdges{VARIABLE_WIDTH, -7.0f, -5.0f, -3.0f, -1.0f, 1.0f, 3.0f, 5.0f, 7.0f};
std::vector<double> multBinsEdges{VARIABLE_WIDTH, 0.0f, 5.0f, 10.0f, 20.0f, 30.0f, 40.0f, 50.0, 100.1f};
SliceCache cache;
ColumnBinningPolicy<aod::collision::PosZ, aod::cent::CentRun2V0M>
  bindingOnVtxAndMult{{vtxBinsEdges, multBinsEdges}, true}; // true is for 'ignore overflows' (true by default)
SameKindPair<soa::Filtered<soa::Join<aod::Collisions, aod::EvSels, aod::CentRun2V0Ms>>,
             soa::Filtered<soa::Join<aod::Tracks, aod::TrackSelection>>,
             ColumnBinningPolicy<aod::collision::PosZ, aod::cent::CentRun2V0M>>
  pair{bindingOnVtxAndMult, 5, -1, &cache}; // indicates that 5 events should be mixed and under/overflow (-1) to be ignored
```

# Tracks and collisions selection + subscription

```
Filter collisionZVtxFilter = nabs(aod::collision::posZ) < cfgZVtxCut;
Filter trackFilter = (nabs(aod::track::eta) < cfgEtaCut) && (aod::track::pt > cfgPtCutMin) && (aod::track::pt < cfgPtCutMax) &&
                     (requireGlobalTrackInFilter() || (aod::track::isGlobalTrackSDD == (uint8_t) true));


void processSame(soa::Filtered<soa::Join<aod::Collisions, aod::EvSels, aod::CentRun2V0Ms>>::iterator const& collision,
                 soa::Filtered<soa::Join<aod::Tracks, aod::TrackSelection>> const& tracks)
{
}
PROCESS_SWITCH(firstcorrelations, processSame, "Process same event", true);

void processMixed(soa::Filtered<soa::Join<aod::Collisions, aod::EvSels, aod::CentRun2V0Ms>>& collisions,
                  soa::Filtered<soa::Join<aod::Tracks, aod::TrackSelection>> const& tracks)
{
}
PROCESS_SWITCH(firstcorrelations, processMixed, "Process mixed events", true);
```

- **Compile, run, execute and have a look at the ‘dpl-config.json‘**

```
o2-analysis-cf-firstcfcorr -b --configuration json://handsonconfig.json |
  o2-analysis-collision-converter -b --configuration json://handsonconfig.json |
  o2-analysis-tracks-extra-converter -b --configuration json://handsonconfig.json |
  o2-analysis-zdc-converter -b --configuration json://handsonconfig.json |
  o2-analysis-bc-converter -b --configuration json://handsonconfig.json |
  o2-analysis-centrality-table -b --configuration json://handsonconfig.json |
  o2-analysis-multiplicity-table -b --configuration json://handsonconfig.json |
  o2-analysis-timestamp -b --configuration json://handsonconfig.json |
  o2-analysis-trackselection -b --configuration json://handsonconfig.json |
  o2-analysis-trackextension -b --configuration json://handsonconfig.json |
  o2-analysis-event-selection -b --configuration json://handsonconfig.json > execution.log
```

# The configuration file

## Copy from dpl-config.json

```
{
    ..................
    "firstcorrelations": {
       ..................
       "processSame": "true",
       "processMixed": "true"
    },
    ..................
}
```

# Do the processing

```
template <typename TTarget, typename TTracks>
void fillCorrelations(TTarget target, TTracks tracks1, TTracks tracks2, float centrality, float posZ)
{
  for (auto& track1 : tracks1) {
    target->getTriggerHist()->Fill(CorrelationContainer::kCFStepReconstructed, track1.pt(), centrality, posZ, 1.0);

    for (auto& track2 : tracks2) {
      if (track1 == track2) {
        continue;
      }
      if (doPairCuts && mPairCuts.conversionCuts(track1, track2)) {
        continue;
      }
      float deltaPhi = track1.phi() - track2.phi();
      if (deltaPhi > 1.5f * PI) {
        deltaPhi -= TwoPI;
      }
      if (deltaPhi < -PIHalf) {
        deltaPhi += TwoPI;
      }

      target->getPairHist()->Fill(CorrelationContainer::kCFStepReconstructed,
                                  track1.eta() - track2.eta(), track2.pt(), track1.pt(), centrality, deltaPhi, posZ,
                                  1.0);
    }
  }
}
```

# Process same events

```
void processSame(soa::Filtered<soa::Join<aod::Collisions, aod::EvSels, aod::CentRun2V0Ms>>::iterator const& collision,
                 soa::Filtered<soa::Join<aod::Tracks, aod::TrackSelection>> const& tracks)
{
  const auto centrality = collision.centV0M();

  if (fillCollision(same, collision, centrality) == false) {
    return;
  }
  registry.fill(HIST("eventcount"), -2);
  fillQA(collision, centrality, tracks);
  fillCorrelations(same, tracks, tracks, centrality, collision.posZ());
}
```

# Process mixed events

```
void processMixed(soa::Filtered<soa::Join<aod::Collisions, aod::EvSels, aod::CentRun2V0Ms>>& collisions,
                  soa::Filtered<soa::Join<aod::Tracks, aod::TrackSelection>> const& tracks)
{
  for (auto& [collision1, tracks1, collision2, tracks2] : pair) {

    if (fillCollision(mixed, collision1, collision1.centRun2V0M()) == false) {
      continue;
    }
    registry.fill(HIST("eventcount"), bindingOnVtxAndMult.getBin({collision1.posZ(), collision1.centRun2V0M()}));

    fillCorrelations(mixed, tracks1, tracks2, collision1.centRun2V0M(), collision1.posZ());
  }
}
```

# Finally ...

- **Compile, run, execute and have a look at the results file**

```
o2-analysis-cf-firstcfcorr -b --configuration json://handsonconfig.json |
  o2-analysis-collision-converter -b --configuration json://handsonconfig.json |
  o2-analysis-tracks-extra-converter -b --configuration json://handsonconfig.json |
  o2-analysis-zdc-converter -b --configuration json://handsonconfig.json |
  o2-analysis-bc-converter -b --configuration json://handsonconfig.json |
  o2-analysis-centrality-table -b --configuration json://handsonconfig.json |
  o2-analysis-multiplicity-table -b --configuration json://handsonconfig.json |
  o2-analysis-timestamp -b --configuration json://handsonconfig.json |
  o2-analysis-trackselection -b --configuration json://handsonconfig.json |
  o2-analysis-trackextension -b --configuration json://handsonconfig.json |
  o2-analysis-event-selection -b --configuration json://handsonconfig.json > execution.log
```

# How to keep learning

- **O2 documentation**
  - https://aliceo2group.github.io/analysis-framework/
- **O2Physics tutorials**
  - https://aliceo2group.github.io/analysis-framework/docs/tutorials/
  - `alice/O2Physics/Tutorials`
- **Your colleagues O2Physics tasks**
  - `alice/O2Physics/PWG...`
- **O2 Analysis mattermost channel**
  - https://mattermost.web.cern.ch/alice/channels/o2-analysis
- **Hyperloop Operation mattermost channel**
  - https://mattermost.web.cern.ch/alice/channels/o2-hyperloop-operation

# Bonus: Run 3 2023 Pb–Pb derived data

# Bonus in case we have enough time

## The data we will be using

- In order to showcase the use of derived data and current Pb-Pb 2023 datataking:
- Derived files based on Pb-Pb reconstruction in cpass1 have been generated for you!
  - Small file (**~5MB**): dropbox cernbox
  - Medium file (**~115MB**): dropbox cernbox
  - Large file (**~367MB**, equivalent to **6x10$^7$ events**): dropbox cernbox
    - Pretty cool: 60M events is 4x larger than all Pb-Pb data taken in 2010!
    - Thanks to derived data use, you will be able to analyse it in minutes in this session if you follow all instructions
- Do not worry about how this was generated for now. You'll learn more about this later!
  - But if you are anyway curious, the code that generated this is also in the Tutorials directory [1]
- My suggestion: start developing with the small file, but download the large file now (while you code)!
  - This way, once your work is done, you can run the final code over the large file
  - Later on, I'll also show you how to use multiple cores, a key functionality of O2, to analyse!
  - In O2 parlance, multi-core processing is achieved via what is called 'pipelining' (more on that later!)

# Goals when coding

- **Do not repeat code unless it is neccesary**
- **Do not repeat code unless it is neccesary**
- **Do not copy tables content in own structures**
- **Do not copy tables content in own structures**

- **Templates come for helping**

- **The procedure is exactly the same for MC analysis**

# Tweaking the helpers
## Template fundation

```
enum datatype {
  kRaw,
  kDerived
};

  template <datatype dt, typename TTarget, typename TCollision>
  bool fillCollision(TTarget target, TCollision collision, float centrality)
  {
    target->fillEvent(centrality, CorrelationContainer::kCFStepAll);

    if constexpr (dt == kRaw) {
      if (!collision.alias_bit(kINT7) || !collision.sel7()) {
        return false;
      }
    }

    target->fillEvent(centrality, CorrelationContainer::kCFStepReconstructed);

    return true;
  }
```

# Tweaking the processing
## Template fundation

```
template <datatype dt, typename TTarget, typename TTracks>
void fillCorrelations(TTarget target, TTracks tracks1, TTracks tracks2, float centrality, float posZ)
{
  for (auto& track1 : tracks1) {
    target->getTriggerHist()->Fill(CorrelationContainer::kCFStepReconstructed, track1.pt(), centrality, posZ, 1.0);

    for (auto& track2 : tracks2) {
      if (track1 == track2) {
        continue;
      }
      if constexpr (dt == kRaw) {
        if (doPairCuts && mPairCuts.conversionCuts(track1, track2)) {
          continue;
        }
      }
      float deltaPhi = track1.phi() - track2.phi();
      if (deltaPhi > 1.5f * PI) {
        deltaPhi -= TwoPI;
      }
      if (deltaPhi < -PIHalf) {
        deltaPhi += TwoPI;
      }

      target->getPairHist()->Fill(CorrelationContainer::kCFStepReconstructed,
                                  track1.eta() - track2.eta(), track2.pt(), track1.pt(), centrality, deltaPhi, posZ,
                                  1.0);
    }
  }
}
```

# Addressing same events processing

```
template <datatype dt, typename TCollision, typename TTracks>
void processSame(TCollision collision, TTracks tracks, float centrality)
{
  if (fillCollision<dt>(same, collision, centrality) == false) {
    return;
  }
  registry.fill(HIST("eventcount"), -2);
  fillQA(collision, centrality, tracks);
  fillCorrelations<dt>(same, tracks, tracks, centrality, collision.posZ());
}

void processRawSame(soa::Filtered<soa::Join<aod::Collisions, aod::EvSels, aod::CentRun2V0Ms>>::iterator const& collision,
                    soa::Filtered<soa::Join<aod::Tracks, aod::TrackSelection>> const& tracks)
{
  const auto centrality = collision.centRun2V0M();
  processSame<kRaw>(collision, tracks, centrality);
}
PROCESS_SWITCH(firstcorrelations, processRawSame, "Process same event", true);

Filter derivedcollisionZVtxFilter = nabs(aod::collision::posZ) < cfgZVtxCut;
Filter derivedtrackFilter = (nabs(aod::exampleTrackSpace::eta) < cfgEtaCut) && (aod::exampleTrackSpace::pt > cfgPtCutMin) && (aod::e

void processDerivedSame(soa::Filtered<aod::DrCollisions>::iterator const& collision, soa::Filtered<aod::DrTracks> const& tracks)
{
  const float centrality = 45.0;
  processSame<kDerived>(collision, tracks, centrality);
}
PROCESS_SWITCH(firstcorrelations, processDerivedSame, "Process same derived data event", false);
```

# Addressing mixed events processing

```
void processRawMixed(soa::Filtered<soa::Join<aod::Collisions, aod::EvSels, aod::CentRun2V0Ms>>& collisions,
                     soa::Filtered<soa::Join<aod::Tracks, aod::TrackSelection>> const& tracks)
{
  LOGF(info, "Received %d collisions", collisions.size());
  for (auto& [collision1, tracks1, collision2, tracks2] : pair) {

    if (fillCollision<kRaw>(mixed, collision1, collision1.centRun2V0M()) == false) {
      continue;
    }
    registry.fill(HIST("eventcount"), bindingOnVtxAndMult.getBin({collision1.posZ(), collision1.centRun2V0M()}));

    fillCorrelations<kRaw>(mixed, tracks1, tracks2, collision1.centRun2V0M(), collision1.posZ());
  }
}
PROCESS_SWITCH(firstcorrelations, processRawMixed, "Process mixed events", true);
```

# Addressing mixed events processing

```
ColumnBinningPolicy<aod::collision::PosZ> bindingOnVtx{{vtxBinsEdges}, true}; // true is for 'ignore overflows' (true by default)
SameKindPair<soa::Filtered<aod::DrCollisions>, soa::Filtered<aod::DrTracks>,
             ColumnBinningPolicy<aod::collision::PosZ>> derivedpair{bindingOnVtx, 5, -1, &cache}; // indicates that 5 events should be mixe
void processDerivedMixed(soa::Filtered<aod::DrCollisions>& collisions, soa::Filtered<aod::DrTracks> const& tracks)
{
  LOGF(info, "Received %d derived data collisions", collisions.size());
  const float derivedcentrality = 45.0;
  for (auto& [collision1, tracks1, collision2, tracks2] : derivedpair) {

    if (fillCollision<kDerived>(mixed, collision1, derivedcentrality) == false) {
      continue;
    }
    registry.fill(HIST("eventcount"), bindingOnVtx.getBin({collision1.posZ()}));

    // TODO mixed event weight missing
    fillCorrelations<kDerived>(mixed, tracks1, tracks2, derivedcentrality, collision1.posZ());
  }
}
PROCESS_SWITCH(firstcorrelations, processDerivedMixed, "Process mixed derived data events", false);
```

# The configuration file

## Copy from dpl-config.json

```
{
    ..................
    "firstcorrelations": {
        ..................
        "processRawSame": "true",
        "processDerivedSame": "false",
        "processRawMixed": "true",
        "processDerivedMixed": "false"
    },
    ..................
}
```

- **Compile, run, execute and have a look at the results file**

```
o2-analysis-cf-firstcfcorr -b --configuration json://handsonconfig.json  > execution.log
```

**Don't forget to switch file names when derived**