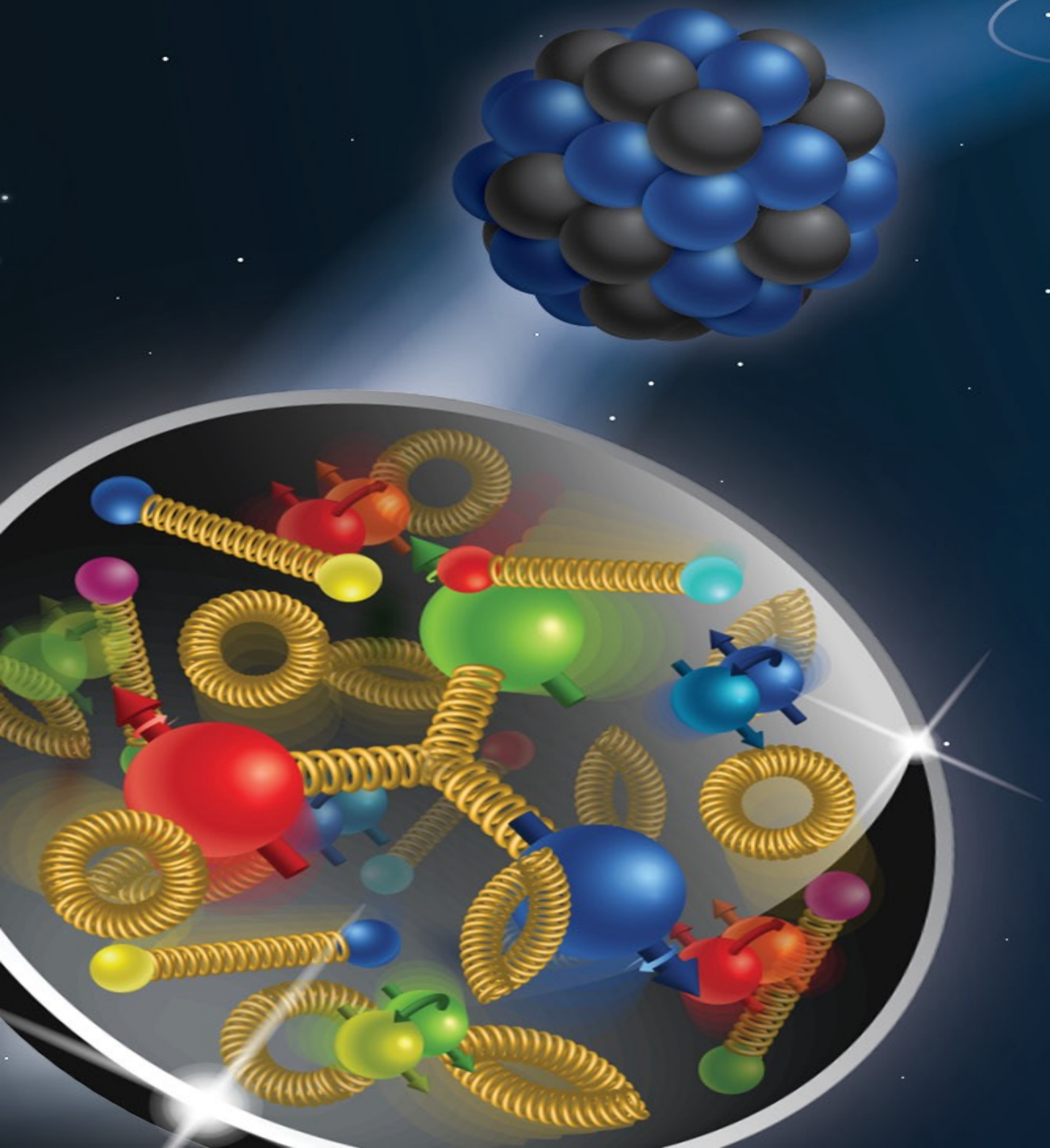


Far-Forward Reconstruction using ePIC DD4HEP + EICRecon

Alex Jentsch (BNL)

ePIC Exclusive, Diffractive, and Tagging PWG Meeting

January 10th, 2024

General overview of geometry

- All detectors have full geometry implementation…but…
  - Vacuum system is not remotely finalized – this affects especially the ZDC performance.
  - Anything done with the ZDC *now* will likely need another look after the material is updated (1-2 months).
  - No real support structure for FF detectors – but realistic material layers included to study impact of material budget on tracking.
  - Magnetic fields are only "official" for primary beam energies (18x275, 10x100, 5x41) – we have other configurations in the compact files…but they need to be used with care.
  - B0 magnetic field map is *there*, but not in the official main branch for now.

Reconstruction – Roman Pots

- Matrix method is used for reconstruction.
  - Matrix is dependent on the magnets – has to be re-calculated for any changes/abnormal circumstances.
  - Beam energy used for simulations MUST be given to EICrecon***
    - **EICrecon/src/algorithms/fardetectors/MatrixTransferStaticConfig.h**

See “ForwardRomanPotRecParticles” branch.

```
#pragma once

namespace eicrecon {

  struct MatrixTransferStaticConfig {

    float     partMass  {0.938272};
    float     partCharge{1};
    long long partPDG   {2212};

    // Defaults here are for RPOTS
    double local_x_offset      {0.0};
    double local_y_offset      {0.0};
    double local_x_slope_offset{-0.00622147};
    double local_y_slope_offset{-0.0451035};
    double crossingAngle       {0.025};
    double nomMomentum         {275.0};

    std::vector<std::vector<double>> aX = {{2.102403743, 29.11067626},
                                           {0.186640381, 0.192604619}};
    std::vector<std::vector<double>> aY = {{0.0000159900, 3.94082098},
                                           {0.0000079946, -0.1402995}};


    double hit1minZ{0};
    double hit1maxZ{0};
    double hit2minZ{0};
    double hit2maxZ{0};

    std::string readout{""};

  };
```

***we need to automate this – has not proven to be straight-forward previously.

Reconstruction – Off-Momentum

- Similar to Roman pots – uses transfer matrix
  - Matrix is dependent on the magnets – has to be re-calculated for any changes/abnormal circumstances.
  - Beam energy used for simulations MUST be given to EICrecon***
    - **EICrecon/src/algorithms/fardetectors/MatrixTransferStaticConfig.h**
    - More complicated since you have to use the momentum of the “spectator” you are interested in.

```
#pragma once

namespace eicrecon {

  struct MatrixTransferStaticConfig {

    float     partMass  {0.938272};
    float     partCharge{1};
    long long partPDG   {2212};

    // Defaults here are for RPOTS
    double local_x_offset      {0.0};
    double local_y_offset      {0.0};
    double local_x_slope_offset{-0.00622147};
    double local_y_slope_offset{-0.0451035};
    double crossingAngle       {0.025};
    double nomMomentum         {275.0};

    std::vector<std::vector<double>> aX = {{2.102403743, 29.11067626},
                                           {0.186640381, 0.192604619}};
    std::vector<std::vector<double>> aY = {{0.0000159900, 3.94082098},
                                           {0.0000079946, -0.1402995}};


    double hit1minZ{0};
    double hit1maxZ{0};
    double hit2minZ{0};
    double hit2maxZ{0};

    std::string readout{""};

```

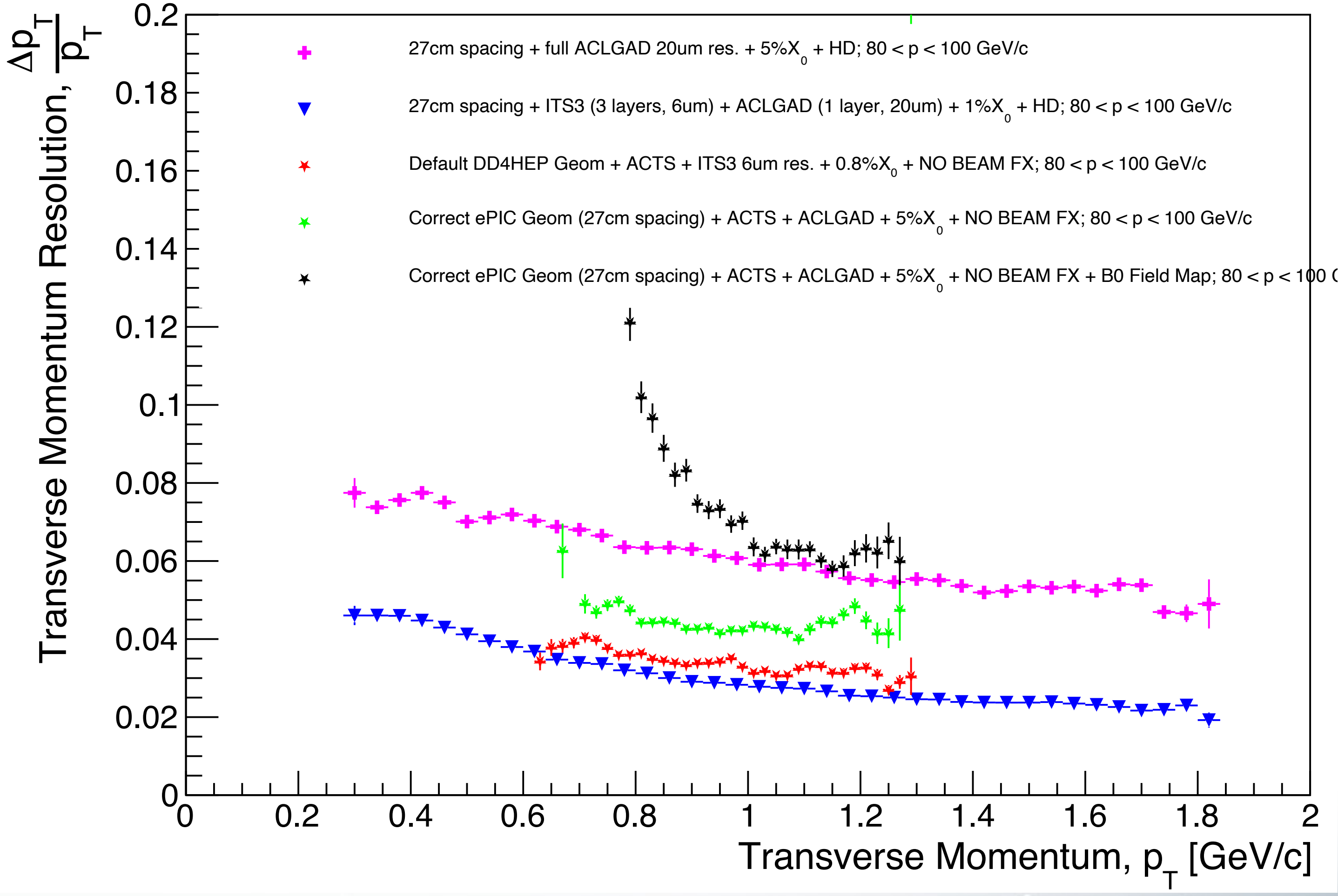
Example: e+d 10x110 GeV/n → nomMomentum = 110.0

See “ForwardOffMRecParticles” branch.

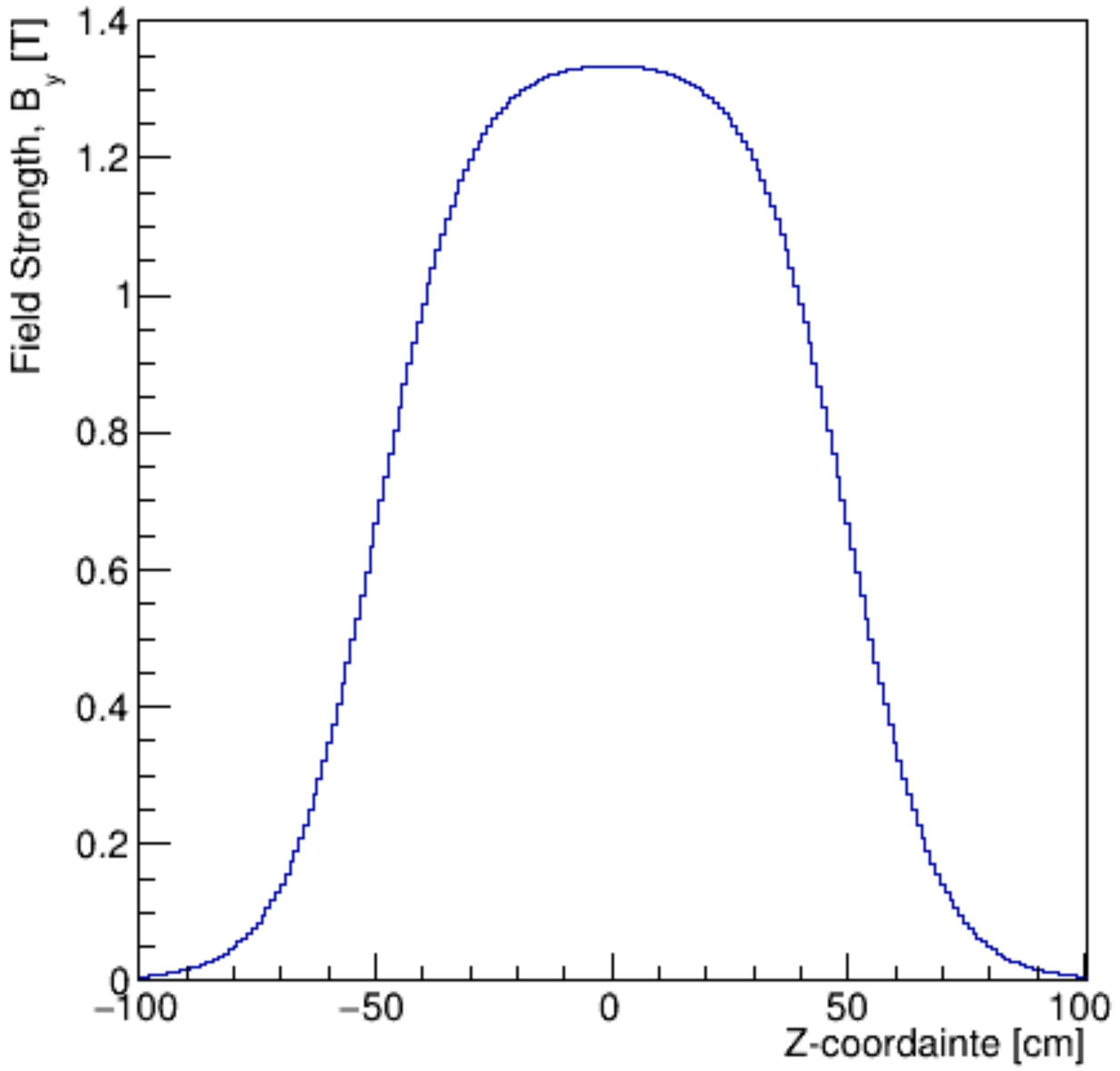
***we need to automate this – has not proven to be straight-forward previously.

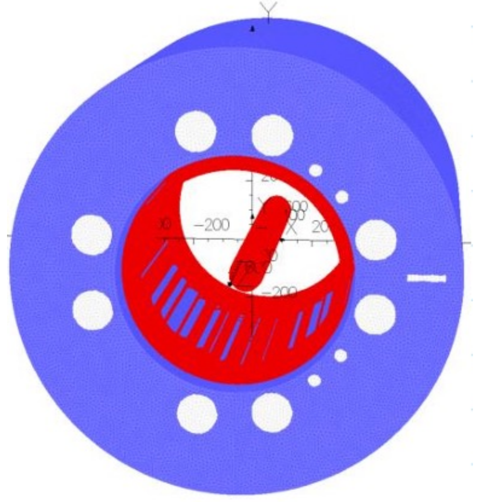
B0 Tracker

- In principle works out of the box.
  - See ”ReconstructedChargedParticles” branch.
  - Uses ACTS, should function just like analysis with main tracker.
  - For now, still using constant dipole + quad field – fieldmap is available and will be the “standard” soon.



B0 Field Map





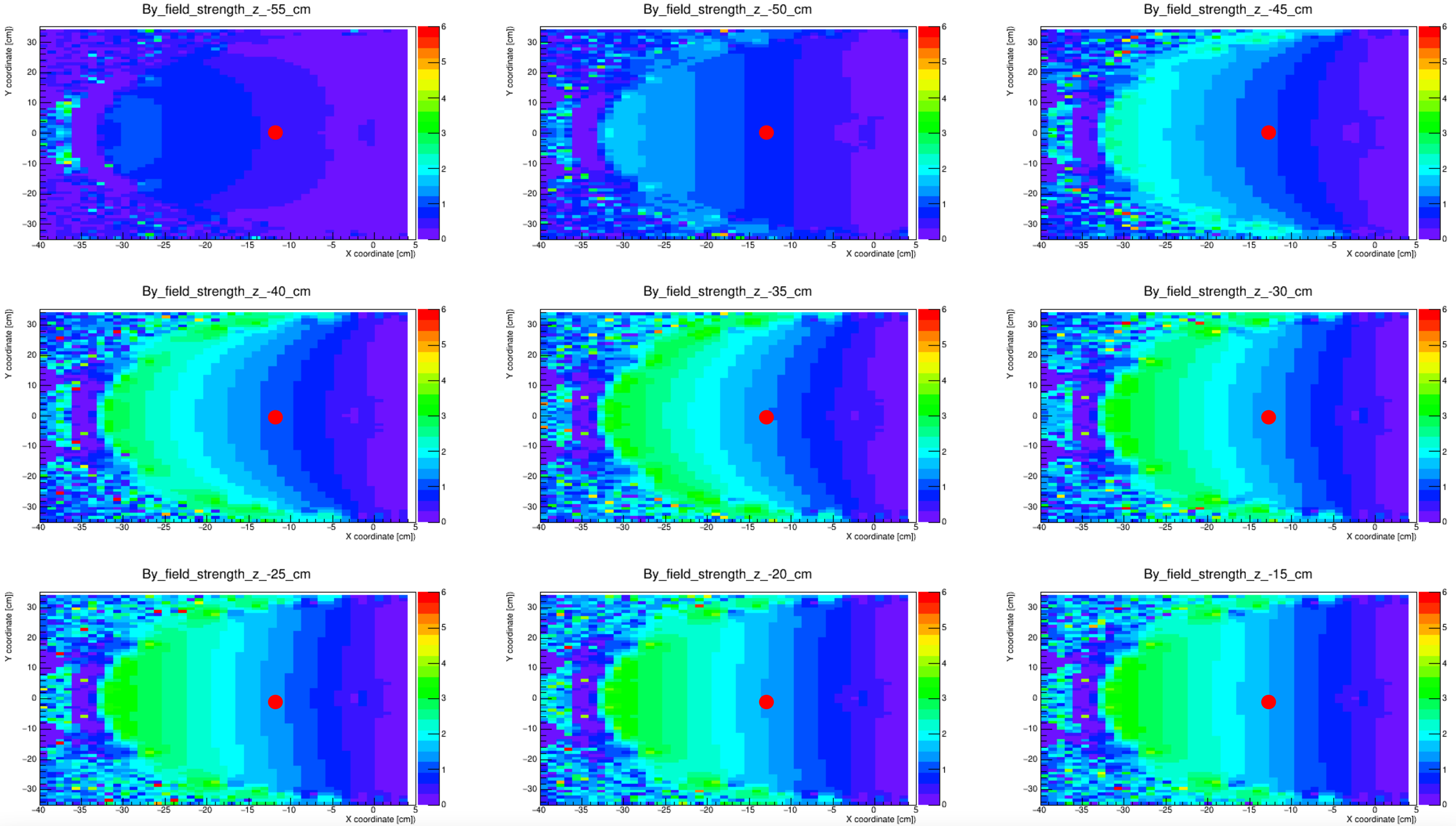
```
e-beam @ +34 mm and hadrons @ -126 mm
(neglecting 25 mrad angle) in this model

Z=0 is center of the magnet @ 6.4m
```

- X = -126 mm, y = 0 mm (before rotation and shift to fit along beamline).
  - By field strength along the hadron beam.
  - Gaussian field shape!

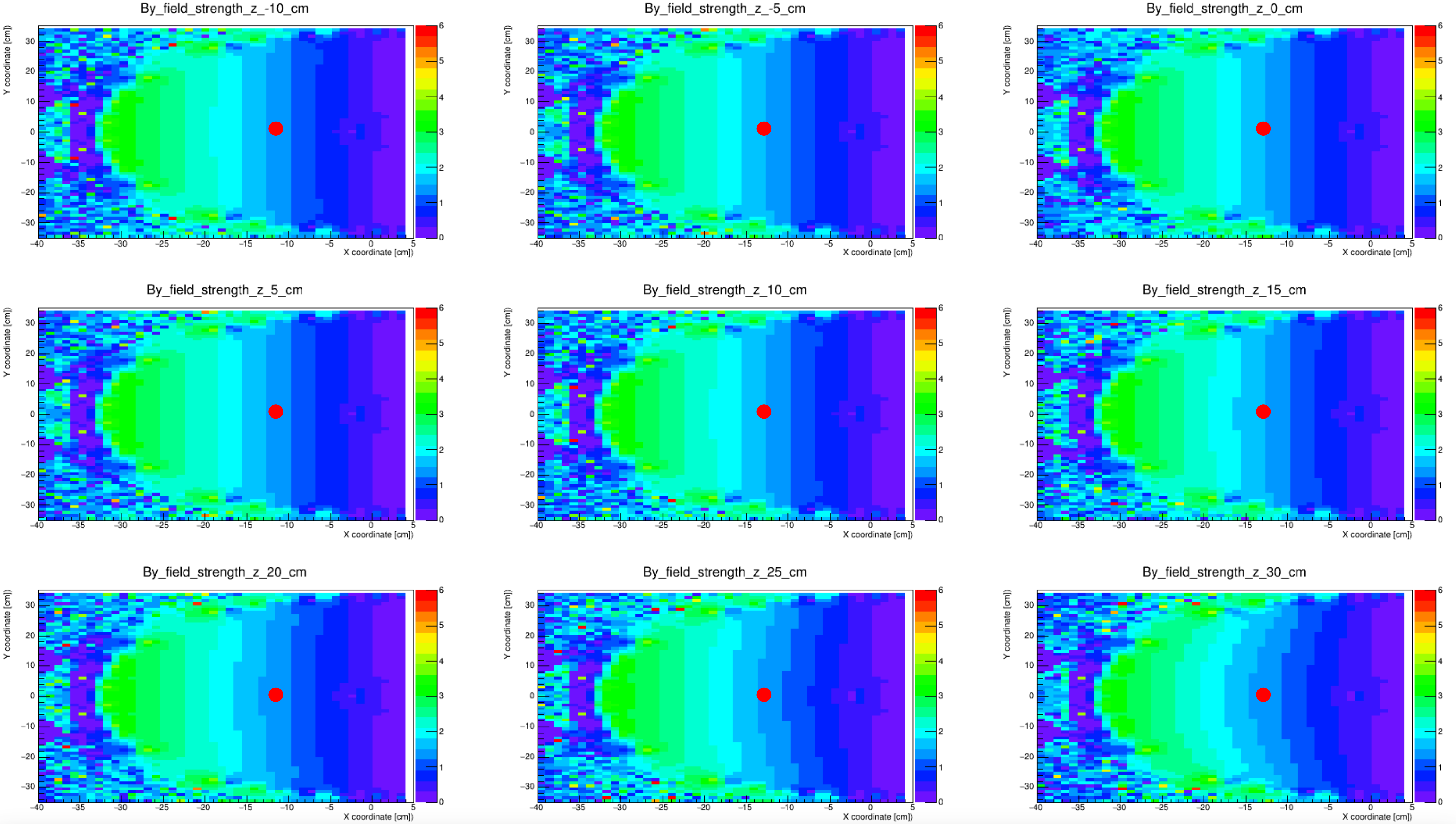
B0 Field Map

Left to right, top to bottom – increasing Z (IP to center of magnet)

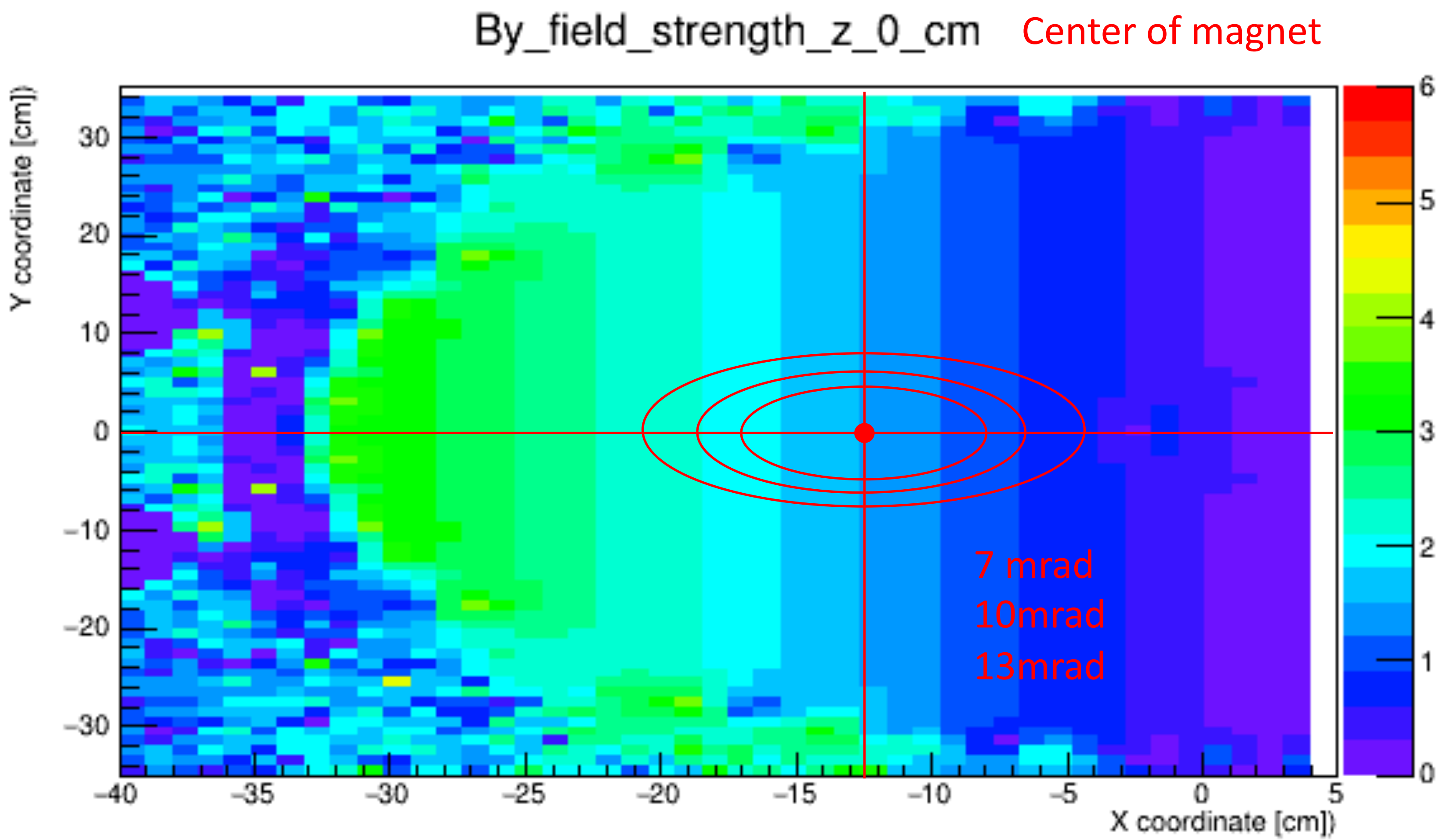


B0 Field Map

Left to right, top to bottom – increasing Z (IP to center of magnet)



B0 Field Map



- Field seen by protons is pT/angle dependent – will add an additional smearing dependence on position within the magnetic field.

B0 EMCAL and ZDC EMCAL

- Energy information is present, clustering is still simplistic/not present.
- Acceptances and interaction with material realistic, so doing a rudimentary clustering should be fairly simple.

```
TTreeReaderArray<float> b0_cluster_x = {tree_reader, "B0ECalClusters.position.x"};
TTreeReaderArray<float> b0_cluster_y = {tree_reader, "B0ECalClusters.position.y"};
TTreeReaderArray<float> b0_cluster_z = {tree_reader, "B0ECalClusters.position.z"};
TTreeReaderArray<float>  b0_cluster_energy = {tree_reader, "B0ECalClusters.energy"}; //deposited energy in cluster
```

ZDC HCAL

- SiPM-on-Tile HCAL HEXSPLIT available, but not clear if part of main repo.
- Material interactions are all what they should be, but reconstruction may still be a problem.
  - Following up with ZDC experts and will report back soon.

Setting up the environment (assuming Mac + Docker)

Very helpful tutorial page: https://eic.github.io/tutorial-jana2/aio/index.html

- Start eic-shell, and make sure to upgrade the container.

```
./eic-shell --upgrade
```

- Once in eic-shell, setup the environment.

```
source /opt/detectors/setup.sh
```

- Now, let's go ahead and make a test sample with the particle gun.

```
npsim --compactFile ${DETECTOR_PATH}/epic_ip6.xml  --steeringFile ddsim_steer_FF_v2.py
```

The filename inside the Python steering file is intentional – **naming it [name].edm4hep.root** puts the output into the PODIO format which EICrecon **requires.**

- **This will take a little more than an hour to run (100 events), and produce a ~100MB output file.**

Now for the EICrecon part

Very helpful tutorial page: https://eic.github.io/tutorial-jana2/aio/index.html

- **EICrecon part:**
  - Clone the EICrecon repo.
  - Now, perform the compilation.
  - Then, source your installation (assuming you're already in the EICrecon directory).
  - Now, you can run EICrecon with your test input file.
    - It will produce an output file with the name you specify, but if you don't supply an output file, it will produce a file with the default name of "podio_output.root".

```
git clone https://github.com/eic/EICrecon
```

```
cd EICrecon
cmake –S . –B build
cmake ––build build ––target install –– –j4
```

```
source ./bin/eicrecon–this.sh
```

```
eicrecon [file].edm4hep.root –
Ppodio:output_file=eicrecon_out.root
```

Want to do some analysis?

- I tried to make your life easy:
  - https://github.com/ajentsch/exclusive_PWG_analysis
- Some basic analysis code which can the EICRecon output and produce ROOT histograms.
  - You can of course add what you want.
  - This is not a repo for anyone to push to – it's just provided as a starting point for people needing a starting point.

Other Tasks Needing Attention

- Solve pass-through issues for event-level information to EICrecon (e.g. beam energy/species).
- Add/complete benchmarks for various final states for purposes of evaluating impacts of various changes to geometry/reconstruction.
- Solve remaining issues related to reconstruction, with real time feedback from users on needed information in output branches (e.g. ACTS reconstruction specifying which detector subsystem produced the track).
- Evaluate impacts of various backgrounds on specific observables and discuss mitigation strategies.
- Standardize some basic analysis tools for common observables (e.g. t-reconstruction), and create an analysis repository for them to expedite the starting of analyses for others in the future.

Discussion