中国科学院高能物理研究所
Institute of High Energy Physics
Chinese Academy of Sciences

Precise simulation of drift chamber in the CEPC experiment

Wenxing Fang (IHEP)
On behalf of CEPC drift chamber working group

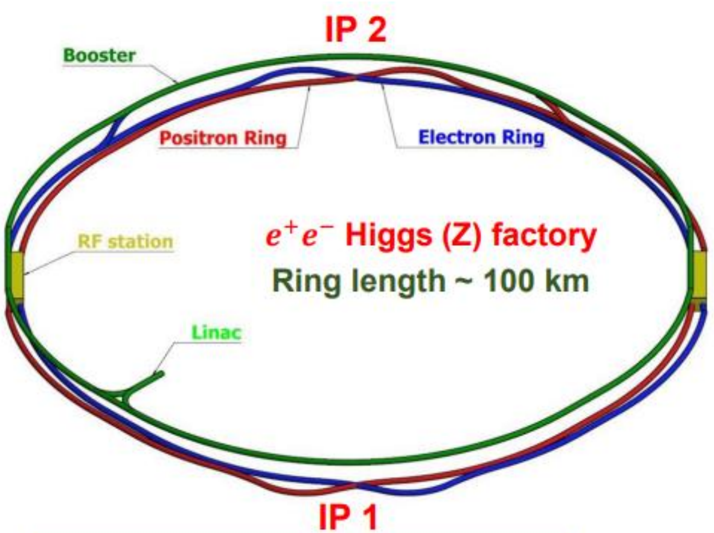
ACAT 2021

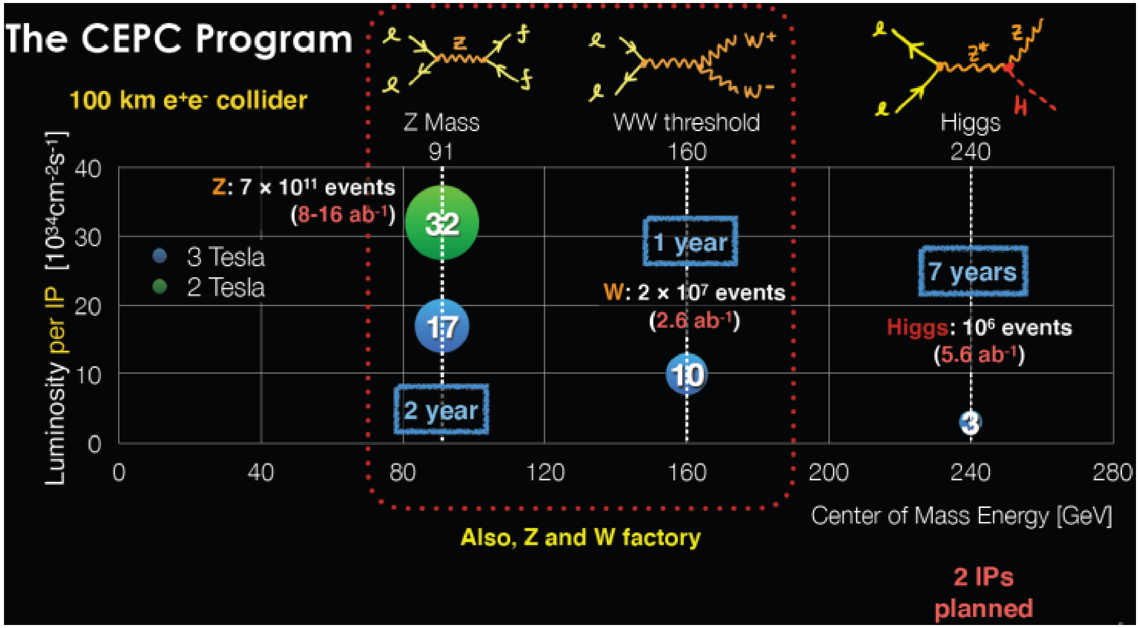
Outline

- Introduction:
  - CEPC experiment
  - CEPC 4^{th} concept detector design
  - dN/dx method
  - CEPC software
- Geant4 combined Garfield++ simulation
- Fast signal response simulation using ML
- Summary

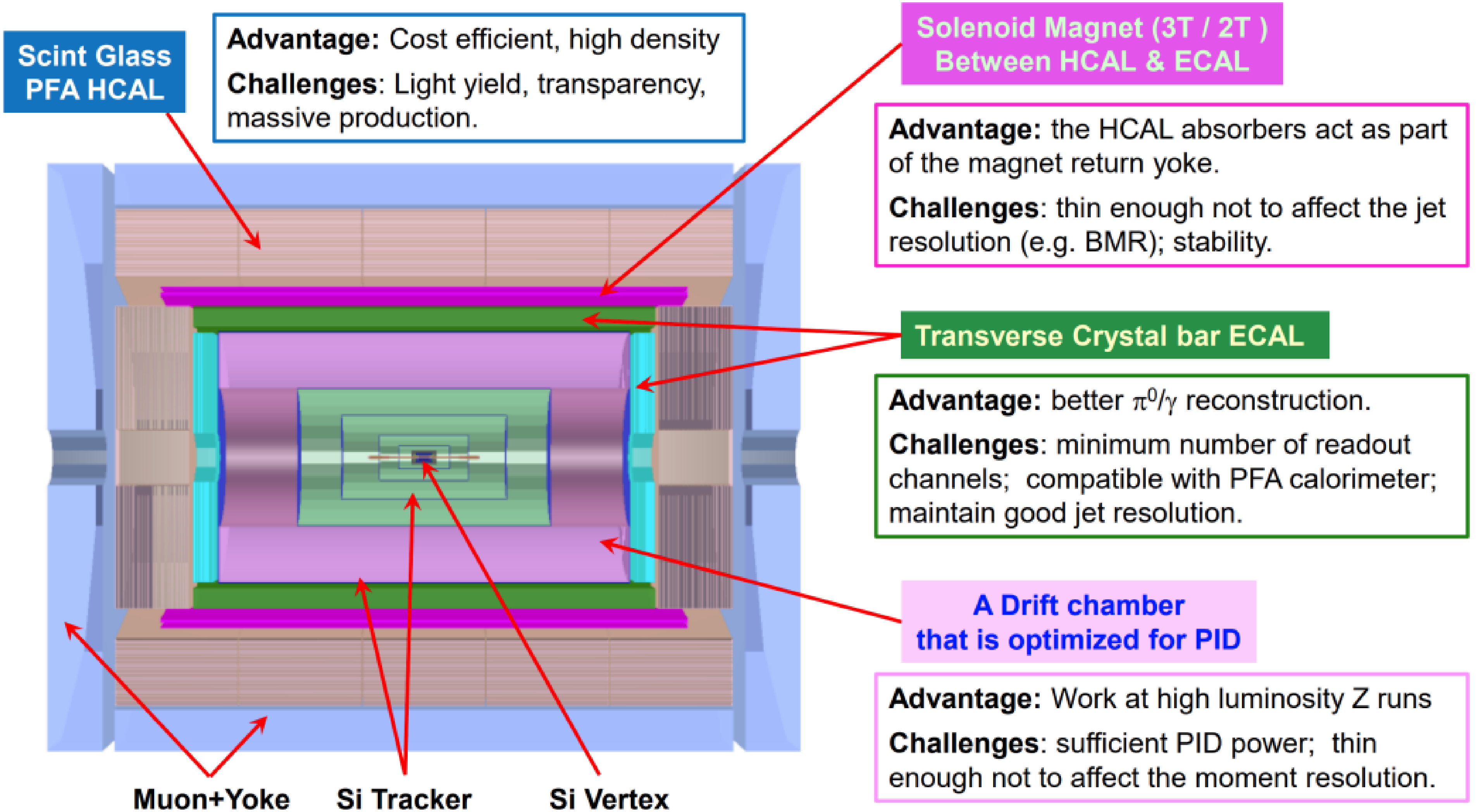
Circular Electron Positron Collider (CEPC)

- The CEPC is a Higgs(Z/W) factory in China
- ~1M Higgs at $\sqrt{s} = 240\ \mathrm{GeV}\ (\mathrm{ZH})$, ~Tera Z at Z pole and ~10M $\mathrm{W^+W^-}$ pair and possible $\mathrm{t\bar{t}}$ pair production
- Higgs, EW, flavor physics, BSM
- Possible Super pp Collider (SppC) of $\sqrt{s}$~100 TeV in the future





The 4^{th} Conceptual Detector Design



Advantage: Cost efficient, high density

Challenges: Light yield, transparency, massive production.

Advantage: the HCAL absorbers act as part of the magnet return yoke.

Challenges: thin enough not to affect the jet resolution (e.g. BMR); stability.

Advantage: better π^0/γ reconstruction.

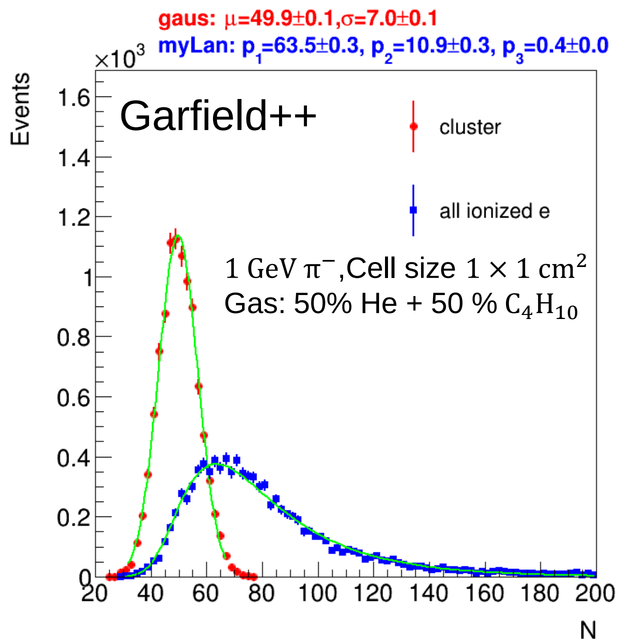
Challenges: minimum number of readout channels; compatible with PFA calorimeter; maintain good jet resolution.

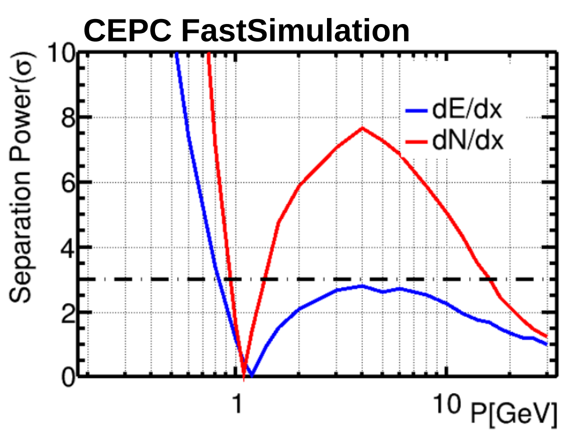
Advantage: Work at high luminosity Z runs

Challenges: sufficient PID power; thin enough not to affect the moment resolution.

Particle Identification with dE/dx and dN/dx

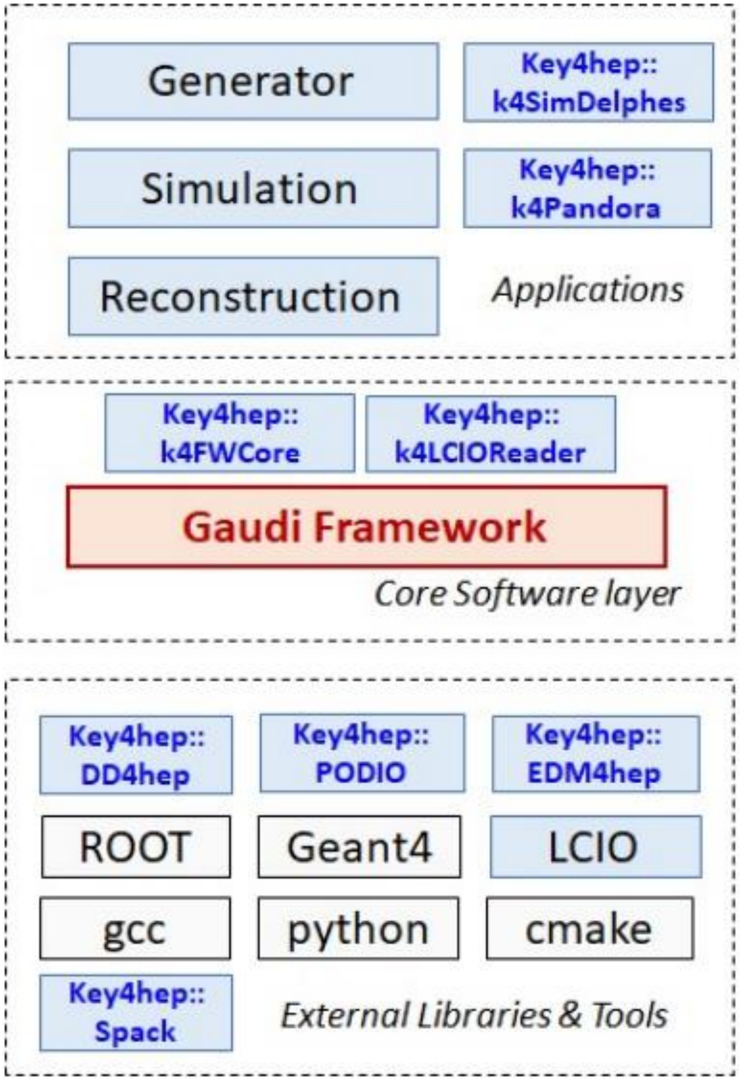
- Traditionally: using dE/dx
  - Due to the production of delta electron, the deposited energy follows Landau distribution
  - Using truncated mean to get the expected dE/dx will lost some measured information
  - Uncertainty from gas gain
  - Resolution is ~6%
- New technique: using dN/dx (primary ionization counting) method
  - The number of primary ionization follows Poisson distribution
  - Resolution could reaches <3%(arXiv:2105.07064)
  - Very promising results from fast simulation. In the end, the full simulation is needed to prove that



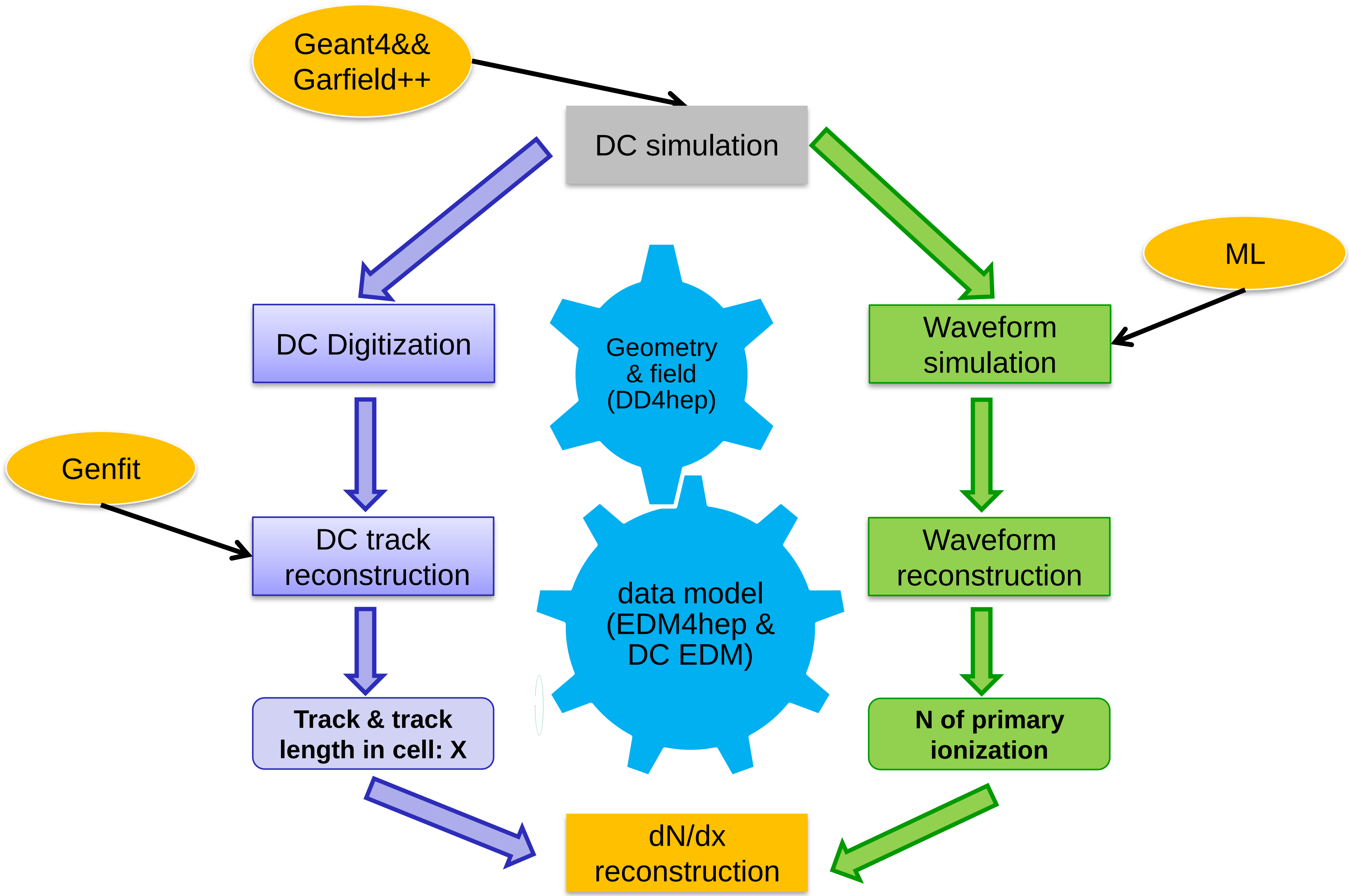


CEPCSW Core Software

- CEPCSW software structure
  - Core software
  - Applications: simulation, reconstruction and analysis
  - External libraries
- Core software
  - Gaudi/Gaudi Hive: defines interfaces to all software components and controls their execution
  - CEPC-specific framework software: generator, Gean4 simulation, beam background mixing, fast simulation, machine learning interface, etc.
  - EDM4hep:generic event data model for HEP experiments (see ? poster)
  - K4FWCore: manages the event data
  - DD4hep: geometry description, non-uniform B field



Drift Chamber Simulation and Reconstruction Flow

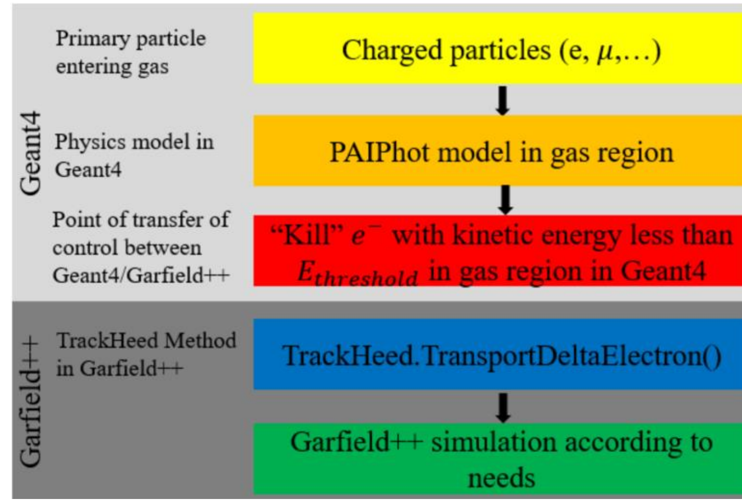


Drift Chamber Ionization Simulation

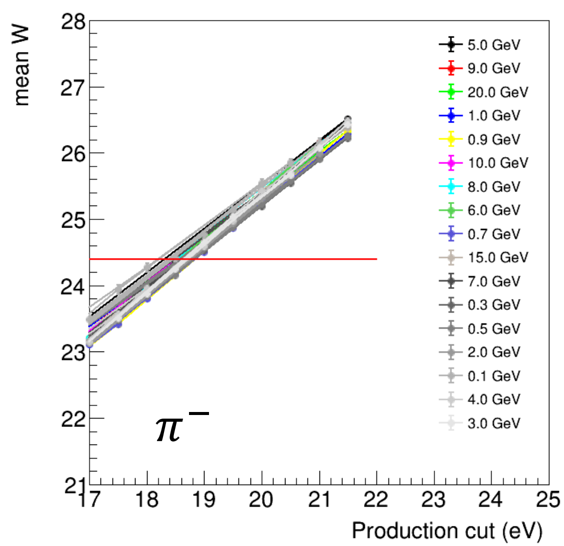
- As we know Geant4 can not simulate the ionization process properly (arXiv:2105.07064)
- Garfield++ is commonly used for precise ionization simulation for simple geometry
- In order to do a detailed drift chamber simulation, including particle interaction with detector materials, ionization in gas, drift and avalanche processes in drift chamber cell, combining Geant4 and Garfield++ is needed
- This paper "Interfacing Geant4, Garfield++ and Degrad for the Simulation of Gaseous Detectors" introduces some ways to combine Geant4 and Garfield++ to get correct energy deposition or total number of ionized electrons (adopted by COMET experiment)
- However, it can not give both correct number of primary ionization and total number of ionized electrons (see next slide)

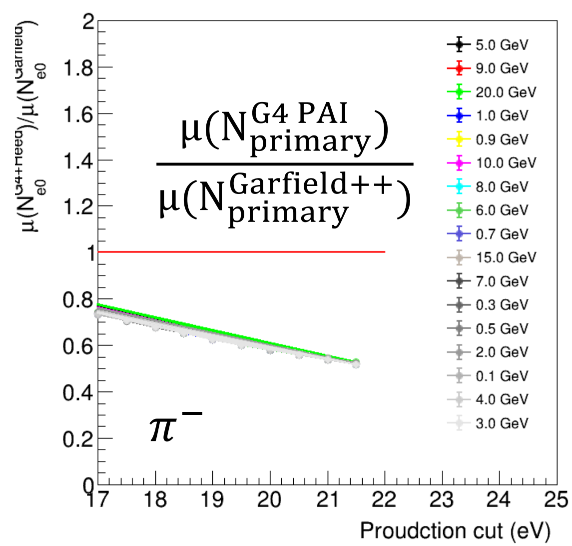
Drift Chamber Ionization Simulation by G4 PAI

- Method:
  - Geant4 PAI model to simulate primary or secondary ionization
  - TrackHeed (from Garfield++) to simulate ionization from residual delta electron



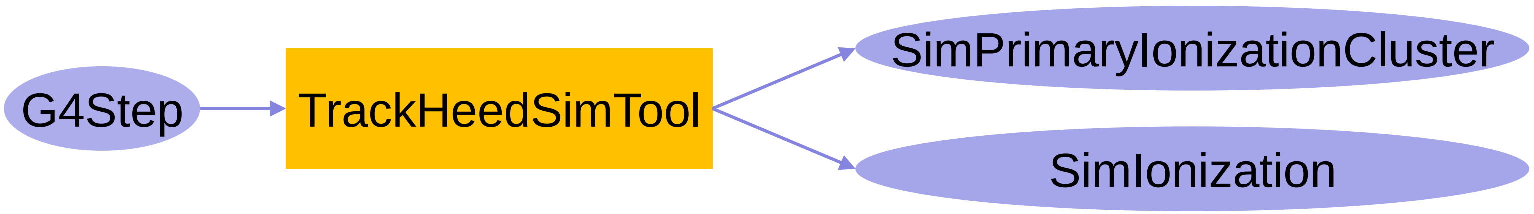
- It was found that the primary ionization produced by this method is much less than Garfield++
- Confirmed with authors





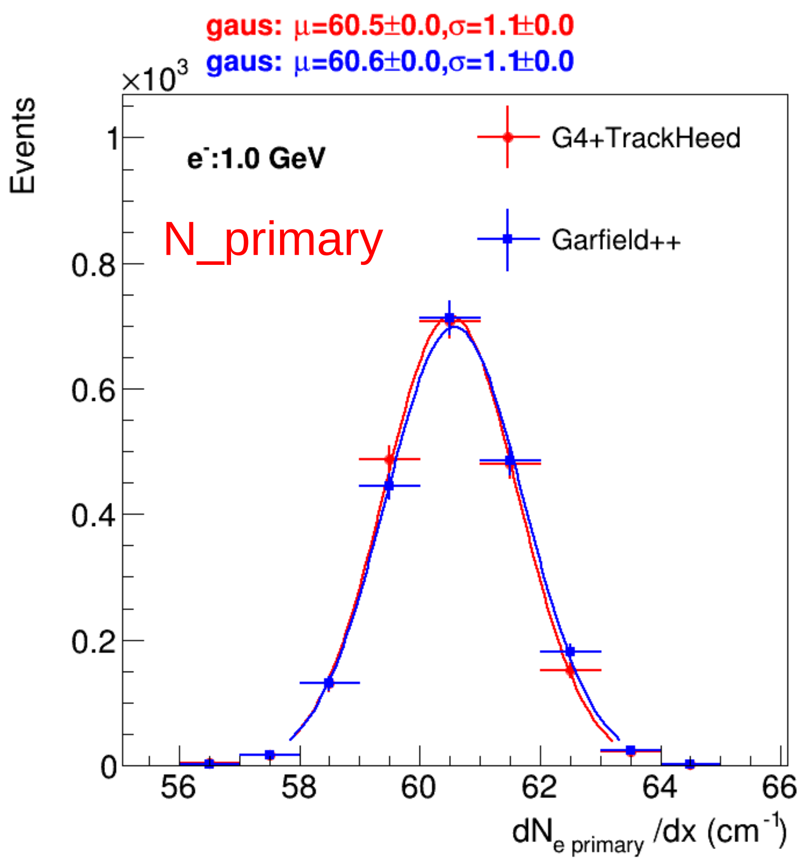
Proposed Drift Chamber Ionization Simulation

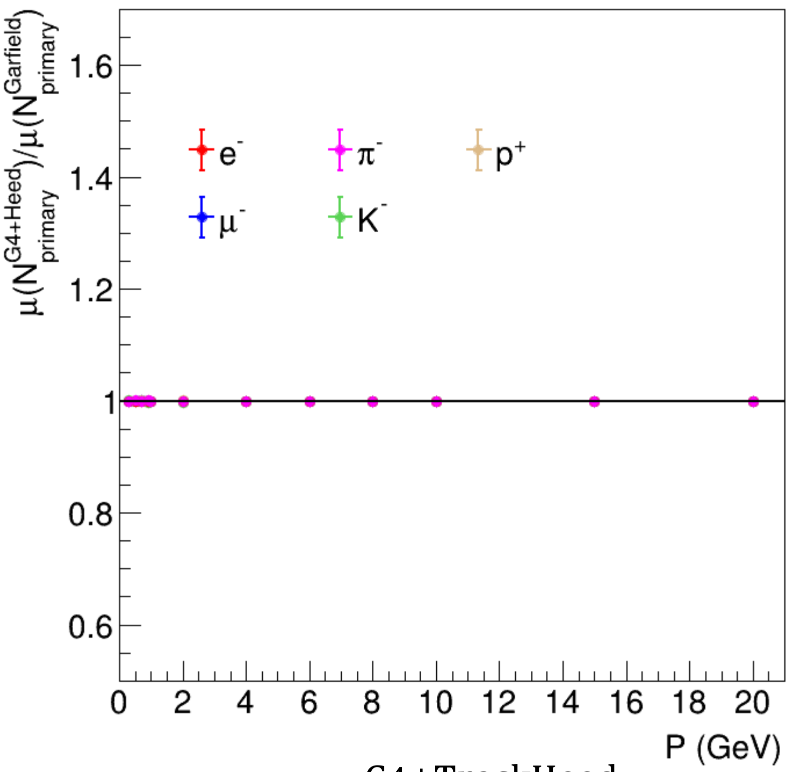
- Combining Geant4 and Garfield++ at G4Step level
- TrackHeedSimTool (Gaudi tool) is created for this task
  - Input: G4Step information (particle type, initial position and momenta, ionization path length)
  - Use TrackHeed (used by Garfield++) to simulate one step length (or multi-step length for speed up) ionization (new API contributed to Garfield++ PR)
  - Output: primary and total ionization information (contains position, time, cell id), saved in EDM
  - The kinetic energy of G4Track will be updated according to the energy loss in the ionization
  - Non-uniform magnetic field can be handled easily



Ionization Simulation Performance Check

- Gas: 50% He + 50 % C_4H_{10}



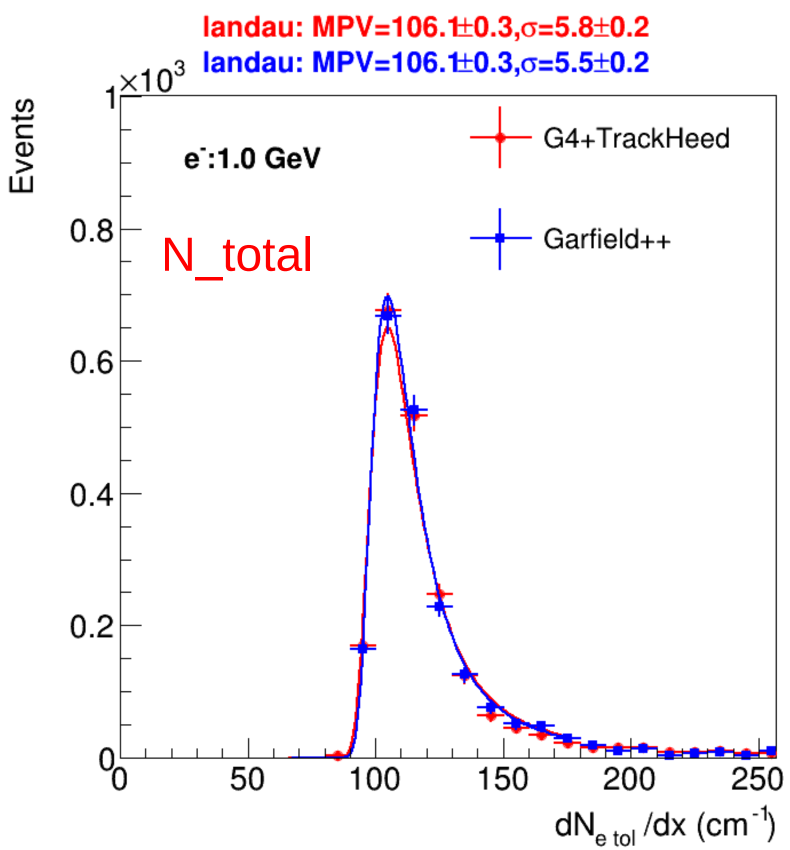


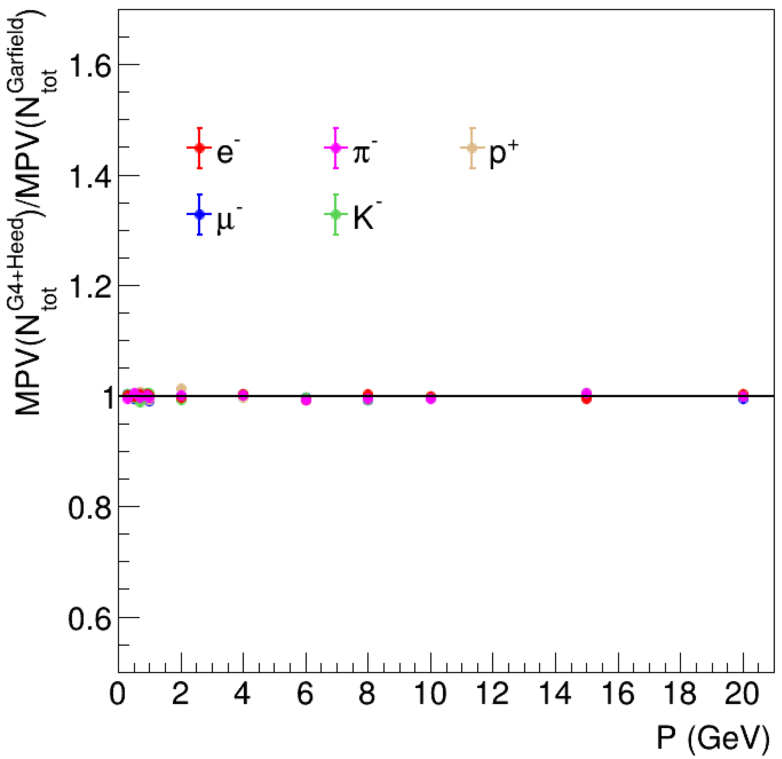
$$\frac{\mu(N_{\text{primary}}^{\text{G4+TrackHeed}})}{\mu(N_{\text{primary}}^{\text{Garfield++}})}$$

Consistent with Garfield++ standalone simulation results, see backup for more details

Ionization Simulation Performance Check

- Gas: 50% He + 50 % C_4H_{10}



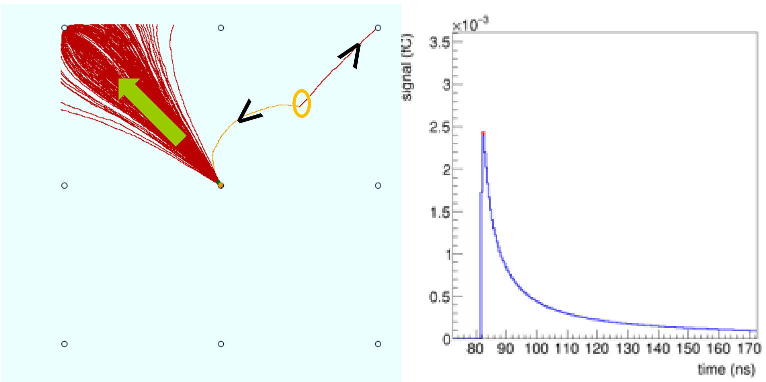


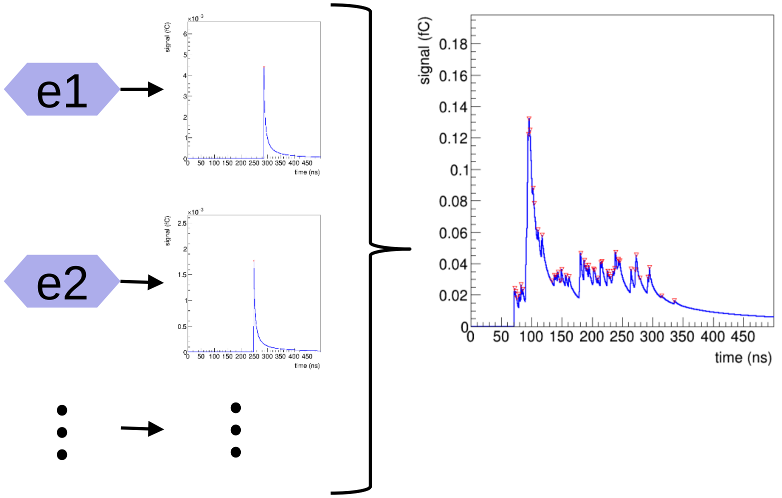
$$\frac{\mathrm{MPV}(\mathrm{N}_{\mathrm{tot}}^{\mathrm{G4+TrackHeed}})}{\mathrm{MPV}(\mathrm{N}_{\mathrm{tot}}^{\mathrm{Garfield++}})}$$

Consistent with Garfield++ standalone simulation results, see backup for more details

Signal response simulation

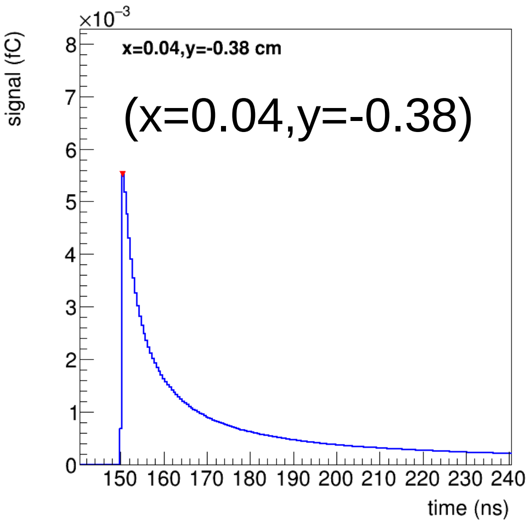
- Simulating the drift and avalanche processes of ionized electron and ions
- Using Garfield++: very precise but extremely time consuming, could take $\mathcal{O}(1)$ to $\mathcal{O}(10)$ seconds just for one electron
- Going to use parameterization (fast simulation) method, parameters are based on Garfield++ simulation results
  - For each electron, simulate its own pulse
  - As done by Garfield++, piling up all pulses from same drift chamber cell gives final signal response

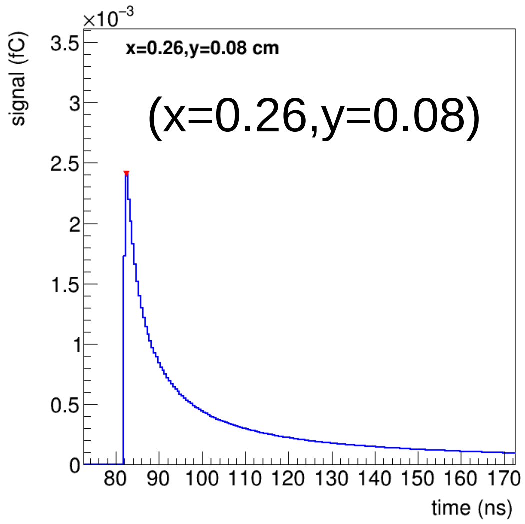


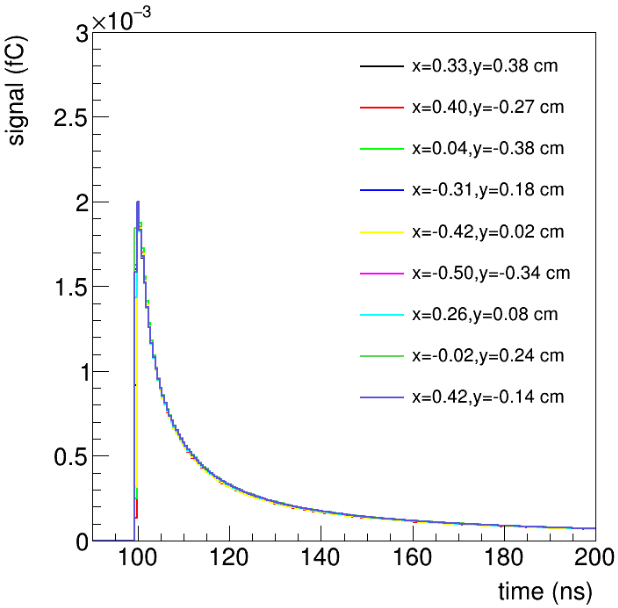


Single electron pulse study

- Performed single electron simulations using Garfield++
- All single electron pulses are similar after normalization. For example, if its peak position is shifted to some value (e.g. 100 ns) and its peak value is scaled to some value (e.g. 2×10^{-3})

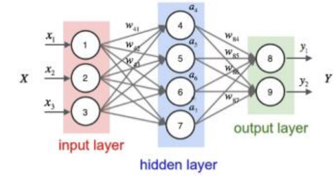






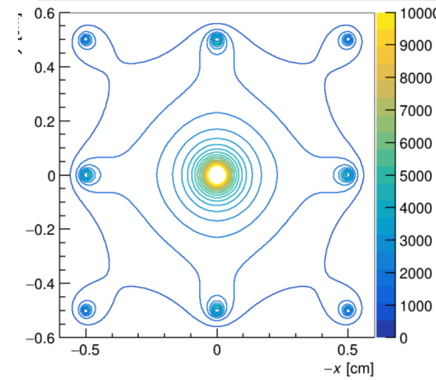
- Simulating single electron pulse ≈ simulating peak_time and peak_value of the pulse + using pulse template

Dataset and ML Model



- Garfield++ simulation events:
  - 350k single electron event with electron position uniformly distributed $1 \times 1\ \mathrm{cm}^2$ drift chamber cell
  - Gas: 50% He + 50 % C_4H_{10}
  - Center signal wire (2000 V), eight field wires (0 V)
- Model: fully connected neutral network
  - Consist of input, hidden, and output layers
  - Input: Local x and y positions of ionized electrons,N(0,1) distribution noise
  - Output: peak time and value of single electron pulse
- Loss: a differentiable two sample test statistics based on smoothed k-nearest neighbor tests (arXiv:1709.01006) between real data and generated data

Electric field contour plot



Input = x, y, noise ~ N(0, 1)

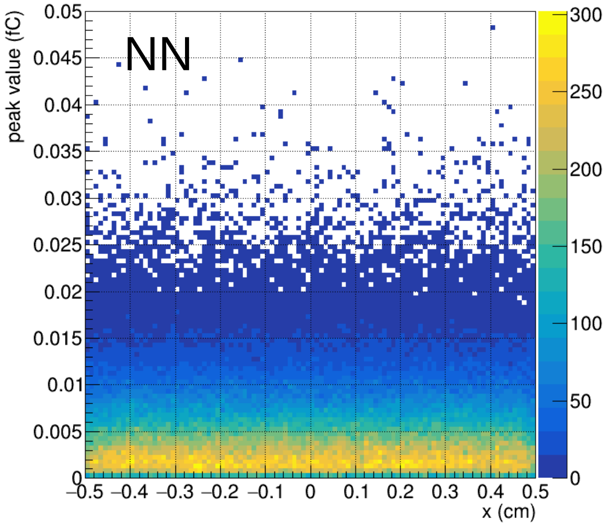
Dense(3, 814, activation='relu')

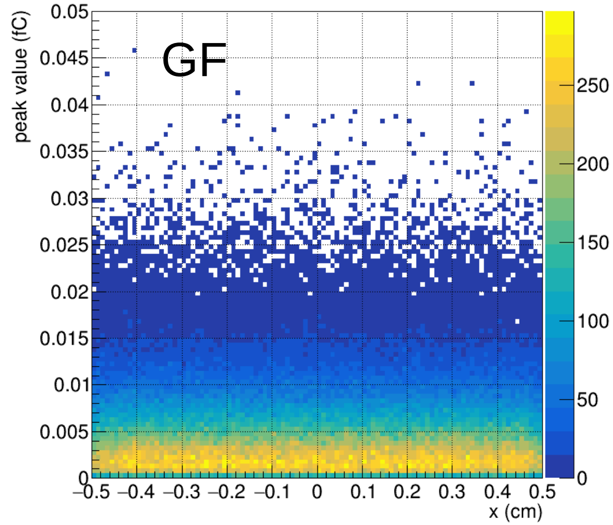
Dense(814, 814, activation='relu')

gen(time, value)=Dense(814,2)

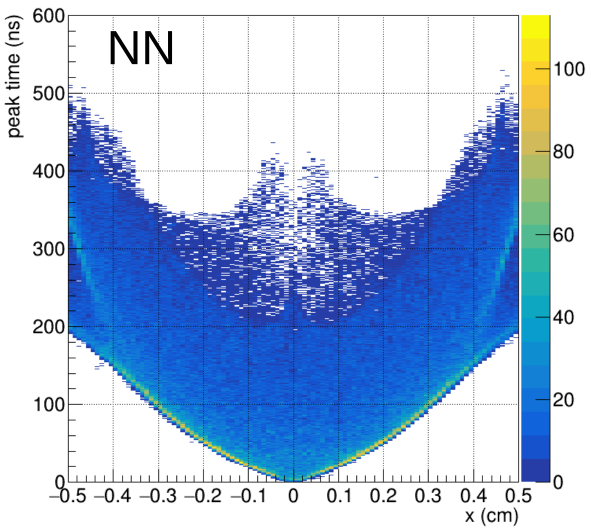
Mini-batch training: using [Faiss]() (fast similar search) to get a batch (1024) of closet data (L2 distance)

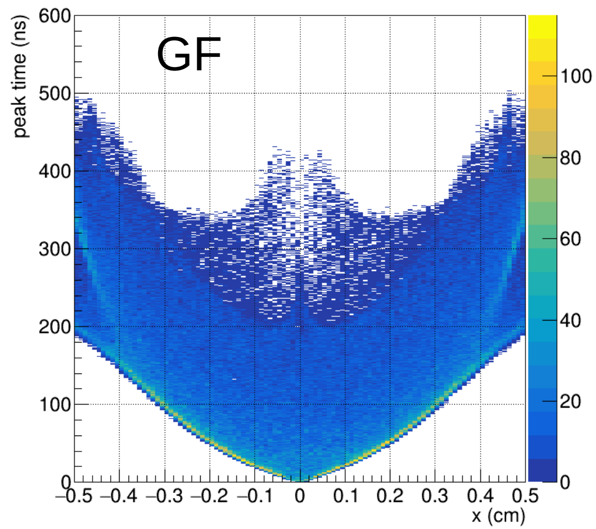
Simulation performance (2D distribution)





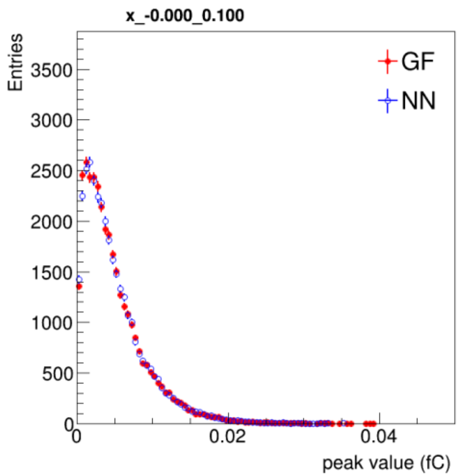
Peak value vs x

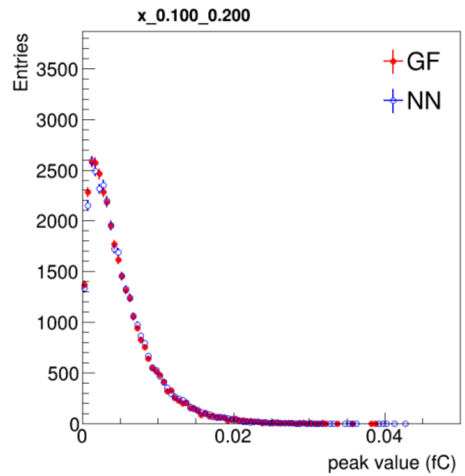


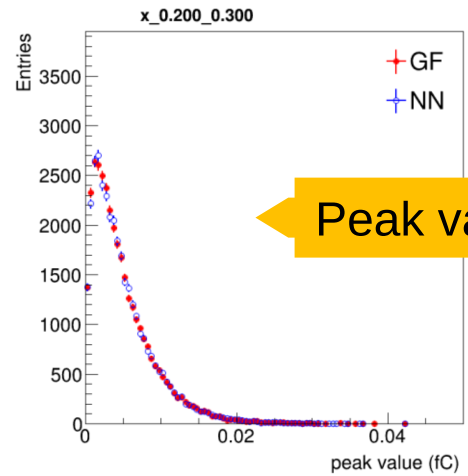


Peak time vs x

One Dimension Check





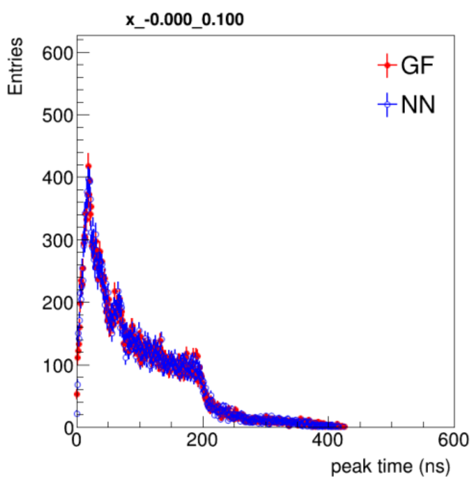


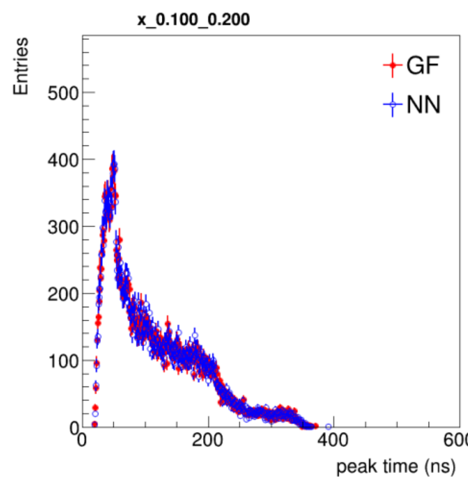
Peak value

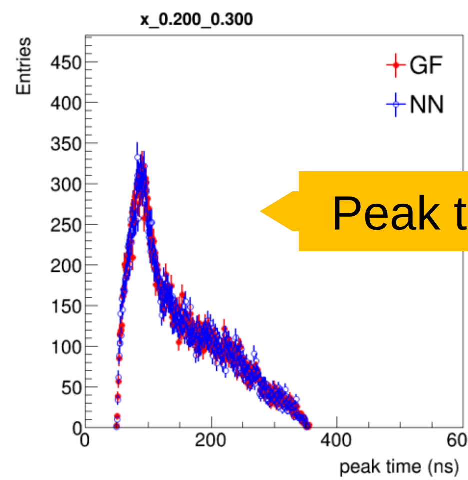
x(0,0.1) cm

x(0.1,0.2) cm

x(0.2,0.3) cm



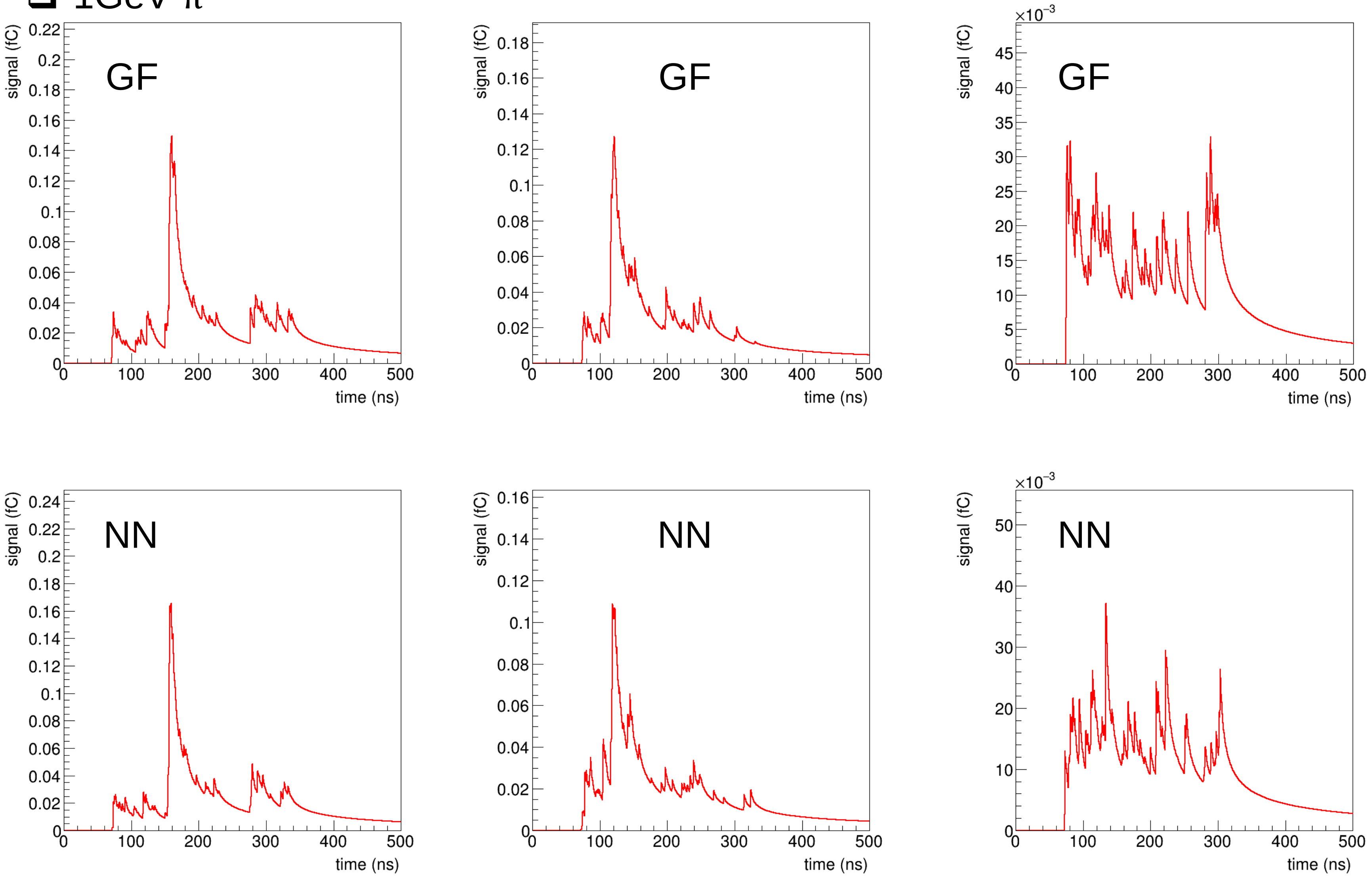




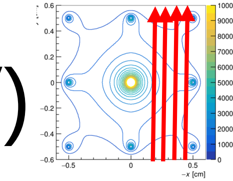
Peak time

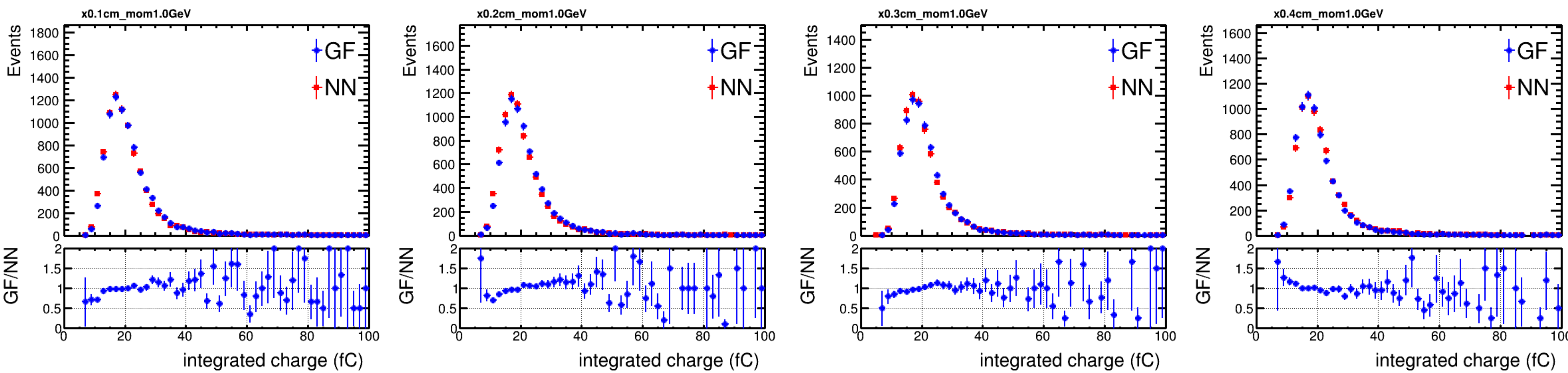
Display of signal response

- ❑ 1GeV π^-



Performance check: electron (1 GeV)



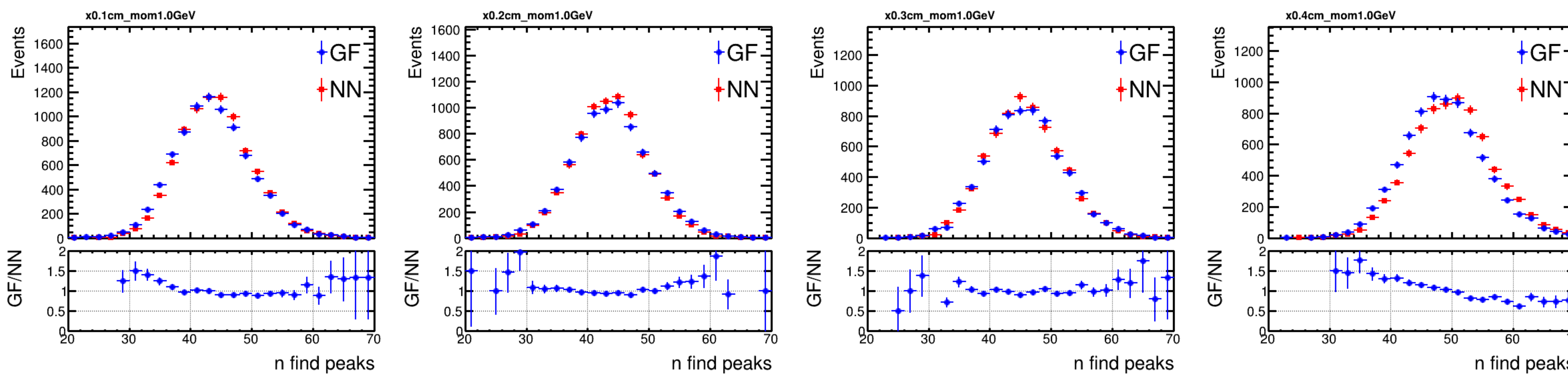


x=0.1 cm

x=0.2 cm

x=0.3 cm

x=0.4 cm



- Checked total integrated charge and number of found peaks(using scipy.signal.find_peaks)
- Good agreement in general (a little bias for number of found peaks for x=0.4 cm. Could be improved in future), Similar results for other energy points (0.5, 5, and 10 GeV)

Method advantages

- Average time for one cell $(1 \times 1\mathrm{cm}^2)$ simulation for 1 GeV π^- (gas: $50\ \mathrm{He} + 50 \mathrm{C_4H_{10}}$) :
  - Garfield++: ~250 s
  - NN: ~1 s
- This simulation algorithm is general and applicable for different particles. As for different particles, only the ionization part is different, the signal response simulation keeps the same
- By this way, pulse simulation is not related to Geant4 and it is independent between each electron. To further speed up the signal response simulation, GPU or multithreading technique can be easily used

Summary

- A simulation scheme for drift chamber in CEPC experiment has been presented
- The ionization simulation using Geant4 combined with TrackHeed have been described in detail. Results are consistent with Garfield++ simulation
- In order to speed up the simulation of signal response, a fast simulation method using ML has been developed, which gives good agreement with Garfield++ simulation results, and more than 200 times speed up can be achieved

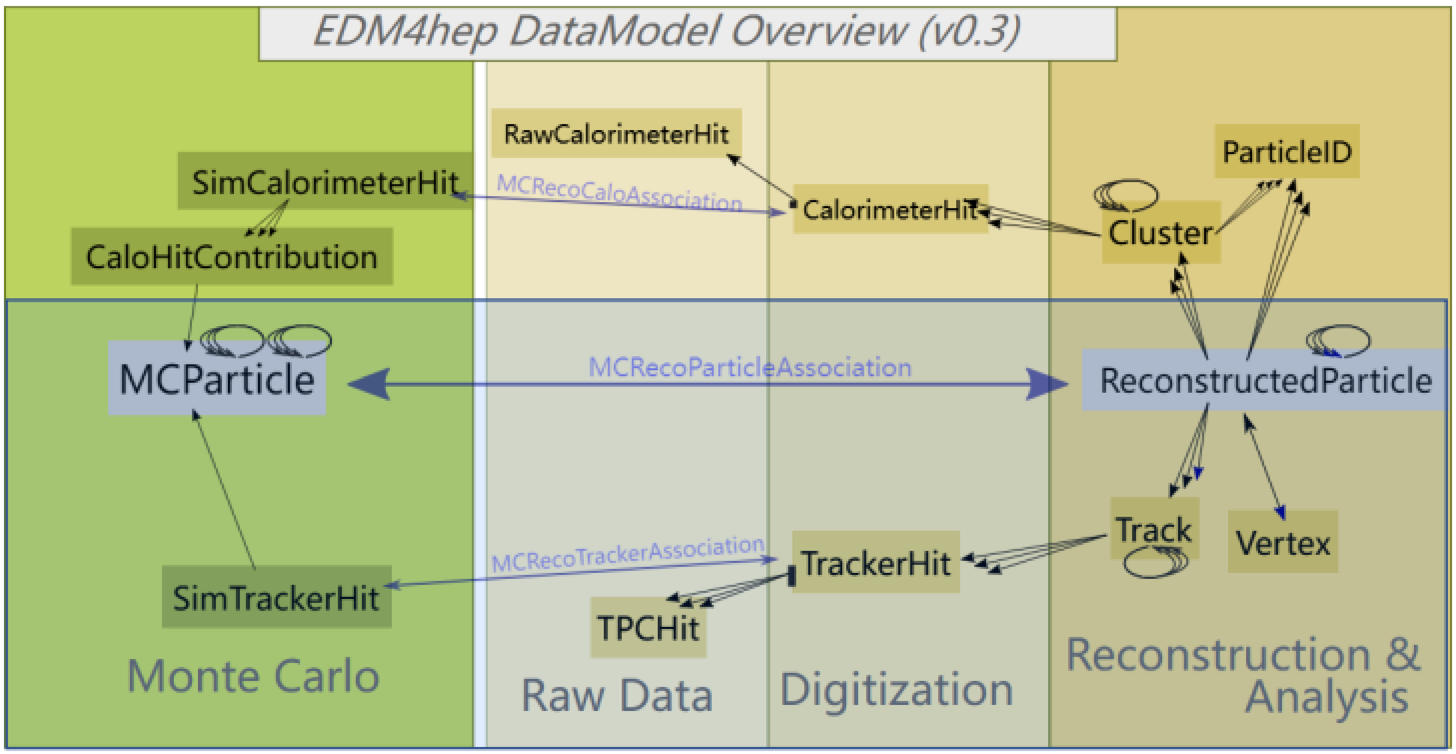
Thanks for your attention !

Acknowledgement

- Thanks EDM4hep and key4hep working groups

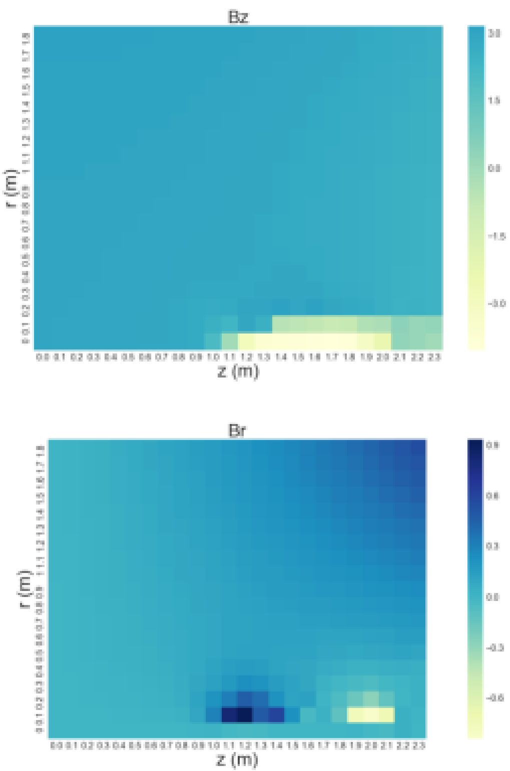
Back up

Event Data Model

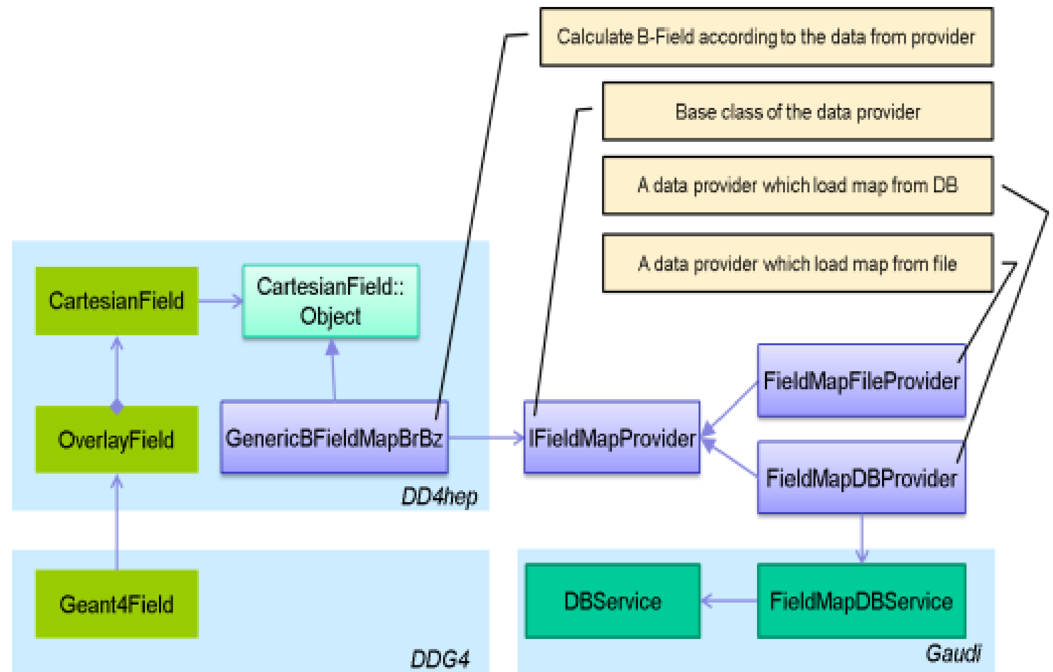


Non-uniform B-Field

- A generic B-field service is developed and integrated with DD4hep
  - CSV-like format data from magnetic group
  - Bz=3Tesla, in DC region non-uniformity<5% in z direction and <55% in radial



Filed map

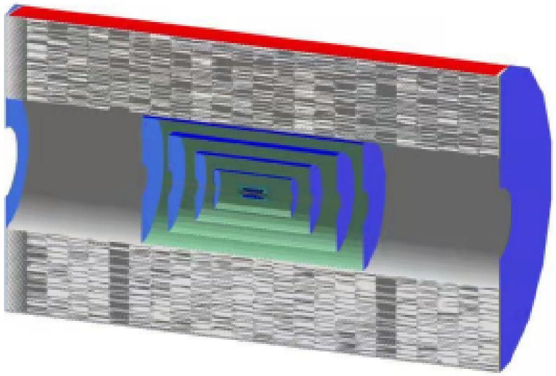


B-field service in CEPCSW

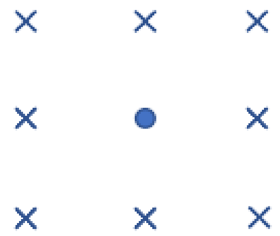
Drift Chamber Parameters in CEPCSW

- The base line configuration of DC in CEPCSW

Half length	2980 *mm*
Inner and outer radius	800 to 1800 *mm*
# of Layers	100
Cell size	~9.6 *mm x 9.6 mm*
Gas	He:C_4H_{10}=90:10
Single cell resolution	0.11 *mm*
Sense to field wire ratio	1:3
Total # of sense wire	81631
Stereo angle	1.64~3.64 *deg*
Sense wire	Gold plated Tungsten ϕ=0.02*mm*
Field wire	Silver plated Aluminum ϕ=0.04*mm*
Walls	Carbon fiber 0.2 *mm*(inner) and 2.8 *mm*(outer)



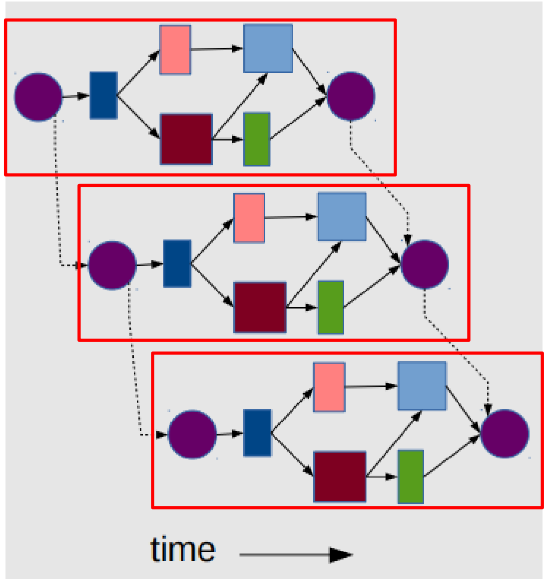
CRD tracker o1 v01



field wire: x, sense wire: o
Cell structure

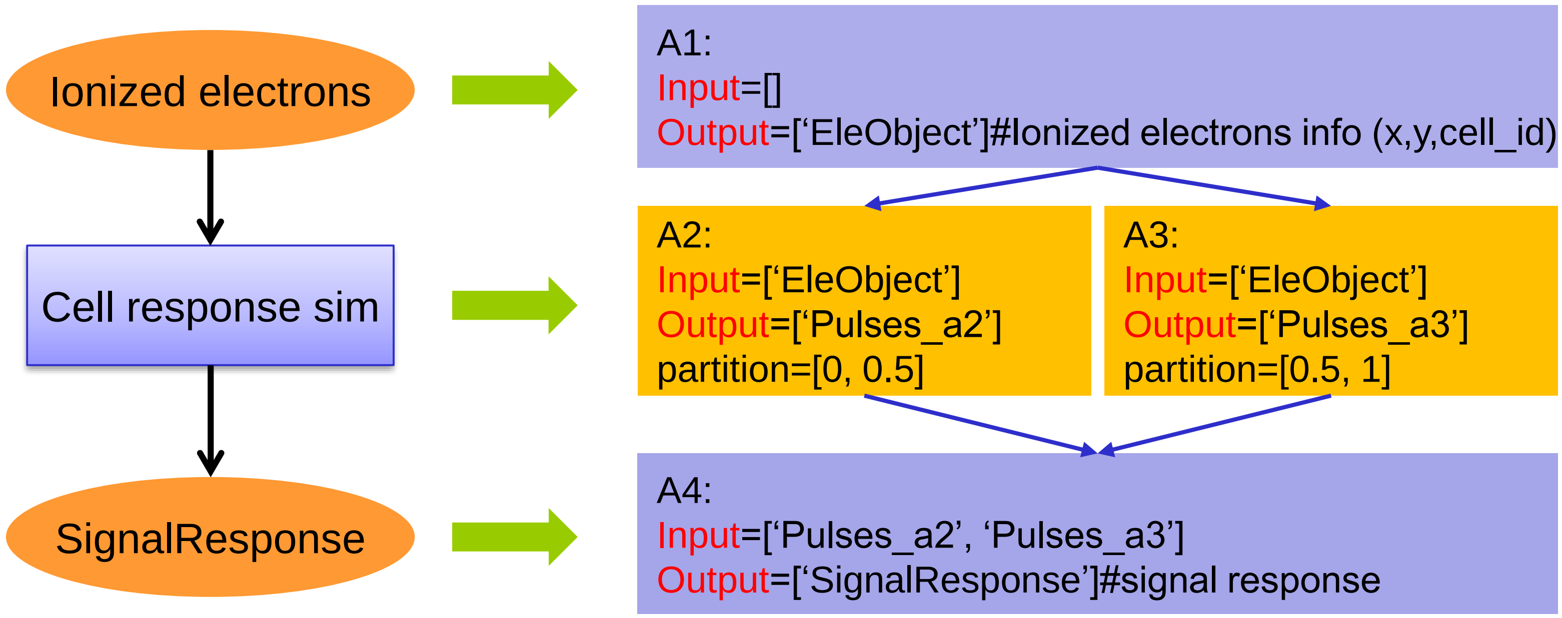
Gaudi Hive

- Gaudi Hive: multi-threaded, concurrent extension to Gaudi
- Data Flow driven mechanism
  - Algorithms declare their data dependencies
    - build a directed acyclic graph - can be used for optimal scheduling
  - Scheduler automatically executes Algorithms as data becomes available
- Algorithms process events in their own thread
- Multiple algorithms and events can be executed simultaneously
- Algorithm Cloning
  - Multiple instances of the same Algorithm can exist, and be executed concurrently, each for different event



Example using dummy data object

- Performing the study using dummy data object



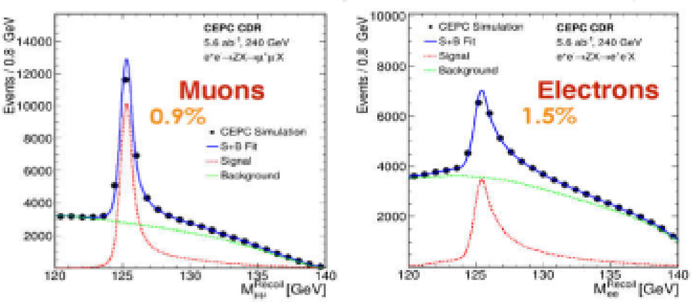
- Working well

Physics requirement

Detector performance requirements in CDR

Physics process	Measurands	Detector subsystem	Performance requirement
$ZH, Z \to e^+e^-, \mu^+\mu^-$ $H \to \mu^+\mu^-$	$m_H, \sigma(ZH)$ $\mathrm{BR}(H \to \mu^+\mu^-)$	Tracker	$\Delta(1/p_T) =$ $2 \times 10^{-5} \oplus \frac{0.001}{p(\mathrm{GeV}) \sin^{3/2}\theta}$
$H \to b\bar{b}/c\bar{c}/gg$	$\mathrm{BR}(H \to b\bar{b}/c\bar{c}/gg)$	Vertex	$\sigma_{r\phi} =$ $5 \oplus \frac{10}{p(\mathrm{GeV}) \times \sin^{3/2}\theta} (\mu\mathrm{m})$
$H \to q\bar{q}, WW^*, ZZ^*$	$\mathrm{BR}(H \to q\bar{q}, WW^*, ZZ^*)$	ECAL HCAL	$\sigma_E^{\mathrm{jet}}/E =$ $3 \sim 4\%$ at 100 GeV
$H \to \gamma\gamma$	$\mathrm{BR}(H \to \gamma\gamma)$	ECAL	$\Delta E/E =$ $\frac{0.20}{\sqrt{E(\mathrm{GeV})}} \oplus 0.01$

Good EM energy resolution is required for bremsstrahlung radiation recovery

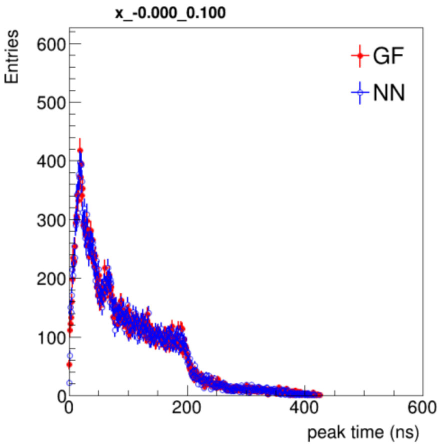


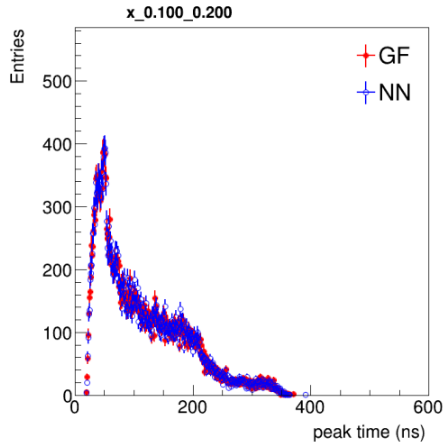
ee → HZ(→ee/μμ), recoil mass against Z boson

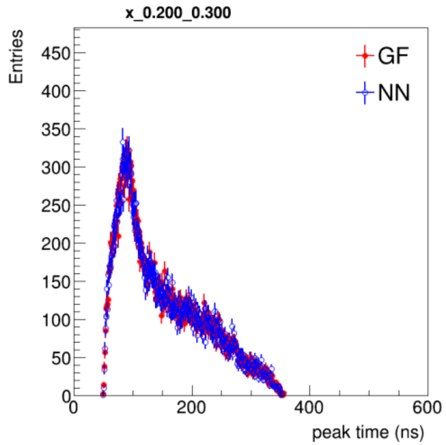
EWK physics:
Precise EM measurement

Flavor physics:
Precise EM measurement
Dedicated hadron identification

1 dimension check



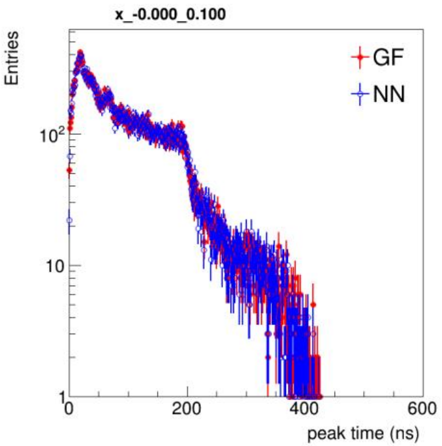


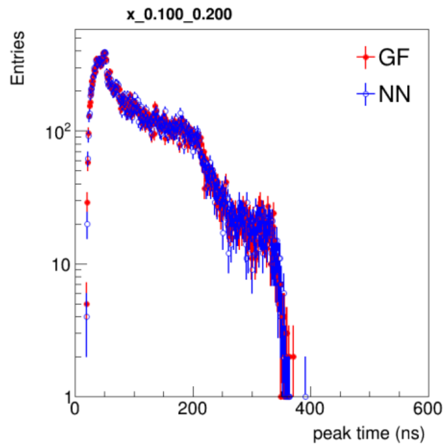


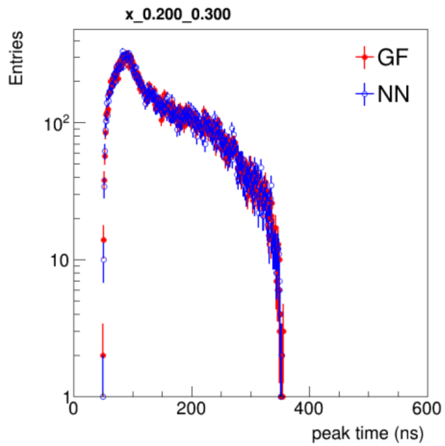
x(0,0.1) cm

x(0.1,0.2) cm

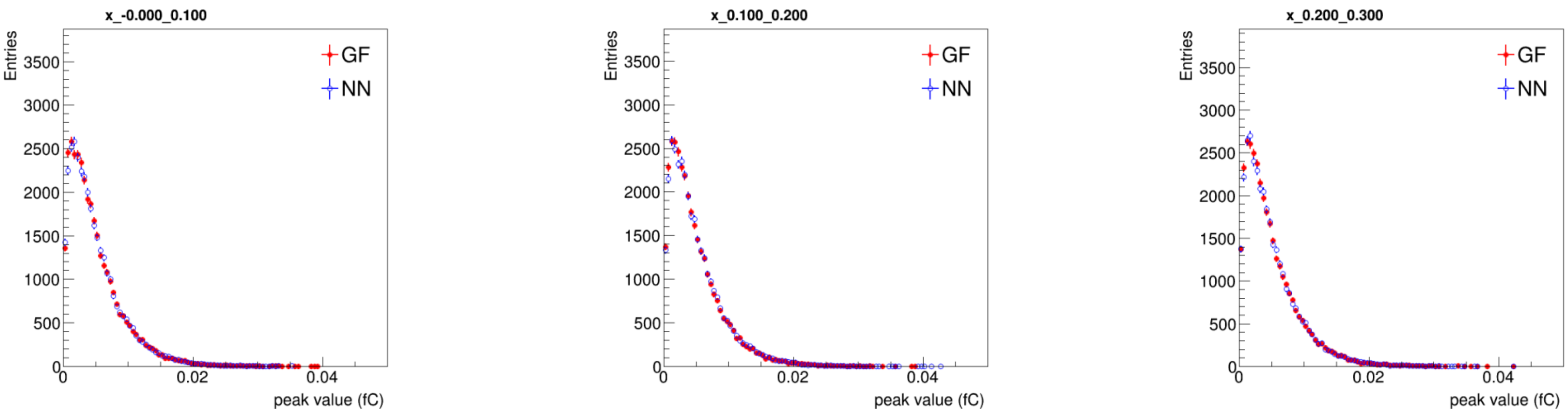
x(0.2,0.3) cm







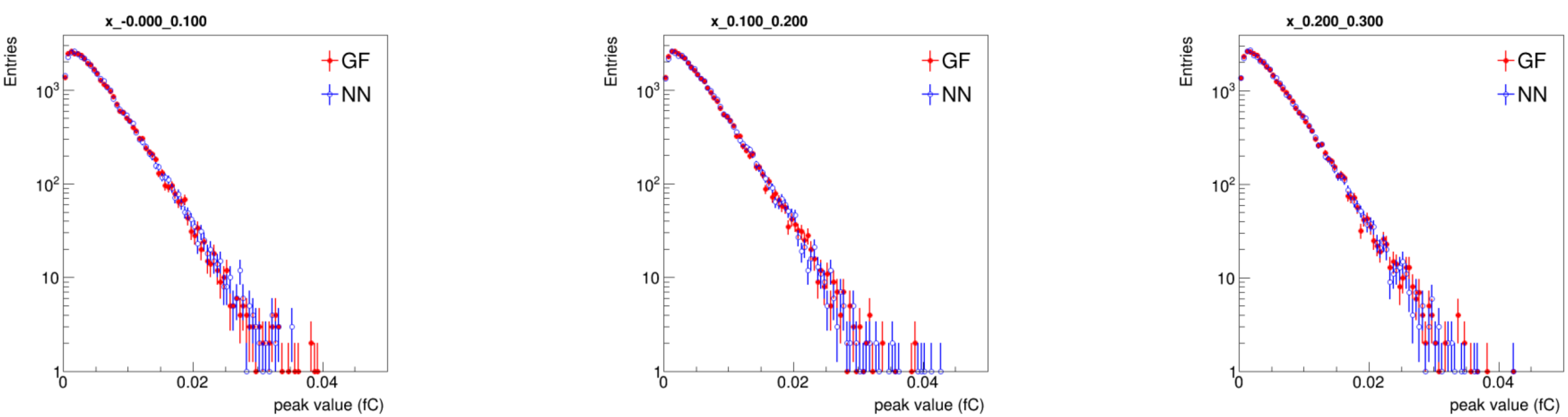
Peak value simulation



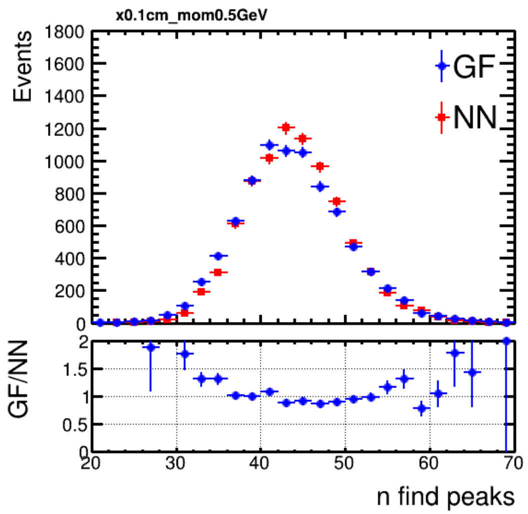
x(0,0.1) cm

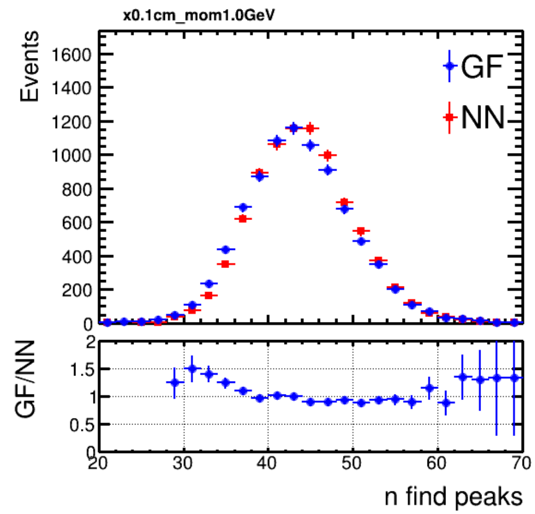
x(0.1,0.2) cm

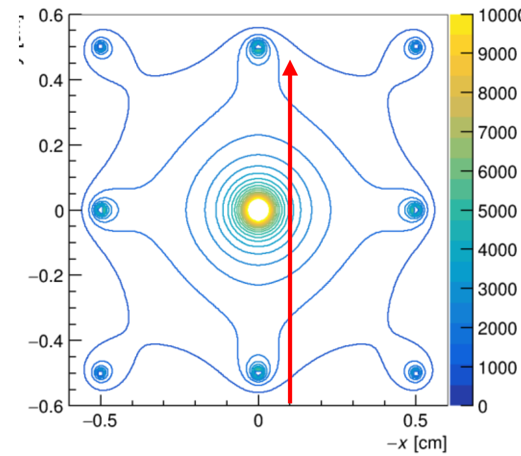
x(0.2,0.3) cm



Performance check: electron (x=0.1 cm)

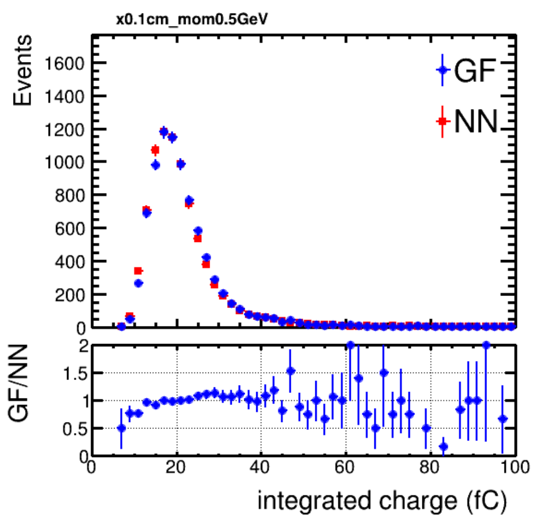


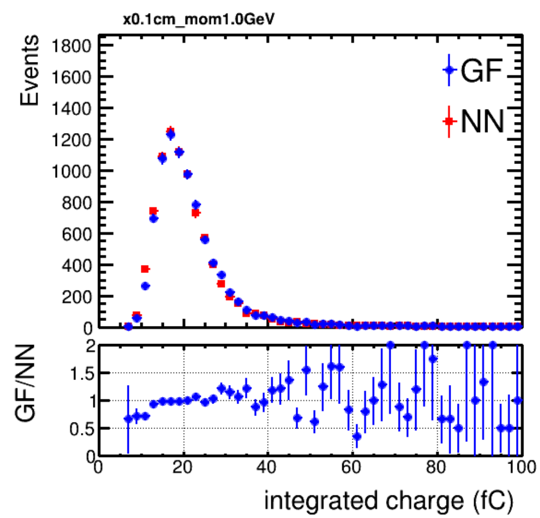




0.5 GeV

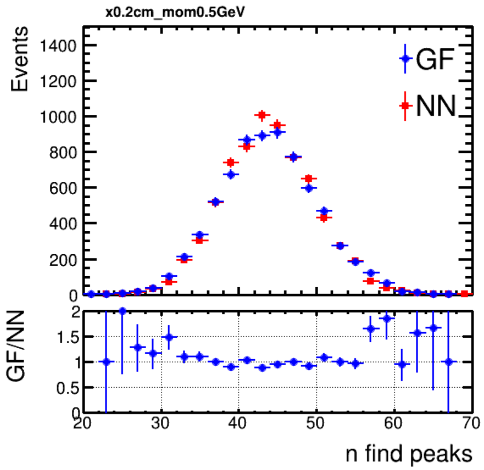
1 GeV

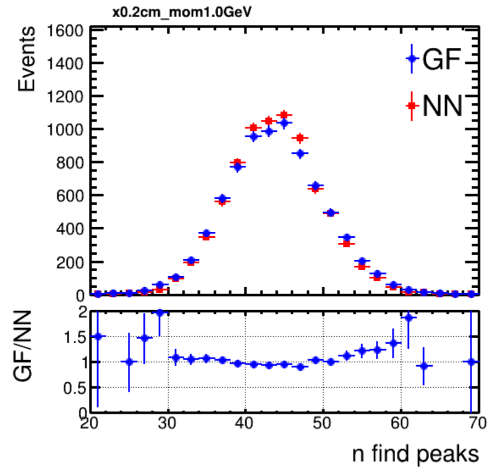


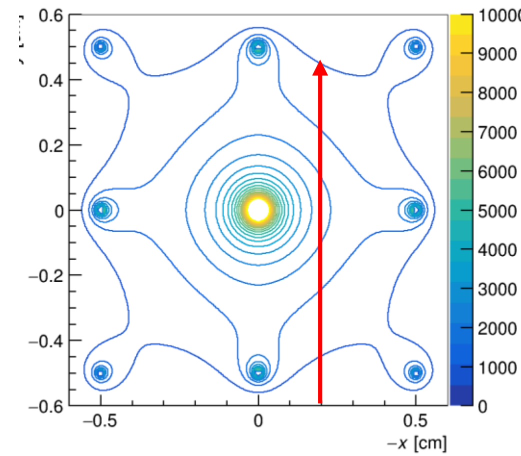


- Checked the number of found peaks(scipy.signal.find_peaks) and total charge
- Good agreement in general
- Similar results for 5 and 10 GeV

Performance check: electron (x=0.2 cm)

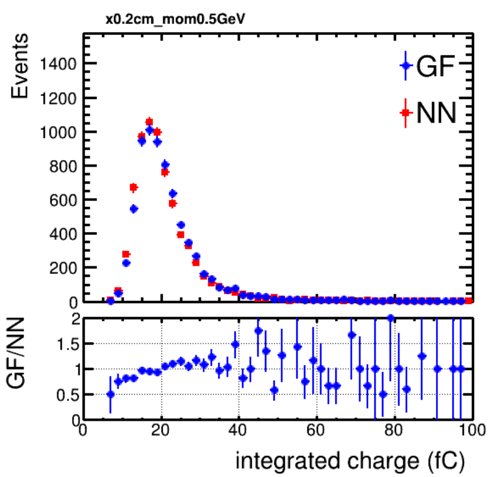


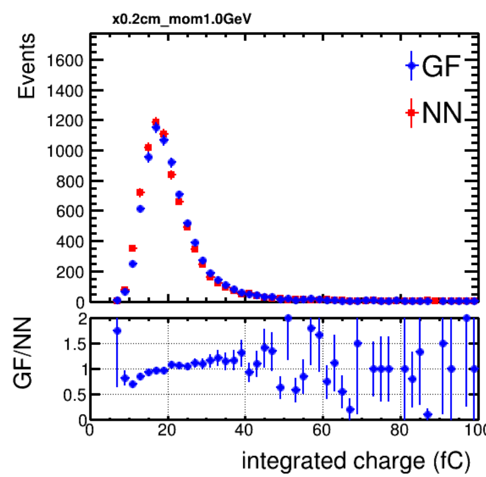




0.5 GeV

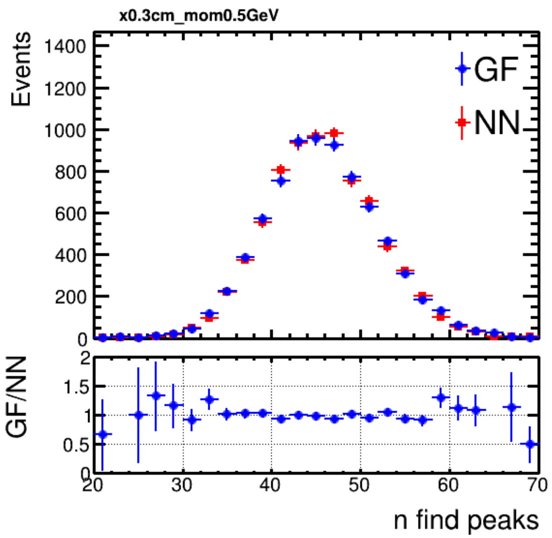
1 GeV

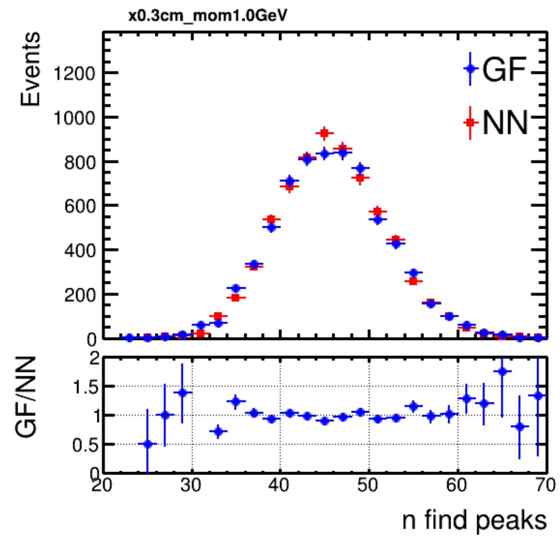


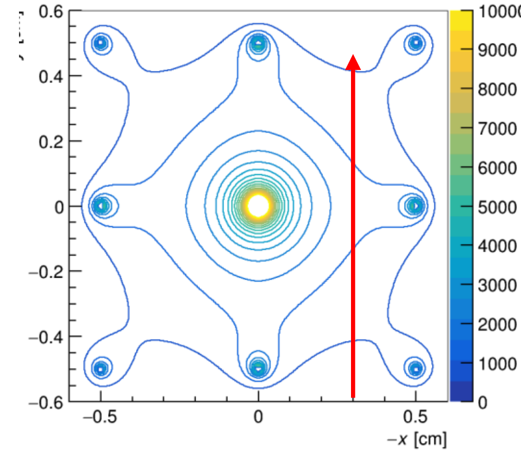


- Good agreement in general
- Similar results for 5 and 10 GeV

Performance check: electron (x=0.3 cm)

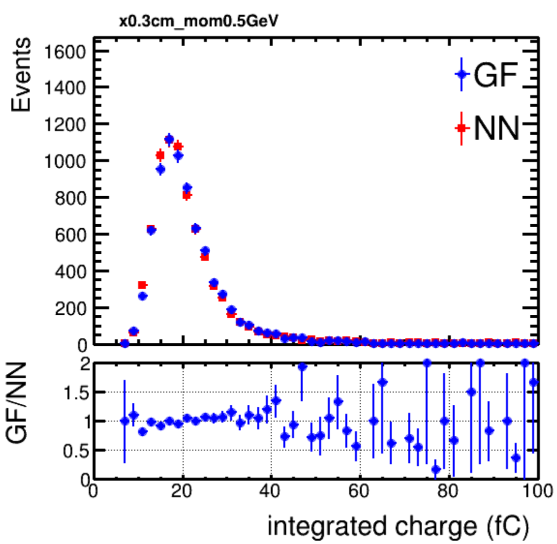


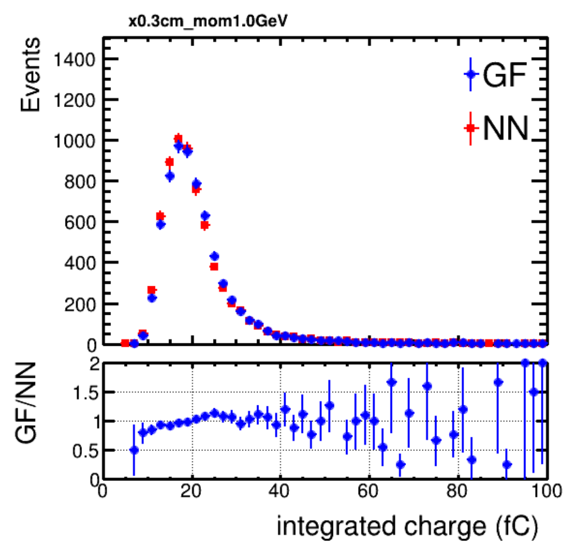




0.5 GeV

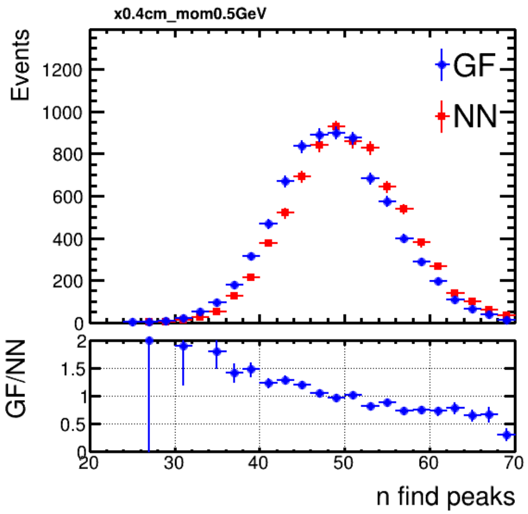
1 GeV

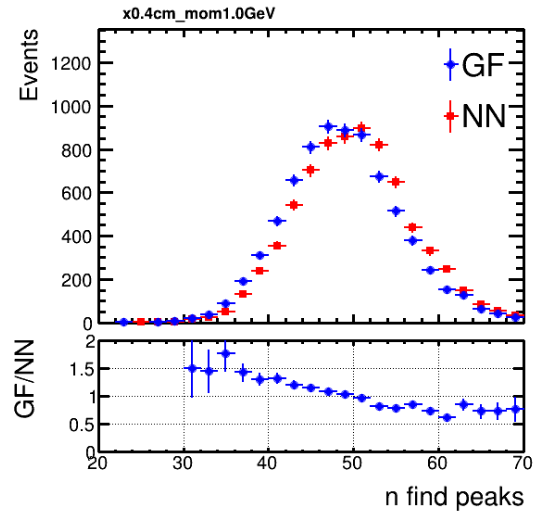


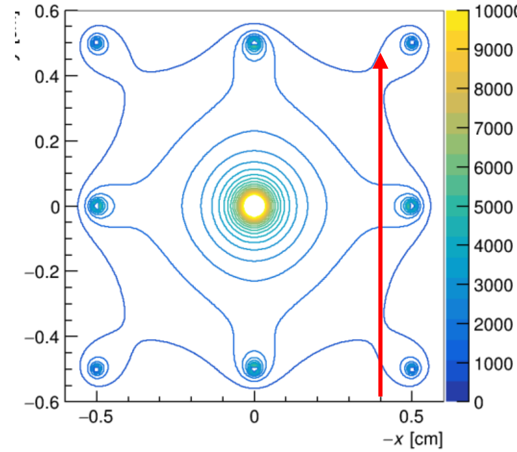


- Good agreement in general
- Similar results for 5 and 10 GeV

Performance check: electron (x=0.4 cm)

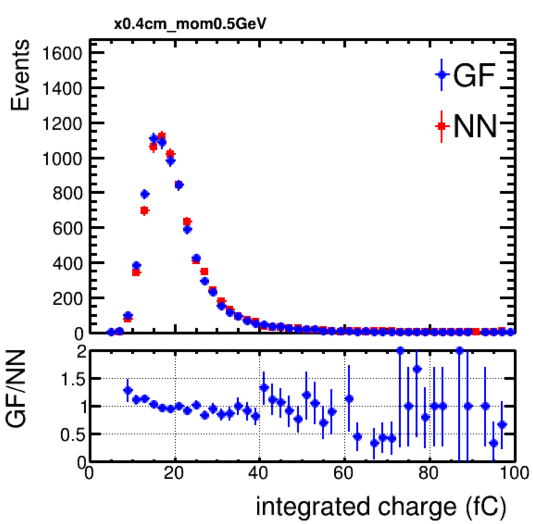


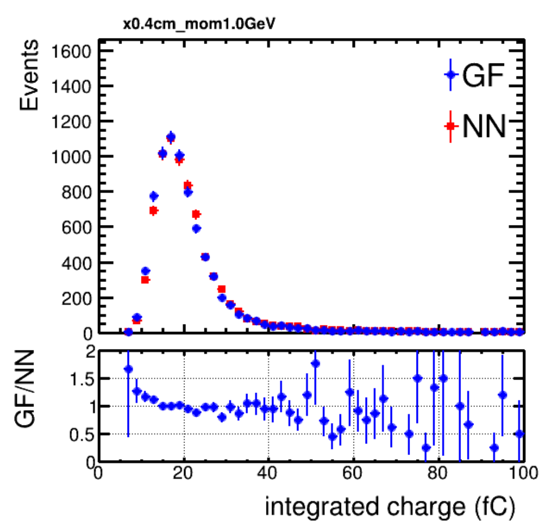




0.5 GeV

1 GeV

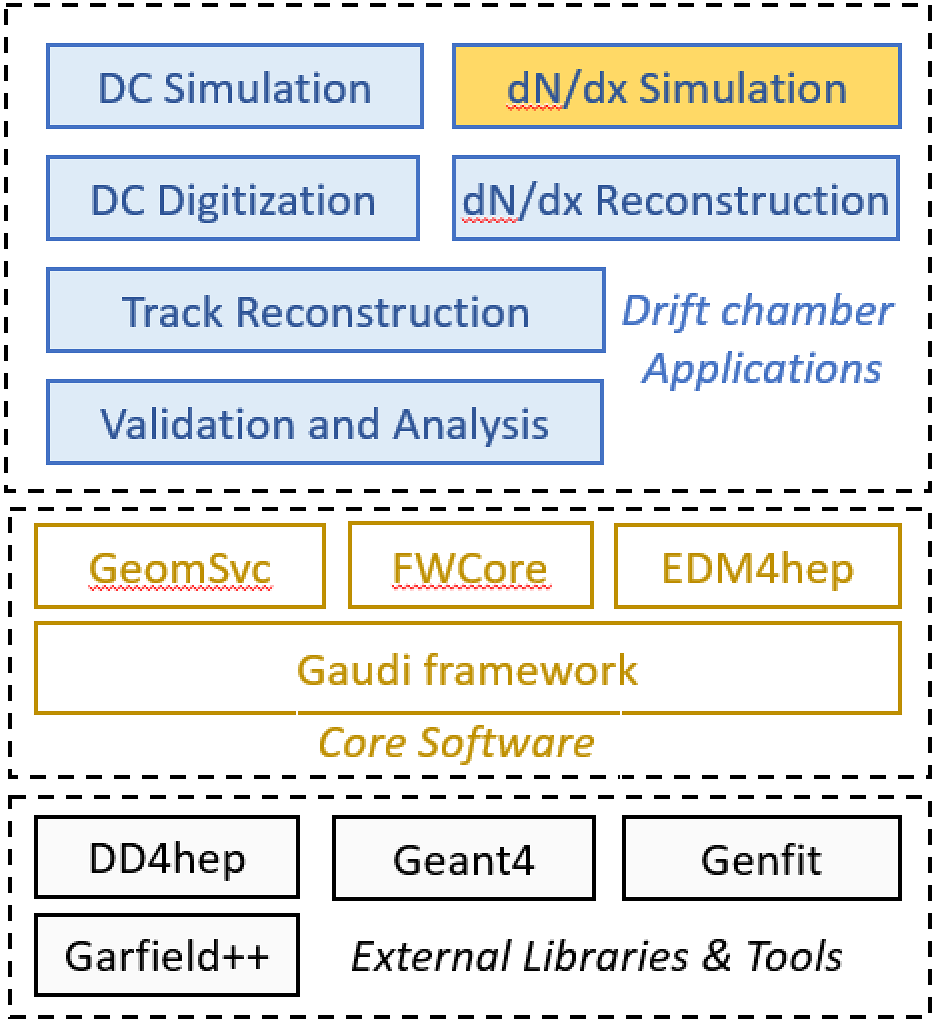




- A little bias for number of found peaks. Could be improved in future
- Similar results for 5 and 10 GeV

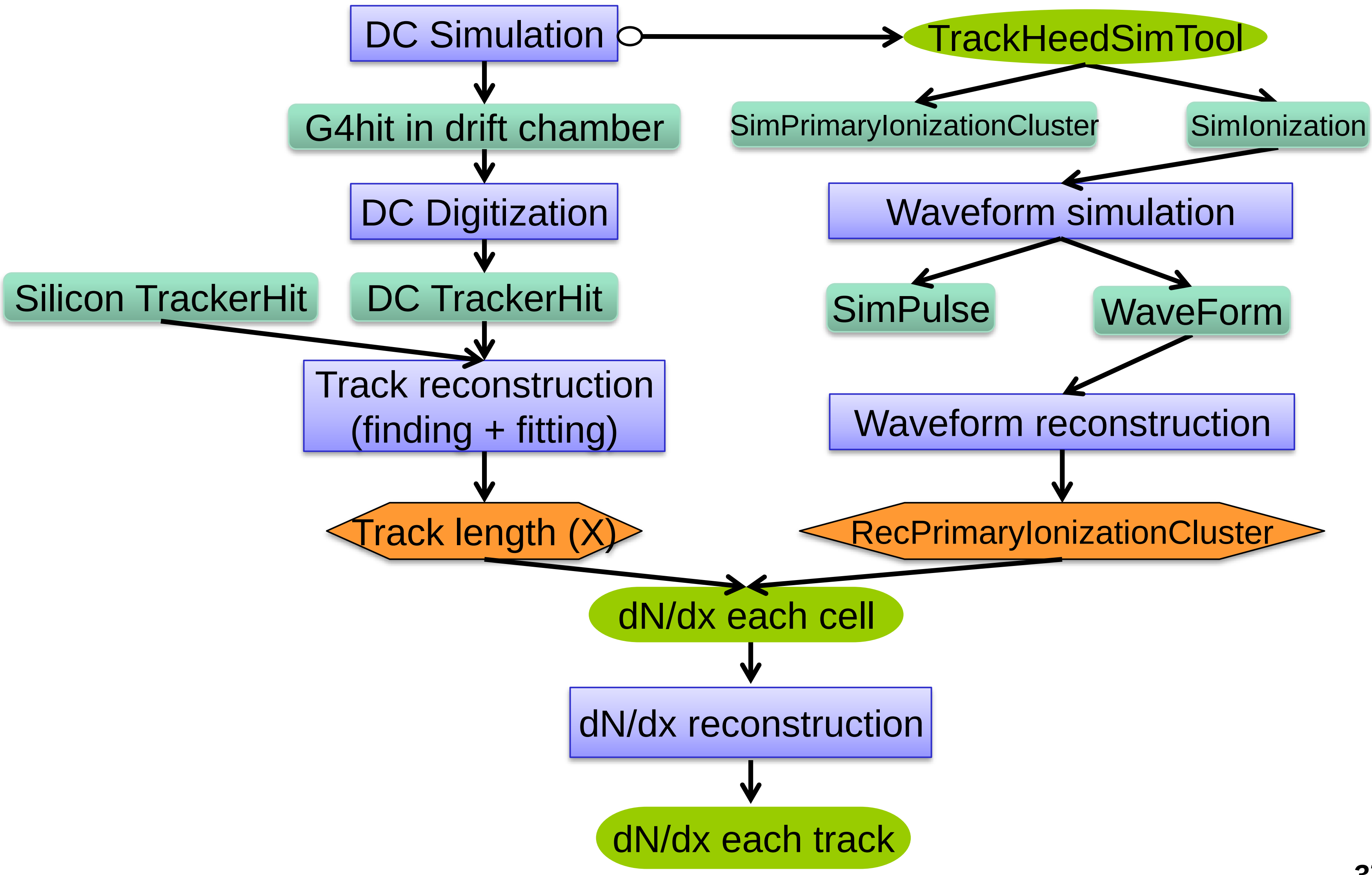
CEPCSW for drift chamber

- Framework:
  - Gaudi
- EDM:
  - Key4hep::EDM4hep
  - Key4hep::FWCore
- Detector geometry and B field:
  - Key4hep::DD4hep
  - GeomSvc
- Drift chamber:
  - DC simulation (Geant4)
  - DC digitization
  - Track reconstruction (Genfit)
  - dN/dx simulation (Garfield++)
  - dN/dx reconstruction

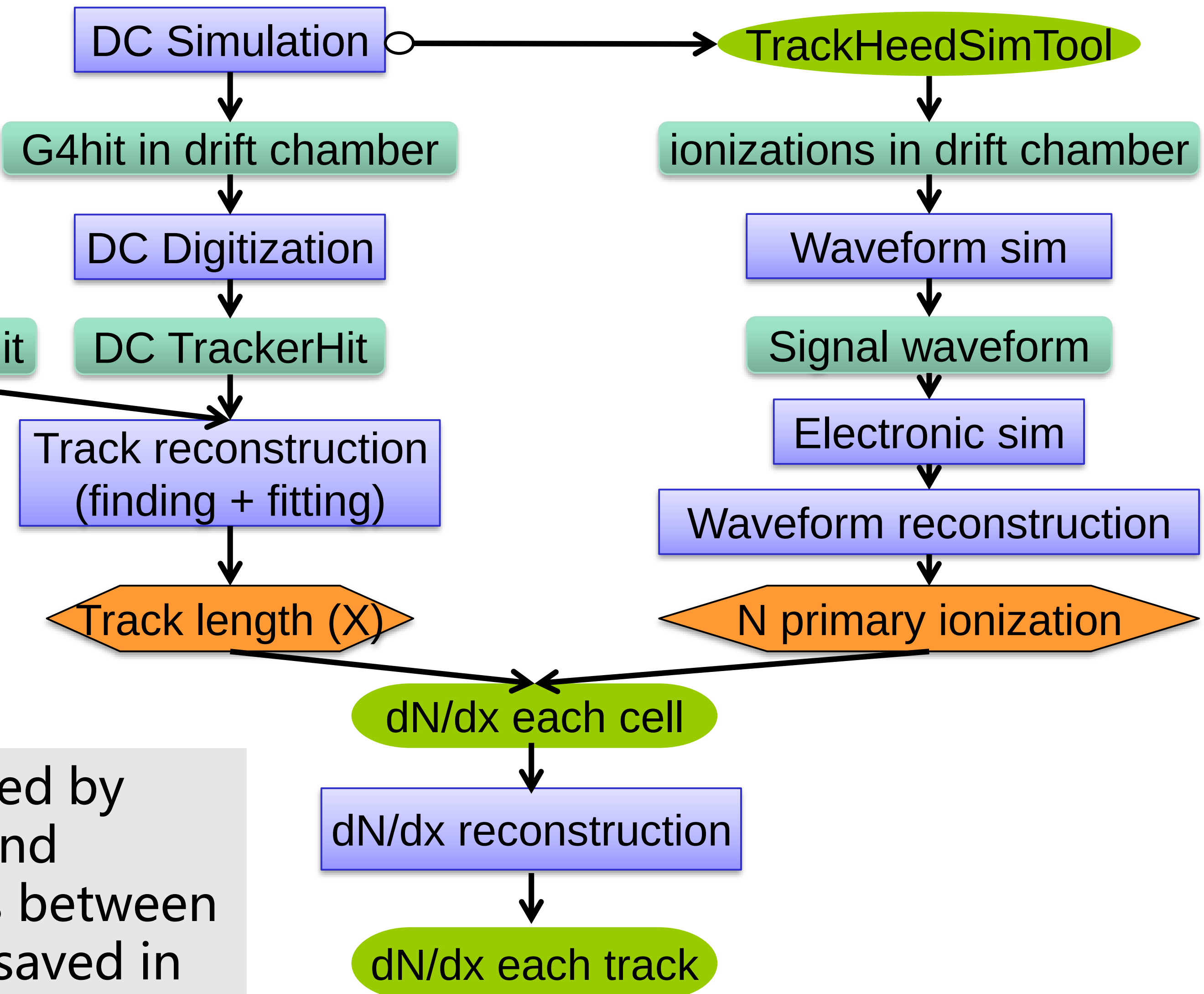


https://github.com/cepc/CEPCSW

DC Simulation and Reconstruction Workflow

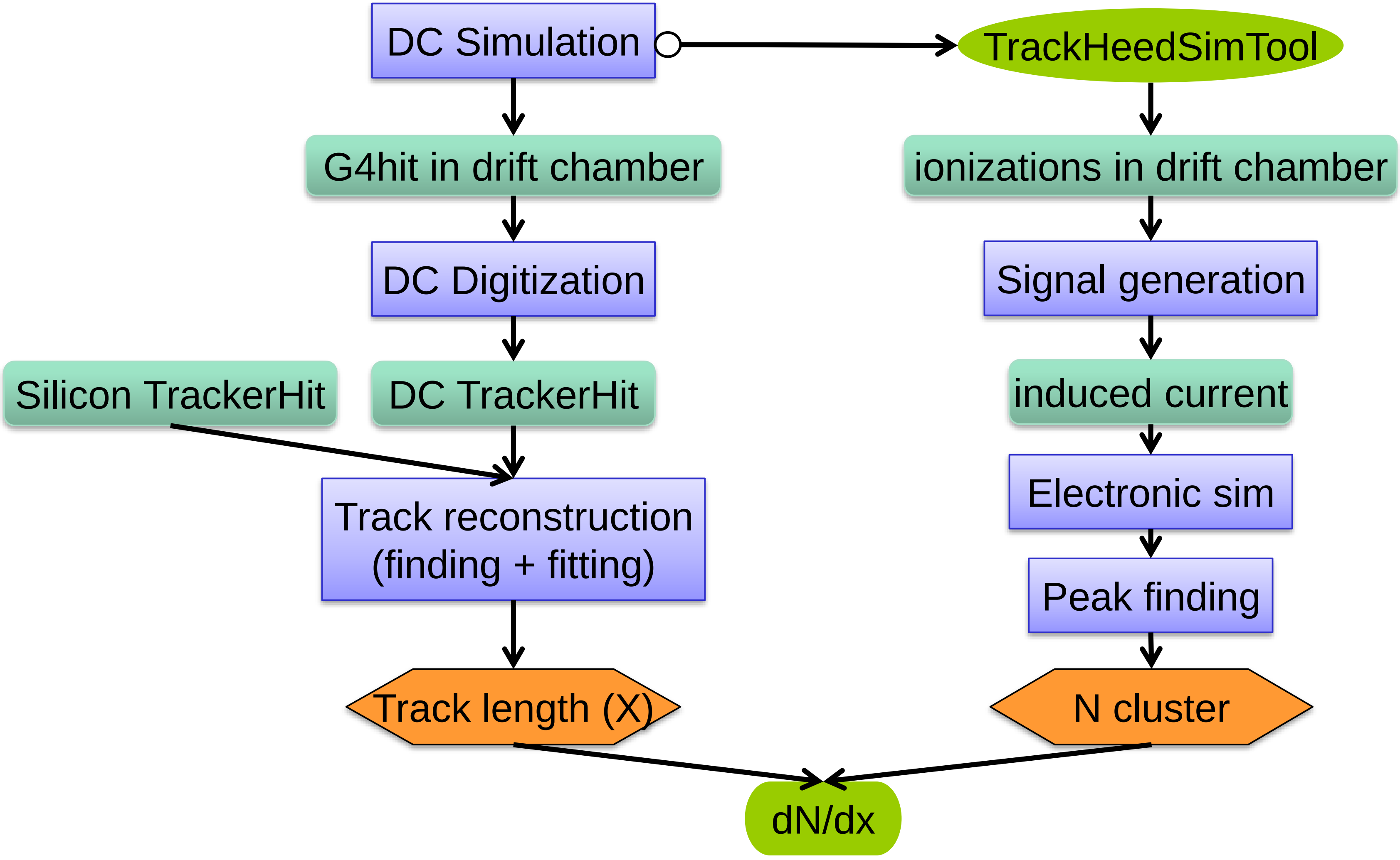


Schema of dN/dx study in CEPCSW



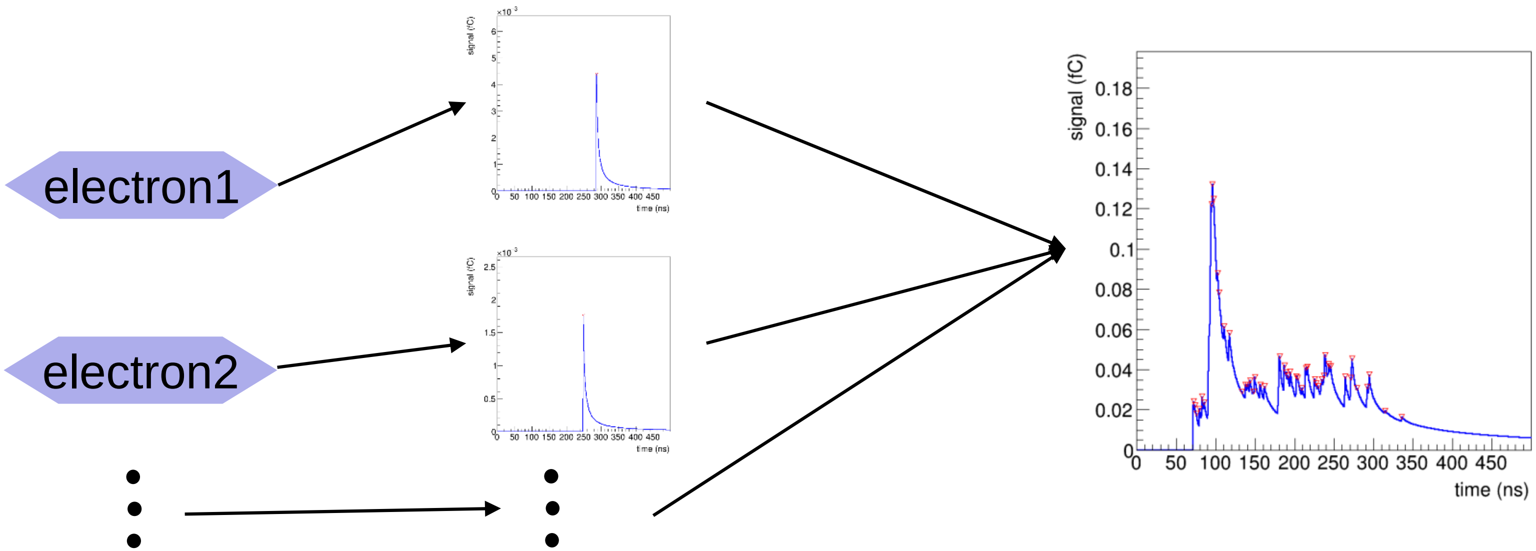
- Data produced by algorithms and relationships between data can be saved in EDM4hep for analysis

Schema of dN/dx study in CEPCSW



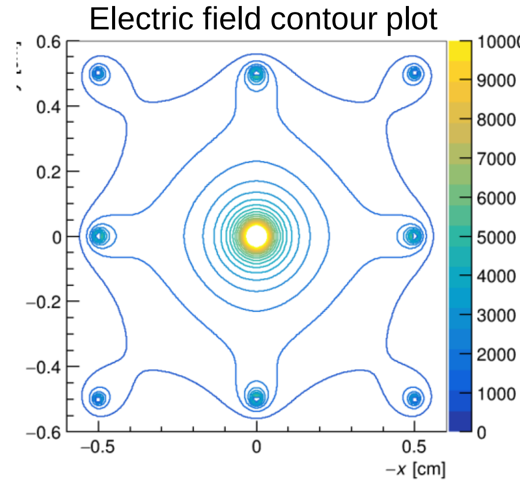
Signal response simulation

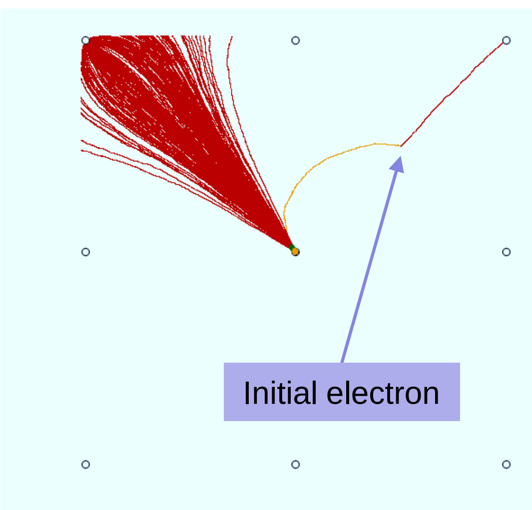
- From ionized electrons to signal response simulation
  - Using Garfield++, simulate the drift and avalanche of electron and drift of ions. Extremely time consuming, $\mathcal{O}(1)$ to $\mathcal{O}(10)$ seconds for different gas just for one electron
  - Going to use parameterization (fast simulation) method (parameters are based on Garfield++ simulation results), will be much faster
    - For each electron, simulate its own pulse
    - As done by Garfield++, piling up all pulses from same drift chamber cell gives final signal response



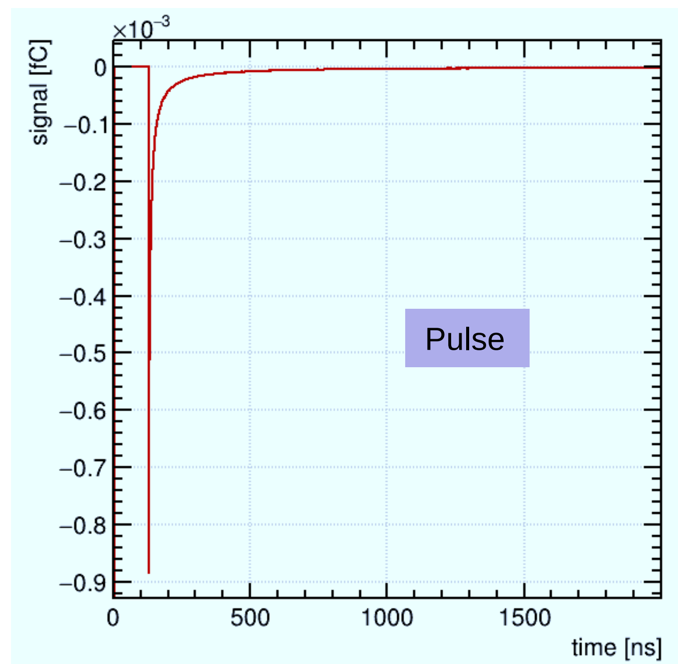
Training Dataset

- Garfield++ simulation:
  - 350k single electron event with electron position uniformly distributed $1 \times 1\ \mathrm{cm}^2$ drift chamber cell
  - Gas: 50% He + 50 % C_4H_{10}
  - Center signal wire (2000 V), eight field wires (0 V)





- One electron drift and avalanche
- Ions drift



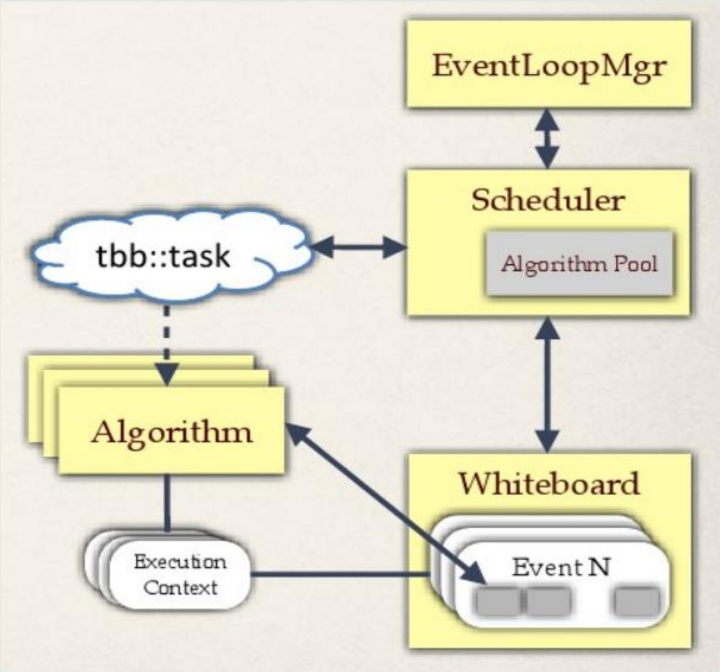
- The Gaudi Hive is studied for multithreaded simulation of drift chamber
- User defined or edm4hep format data is supported in Gaudi Hive
- Using Gaudi::Functional instead of Algorithm have been tried, finding problems with edm4hep data, under investigation

- Future plan:
  - Try to write the output to root files
  - Combining with Geant4 simulation
  - Integrating with k4FWCore, maybe develop a multithreading version of k4FWCore
  - Creating a prototype of CEPCSW based on GaudiHive
- Welcome to check the code: https://github.com/wenxingfang/DCMTSim

Gaudi Hive Operation

Charles Leggett
September 18 2015

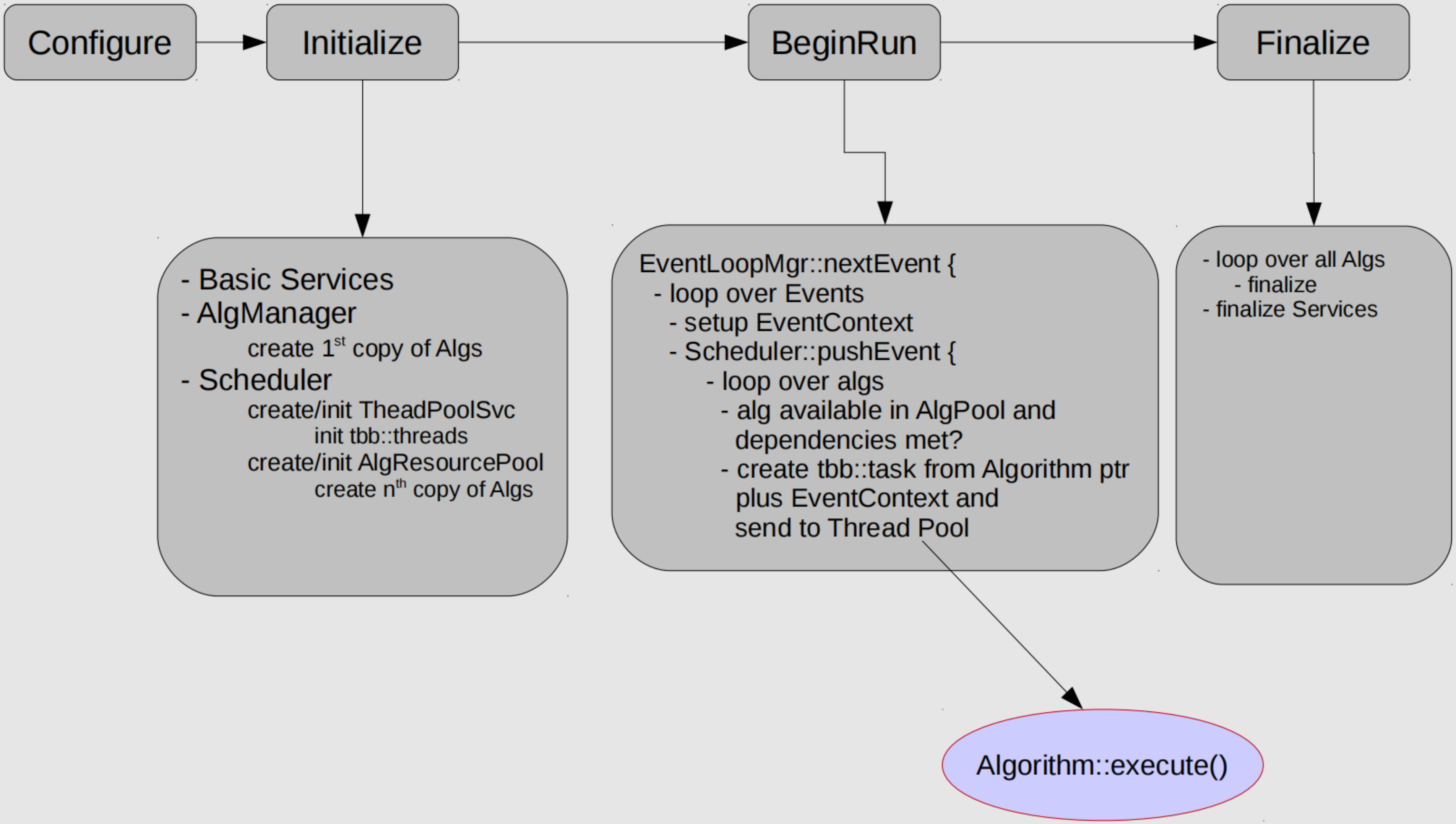
- Configuration, Initialization, Finalization are performed serially in "master" thread
  - only Algorithm::execute is concurrent
- Algorithms are scheduled when data becomes available
  - Algorithms must declare their inputs at initialization or dynamically with DataHandles
  - tbb::task wraps the pair (Algorithm*, EventContext)
- Several instances of the same Algorithm can co-exist
  - **cloning**: create new instance if can be scheduled, and all other instances busy
  - running on different events
  - managed by AlgoPool



- Multiple events are managed simultaneously
  - increases probability of scheduling an Algorithm
  - whiteboard DataStore is thread safe

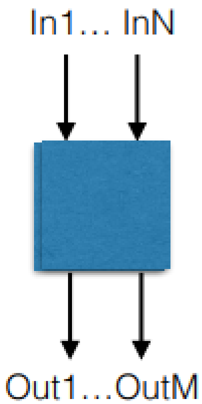
Execution Flow

Charles Leggett
September 18 2015



Gaudi::Functional

- Most algorithms look like "some data in" -> "some data out"
- Standardize the common pattern of getting data our of the TES, working on it, and putting it back in (in a different location).
  - Less code to write
  - More uniform code and easy to understand
  - Can be Re-Entrant, no need for clone, save memory
  - Multithreading friendly



- Patterns available:
  - Consumer, Producer, Filter, Transformer, MultiTransformer, ScalarTransformer , …

Re-Entrant test

- Gaudi::Functional Re-Entrant test

```
class DataMaker_v1 : public Gaudi::Functional::Producer<std::vector<float>()> {
public:
    DataMaker_v1(const std::string& name, ISvcLocator* svcLoc)
                  : Producer( name, svcLoc,
                    KeyValue("OutputLocation", {"MyVec_v1"})) {
     std::string bp_file = "/junofs/users/wxfang/MyGit/tmp/check_G4FastSim_2021
     char* cstr = new char[bp_file.size() + 1];
     strcpy(cstr, bp_file.c_str());
     m_NNPred  = new NNPred(cstr);
}
    std::vector<float> operator()() const override;
protected:
    NNPred* m_NNPred;
};
```

- Gaudi::Functional is re-entrantable
- The pytorch model is re-entrantable

```
evtslots = 10

whiteboard = HiveWhiteBoard("EventDataSvc", EventSlots=evtslots)

slimeventloopmgr = HiveSlimEventLoopMgr(OutputLevel=DEBUG)

scheduler = AvalancheSchedulerSvc(ThreadPoolSize=8, OutputLevel=WARNING)

a1 = DataMaker_v1()
```

```
ApplicationMgr(
    EvtMax=100,
    EvtSel='NONE',
    ExtSvc=[whiteboard],
    EventLoop=slimeventloopmgr,
    TopAlg=[a1],
    HistogramPersistency = "ROOT",
    MessageSvcType="InertMessageSvc")
```

```
HiveSlimEventLo...   DEBUG createdEvts: 5, freeslots: 6
DataMaker_v1          INFO executing DataMaker_v1
HiveSlimEventLo...   DEBUG work loop iteration 7
HiveSlimEventLo...   DEBUG createdEvts: 6, freeslots: 5
DataMaker_v1          INFO executing DataMaker_v1
HiveSlimEventLo...   DEBUG work loop iteration 8
DataMaker_v1          INFO executing DataMaker_v1
HiveSlimEventLo...   DEBUG createdEvts: 7, freeslots: 4
DataMaker_v1          INFO executing DataMaker_v1
HiveSlimEventLo...   DEBUG work loop iteration 9
HiveSlimEventLo...   DEBUG createdEvts: 8, freeslots: 3
DataMaker_v1          INFO executing DataMaker_v1
HiveSlimEventLo...   DEBUG work loop iteration 10
DataMaker_v1          INFO executing DataMaker_v1
HiveSlimEventLo...   DEBUG createdEvts: 9, freeslots: 2
HiveSlimEventLo...   DEBUG work loop iteration 11
HiveSlimEventLo...   DEBUG createdEvts: 10, freeslots: 1
DataMaker_v1          INFO executing DataMaker_v1
HiveSlimEventLo...   DEBUG work loop iteration 12
HiveSlimEventLo...   DEBUG Draining the scheduler
HiveSlimEventLo...   DEBUG Waiting for a context
DataMaker_v1          INFO executing DataMaker_v1
```

Using Gaudi::Functional

A1 = MakerIons("IonsProducer")
A1.OutputLocation="/Event/MyIons"

```
class MakerIons : public Gaudi::Functional::Producer<IonVec()> {
public:
    MakerIons(const std::string& name, ISvcLocator* svcLoc)
             : Producer( name, svcLoc,
               KeyValue("OutputLocation", {"MyIonVec"})) {}

    IonVec operator()() const override;
};
```

A2 = SimWF("SimA2")
A2.InputLocation="/Event/MyIons"
A2.OutputLocation="/Event/MySimA2"
A2.partition=[0 ,0.5]

A3 = SimWF("SimA3")
A3.InputLocation="/Event/MyIons"
A3.OutputLocation="/Event/MySimA3"
A3. partition=[0.5 ,1]

```
struct IonVec{
    int data;
    std::vector<int> cell_id;
    std::vector<float> x;
    std::vector<float> y;
};
```

A4 = MergeWF("MergeWF")
A4.InputLocations=["/Event/MySimA2", "/Event/MySimA3"]
A4.OutputLocation="/Event/MyMergeWF"

```
struct WFVec{
    int data;
    std::vector<int> e_id;
    std::vector<int> cell_id;
    std::vector<float> peak_time;
    std::vector<float> peak_value;
    std::vector<std::vector<float> > charges;
};
```

```
using BaseClass_t = Gaudi::Functional::Traits::BaseClass_t<Gaudi::Algorithm>;
struct SimWF final : Gaudi::Functional::Transformer<WFVec( const IonVec& ), BaseClass_t> {

  SimWF( const std::string& name, ISvcLocator* svcLoc )
      : Transformer( name, svcLoc, KeyValue( "InputLocation", "/Event/MyInt" ),
                     KeyValue( "OutputLocation", "/Event/MyFloat" ) ) {}
  WFVec operator()( const IonVec& input ) const override {
```

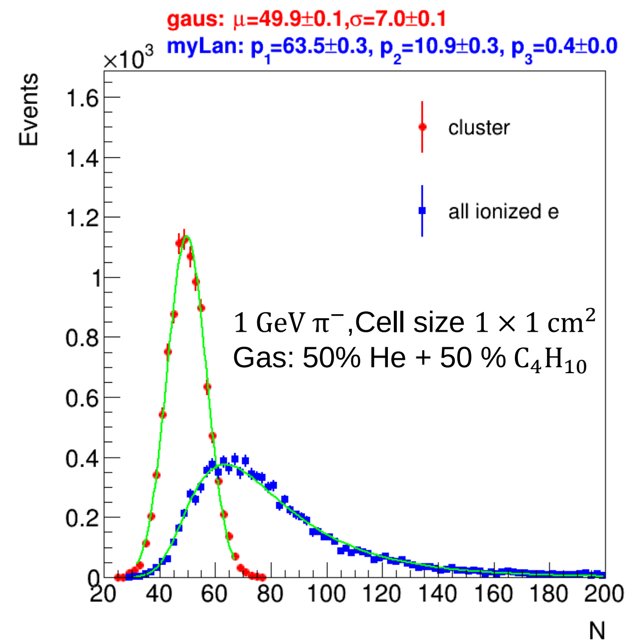
```
struct MergeWF final : Gaudi::Functional::MergingTransformer<WFVec( const Gaudi::Functional::vector_of_const_<WFVec>& ), BaseClass_t> {

  MergeWF( const std::string& name, ISvcLocator* svcLoc )
     : MergingTransformer( name, svcLoc, {"InputLocations", {}},{"OutputLocation", "/Event/MyConcatenatedIntVector"} ) {}

  WFVec operator()( const Gaudi::Functional::vector_of_const_<WFVec>& input ) const override {
```

Motivation

- The particle identification is very important for CEPC flavor physics study. Good hadron separation up to 20 GeV is essential
- Traditionally: using dE/dx method
  - Due to the production of delta electron, the deposited energy follows Landau distribution
  - Resolution is ~6%
- New technique: using dN/dx (cluster counting) method
  - The number of primary ionization follows Poisson distribution
  - Resolution could reaches <3%
- The dN/dx technique will be widely explored in CEPC drift chamber detector

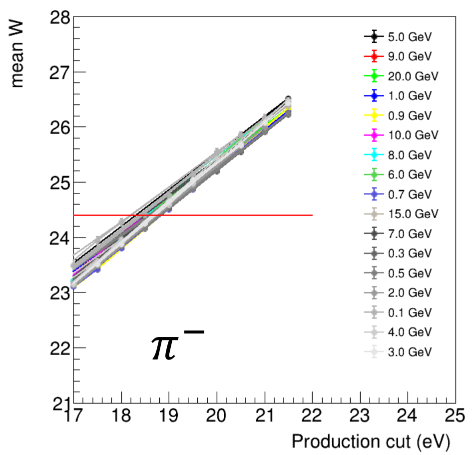


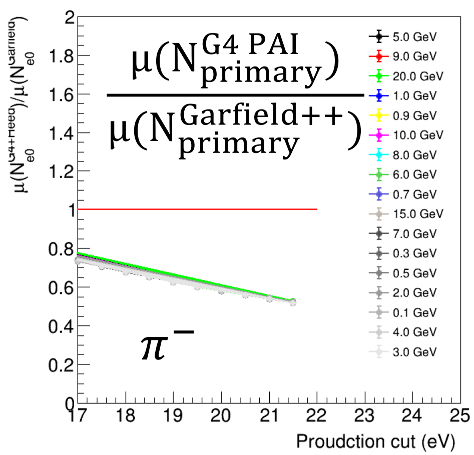
Ionization simulation in gas

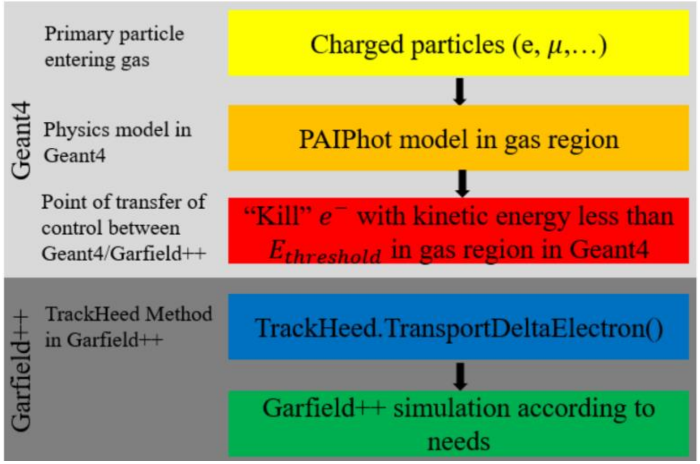
- Garfield++
  - Using Heed PAI model to simulate the ionization in gas precisely
  - Can simulate the drift and avalanche of electrons in gas
  - The drift of ions to cathode can be simulated
  - The induced current can be given
  - It is useful to study and characterize the properties of gas detector with simple geometry but not for full drift chamber detector
- Geant4
  - Can simulate collider events and the interaction between particles and materials in full detector
  - It does not simulate the ionization process properly, neither the drift and avalanche processes
- In order to simulate including particle interaction will detector materials, ionization in gas, drift and avalanche processes in full detector, we try to combined Geant4 and Garfield++ in CEPCSW

Ionization simulation in CEPCSW (G4 PAI)

- First try: according to paper [“Interfacing Geant4, Garfield++ and Degrad for the Simulation of Gaseous Detectors” :]
  - Geant4 PAI model to simulate primary or secondary ionization
  - TrackHeed to simulate ionization from residual delta electron
- However, it was found that the primary ionization produced by this method is much less than Garfield++.



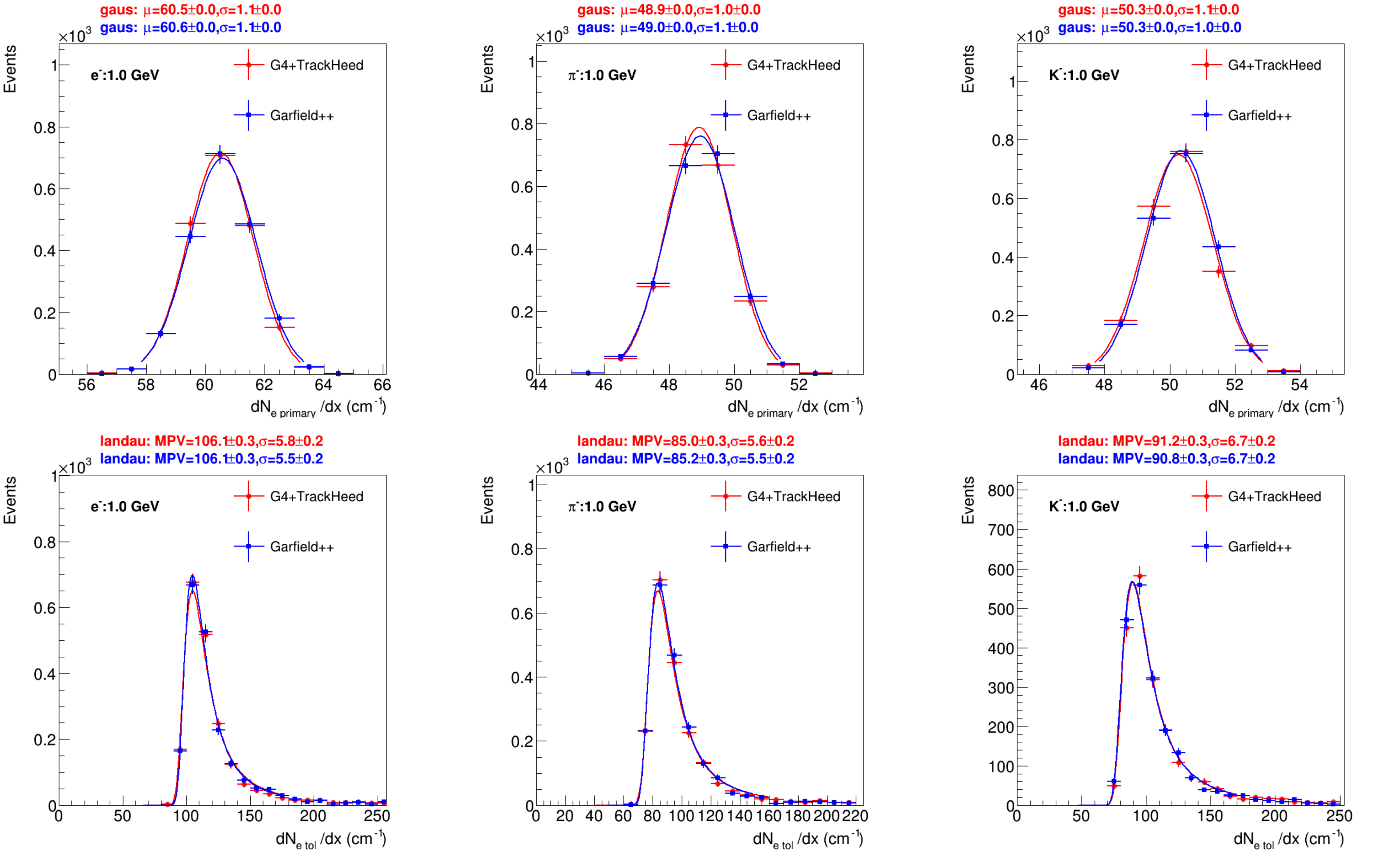




- Checking with authors:
  - This method designed to obtain correct energy deposition (or total ionizations)
  - It is true that this method will give less primary ionizations, so this method is obsoleted

Ionization simulation performance

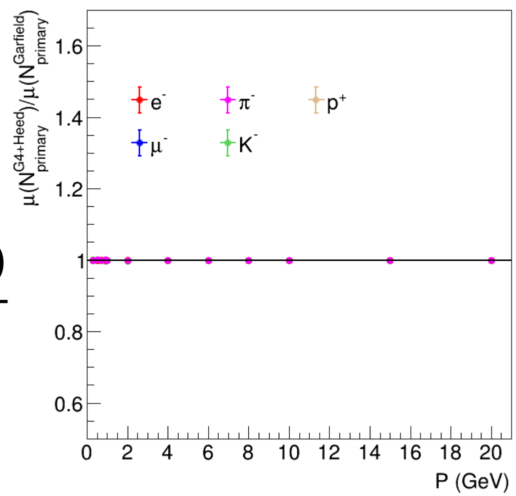
- Gas: 50% He + 50 % C_4H_{10}

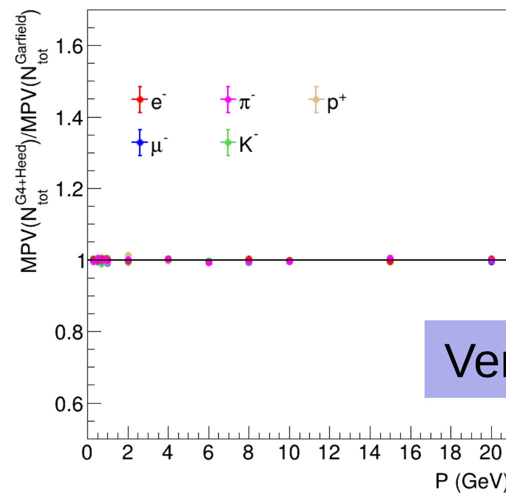


Ionization simulation performance

- Gas: 50% He + 50 % C_4H_{10}

$$\frac{\mu(N^{G4+TrackHeed}_{primary})}{\mu(N^{Garfield++}_{primary})}$$

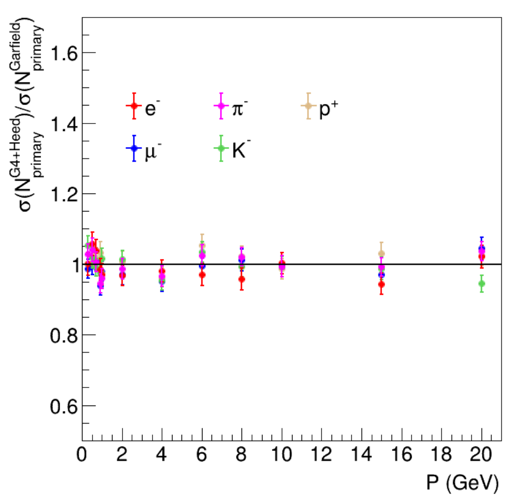


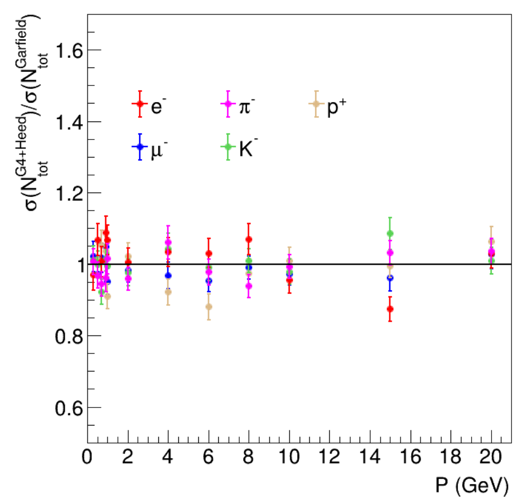


$$\frac{MPV(N^{G4+TrackHeed}_{tot})}{MPV(N^{Garfield++}_{tot})}$$

Very good agreement

$$\frac{\sigma(N^{G4+TrackHeed}_{primary})}{\sigma(N^{Garfield++}_{primary})}$$

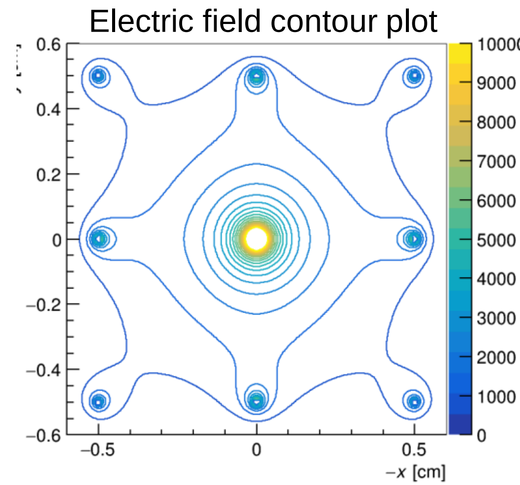


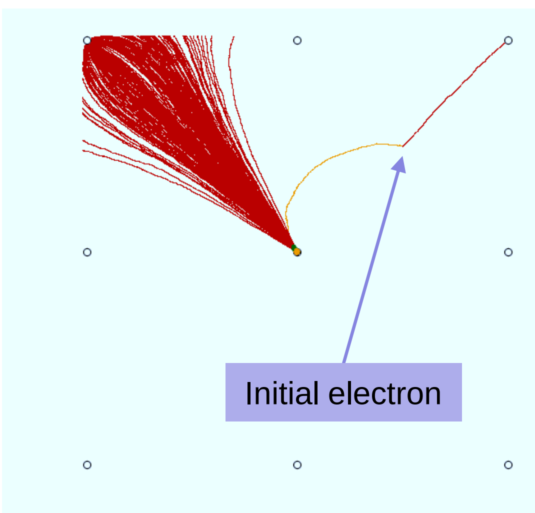


$$\frac{\sigma(N^{G4+TrackHeed}_{tot})}{\sigma(N^{Garfield++}_{tot})}$$

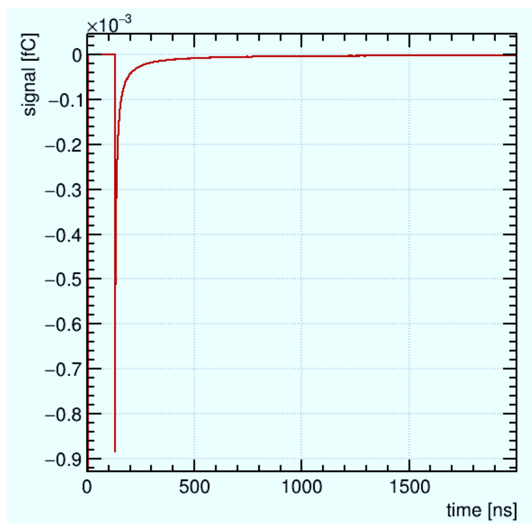
Getting parameters from Garfield++

- Garfield++ simulation:
  - 500k electrons uniformly distributed $1 \times 1\ \mathrm{cm}^2$ drift chamber cell
  - Gas: 50% He + 50 % C_4H_{10}
  - Center signal wire (2000 V), eight field wires (0 V)

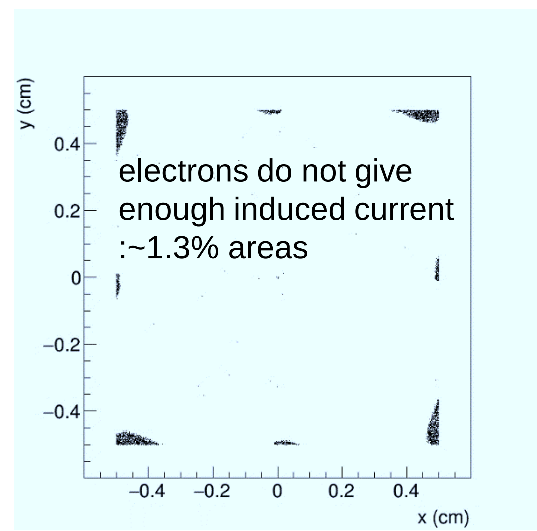




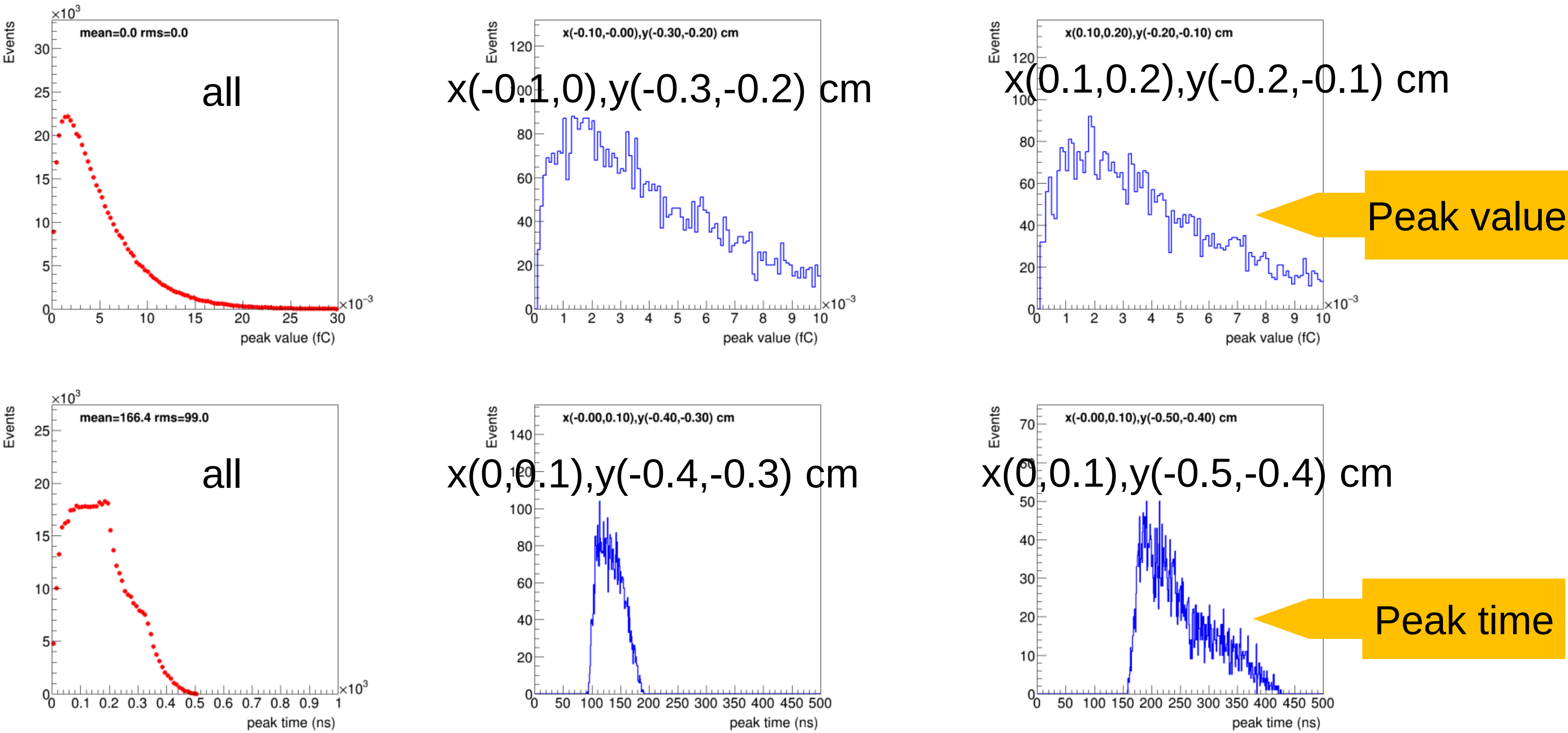
- One electron drift and avalanche
- Ions drift



Induced current



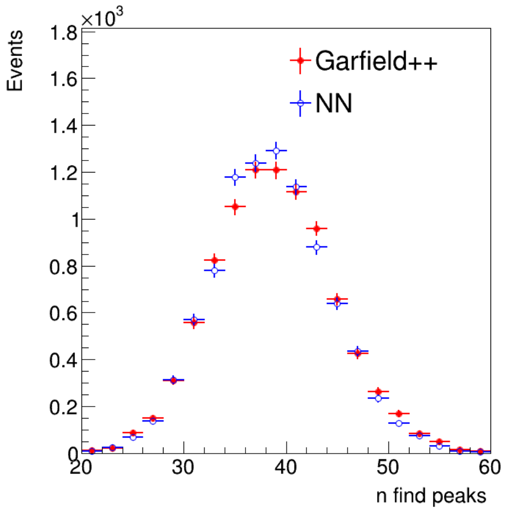
Garfield++ simulation

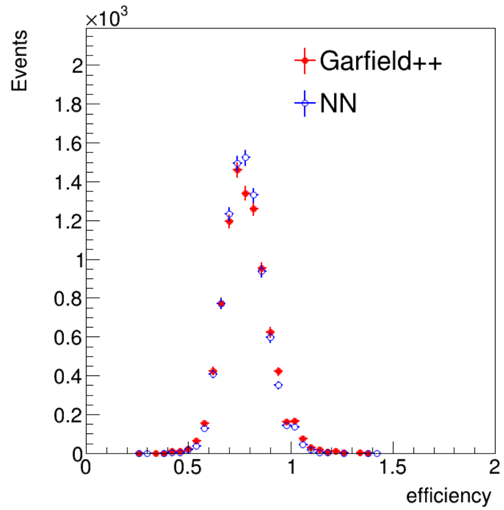


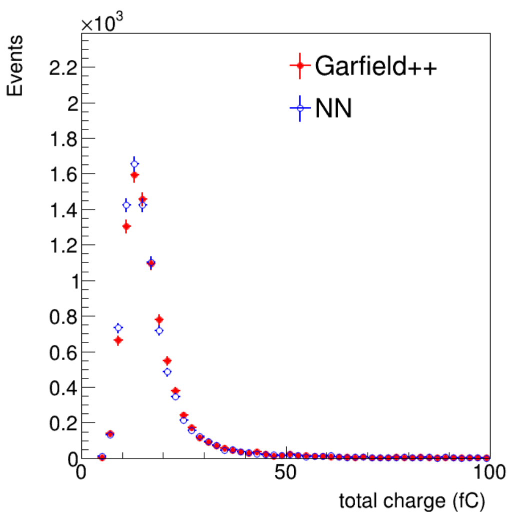
- Simulate (peak_time, peak_value):
  - Sampling method base on which bin the electron (x,y) is located
  - Machine learning method according electron (x, y) without binning

Signal waveform simulation

- 1GeV π^-







- By this way, waveform simulation is not related to Geant4 and it is independent between each electron. Therefore, the waveform simulation can be transported to GPU or using multithreading technique to get speed up

- Good agreement between NN and Garfield++
- Average time for one cell simulation:
  - Garfield++: ~250 s
  - NN: ~1 s

- For different particles, only the ionization part is different, the waveform simulation is the same