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

dN/dX simulation within the CEPCSW framework

Wenxing Fang (IHEP)

On behalf of drift chamber working group

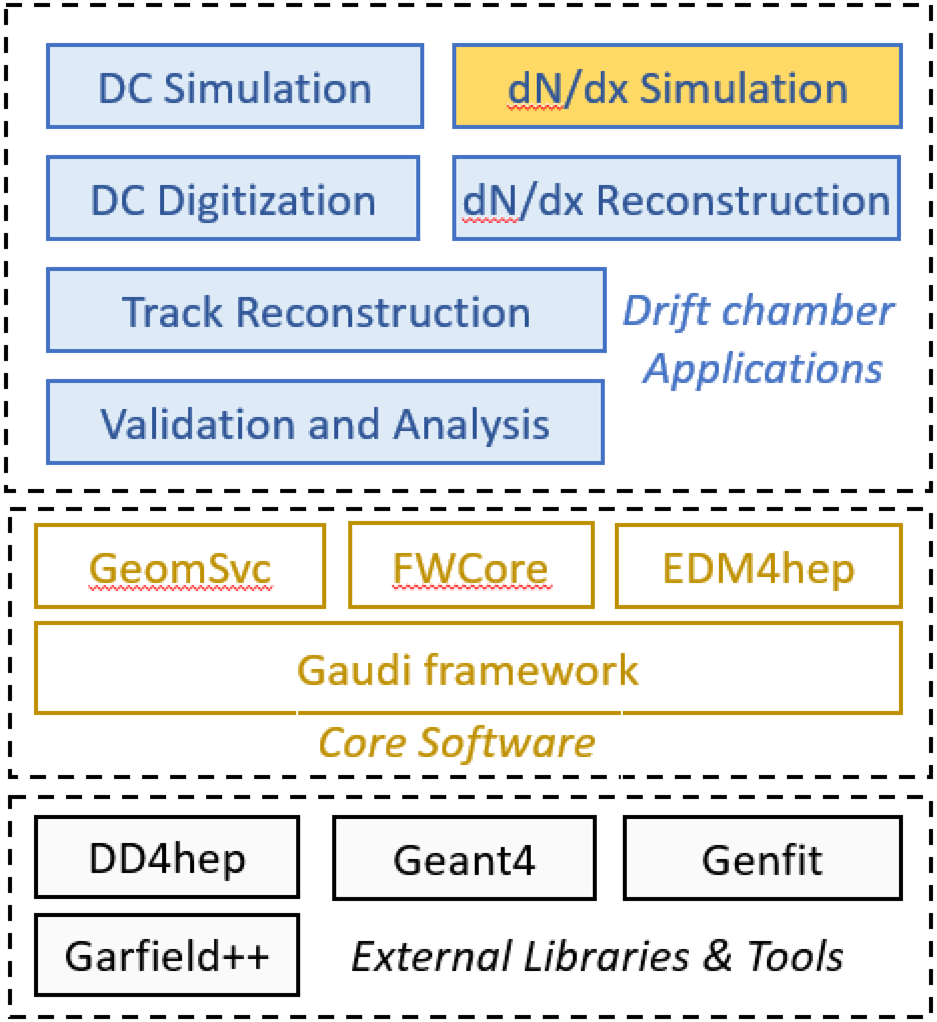
CEPC Day Meeting 2021.08.30

Motivation

- CEPC is a precise experiment
  - Higgs, W, Z, …
  - PID performance is important
- From previous study, the primary ionization counting (dN/dx) method has potential to get very good PID performance (<3% resolution)
- To prove that, the dN/dx method will be detailed studied for CEPC drift chamber. Need precise dN/dx simulation and reconstruction
- We performed the study in CEPCSW framework

CEPCSW for drift chamber

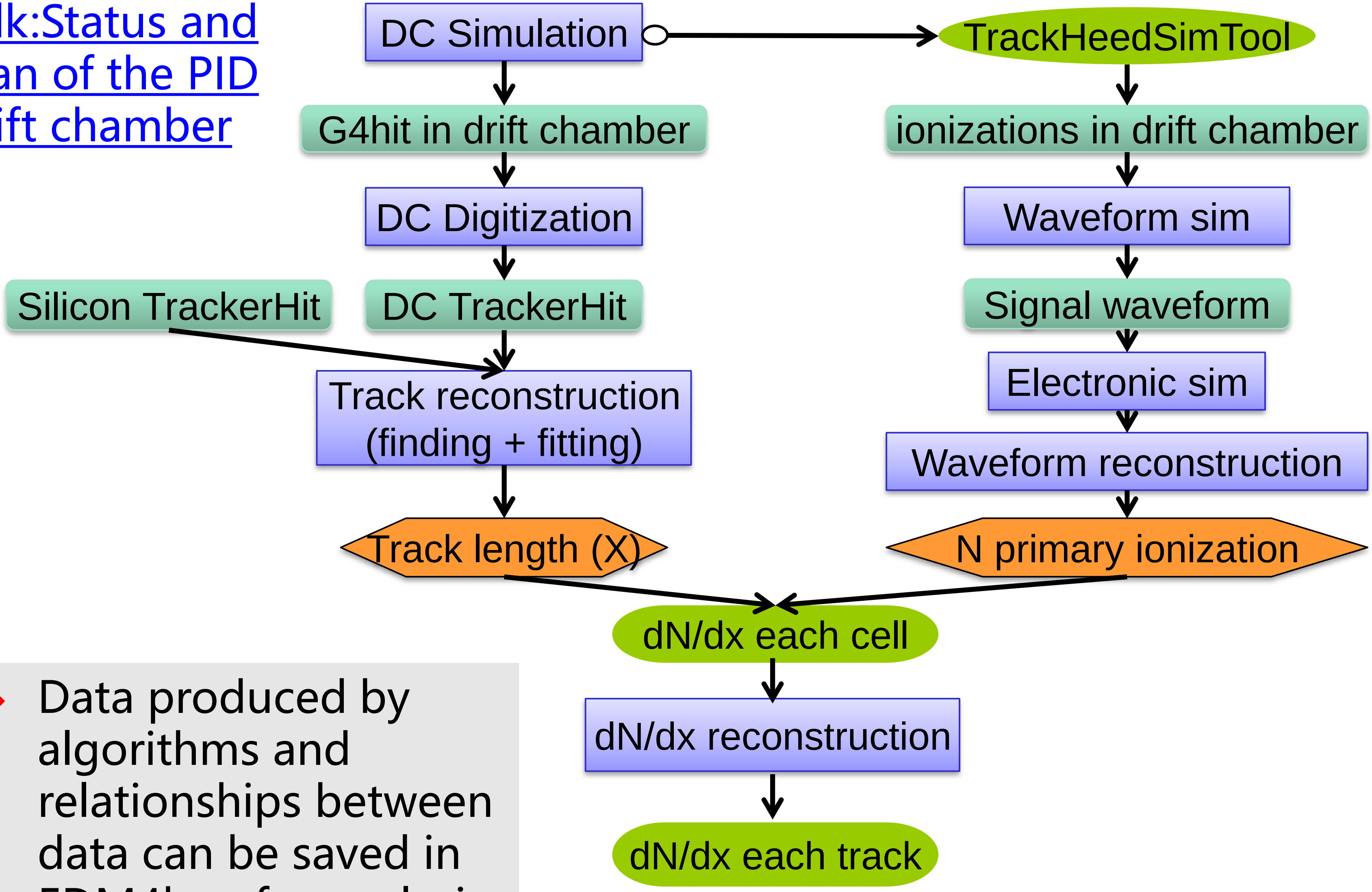
- Framework:
  - Gaudi
- EDM:
  - EDM4hep
  - FWCore
- Detector geometry and B field:
  - DD4hep
  - GeomSvc
- Drift chamber:
  - DC simulation (Geant4)
  - DC digitization
  - Track reconstruction (Genfit)
  - dN/dx simulation (Garfield++)
  - dN/dx reconstruction



https://github.com/cepc/CEPCSW

Schema of dN/dx study in CEPCSW

Talk:Status and plan of the PID drift chamber



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

Ionization simulation

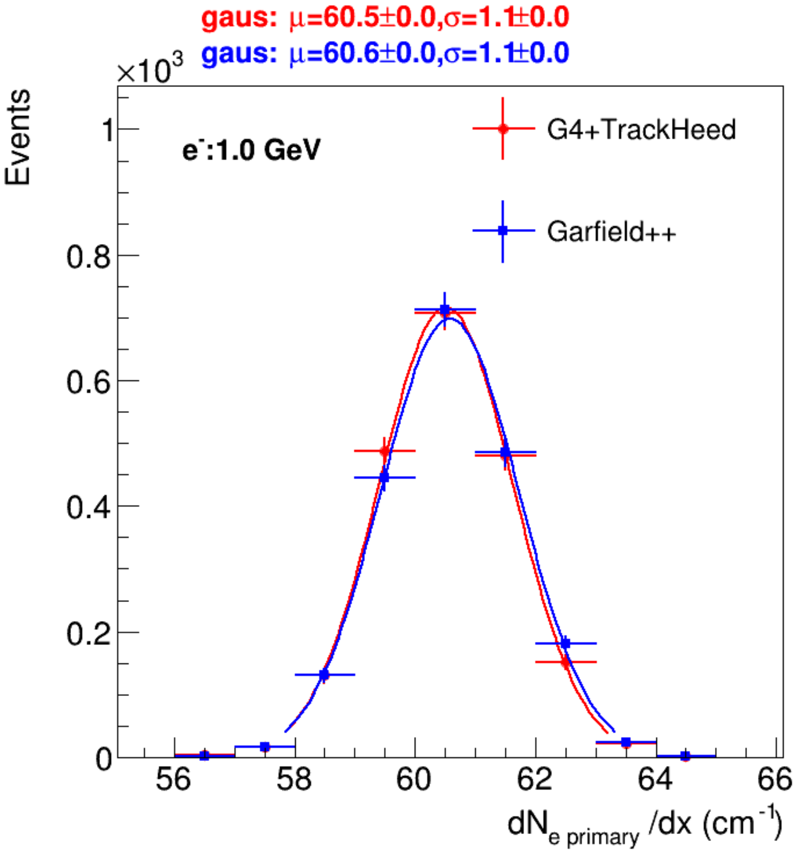
- From previous studies, we know Geant4 can not simulate the ionization process properly (arXiv:2105.07064)
- Garfield++ is commonly used for precise ionization simulation for simple geometry
- In order to do a detailed drift chamber simulation, including particle interaction with detector materials, 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 backup)

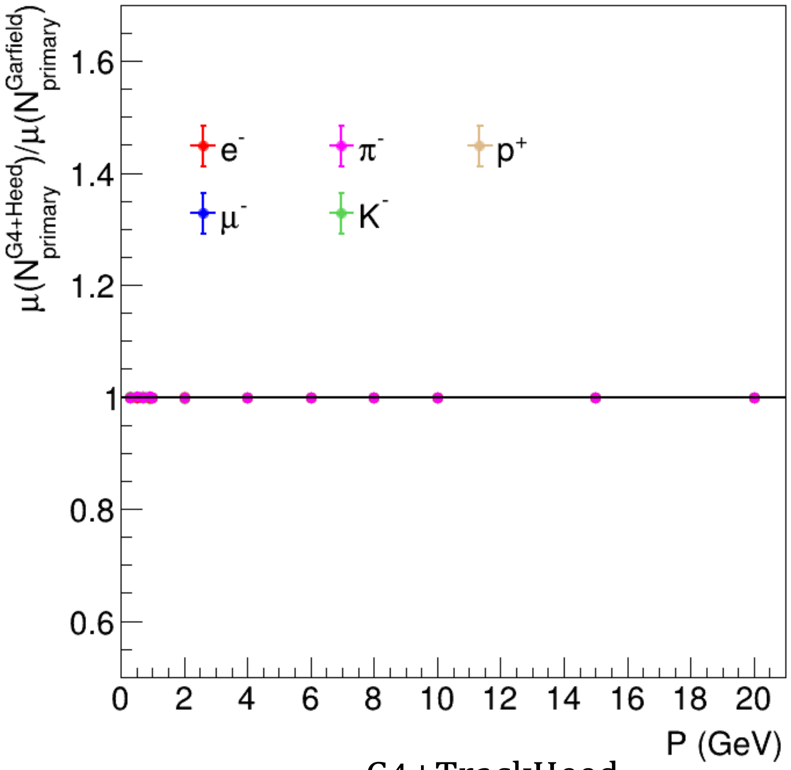
Ionization simulation in CEPCSW

- Combining Geant4 and Garfield++ at G4Step level
- TrackHeedSimTool 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 added to Garfield++ PR)
  - Output: primary and total ionization information (contains position, time, cell id), saved in EDM (SimTrackerHit collection)
  - 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

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



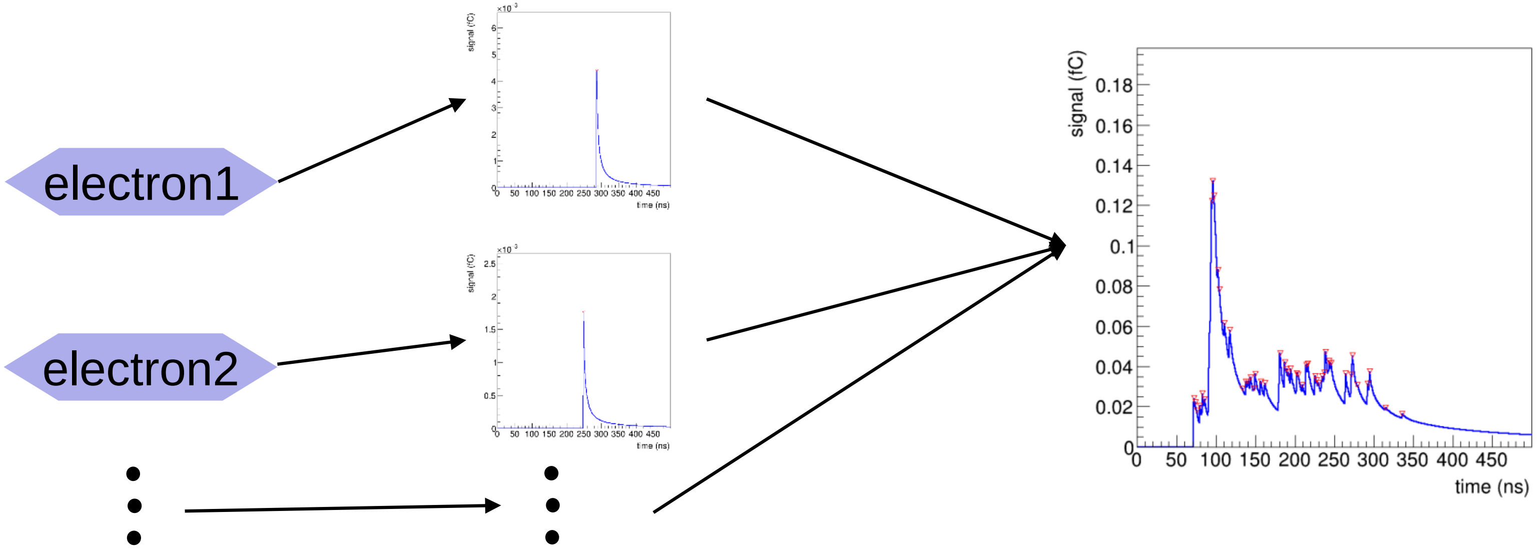


$$\frac{\mu(N_{\text{primary}}^{\text{G4+TrackHeed}})}{\mu(N_{\text{primary}}^{\text{Garfield++}})}$$

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

Signal waveform simulation

- From ionized electrons to signal waveform
  - 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 method (parameters are based on Garfield++ simulation results), will be much faster
    - For each electron, gives one spectrum
    - As done by Garfield++, piling up spectrums from same drift chamber cell gives waveform on the signal wire

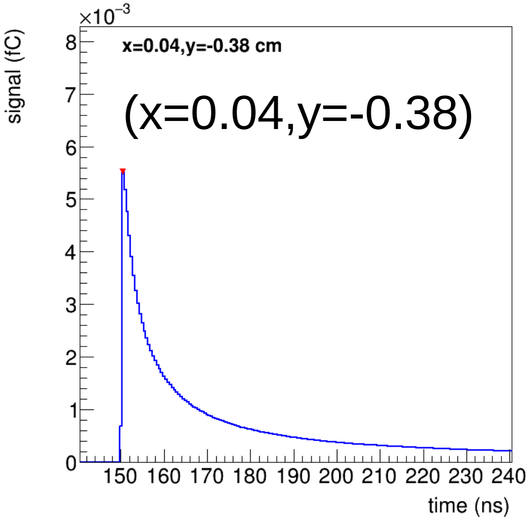


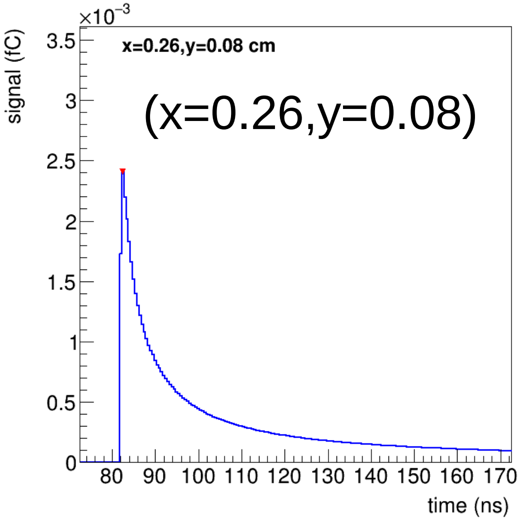
electron1

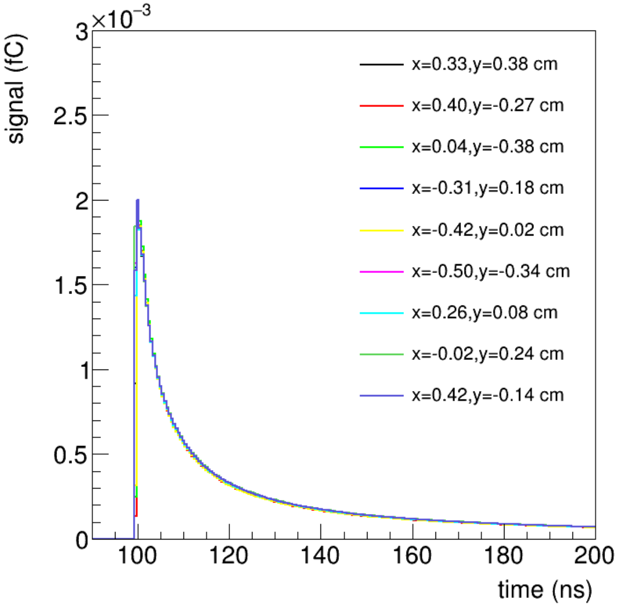
electron2

Getting parameters from Garfield++

- Performed some single electron simulations using Garfield++
- All spectrums are similar if their peak positions are shifted to same value (e.g. 100 ns) and the peak values are scaled to same value (e.g. 2×10^{-3})

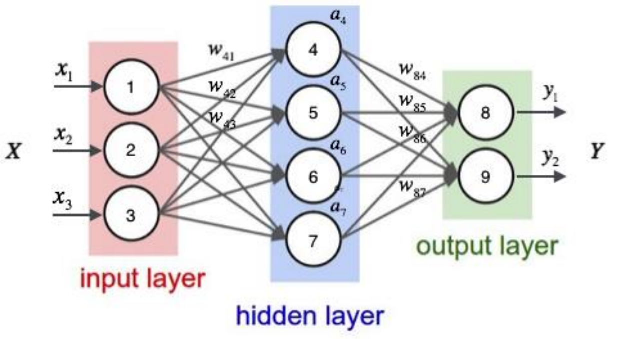






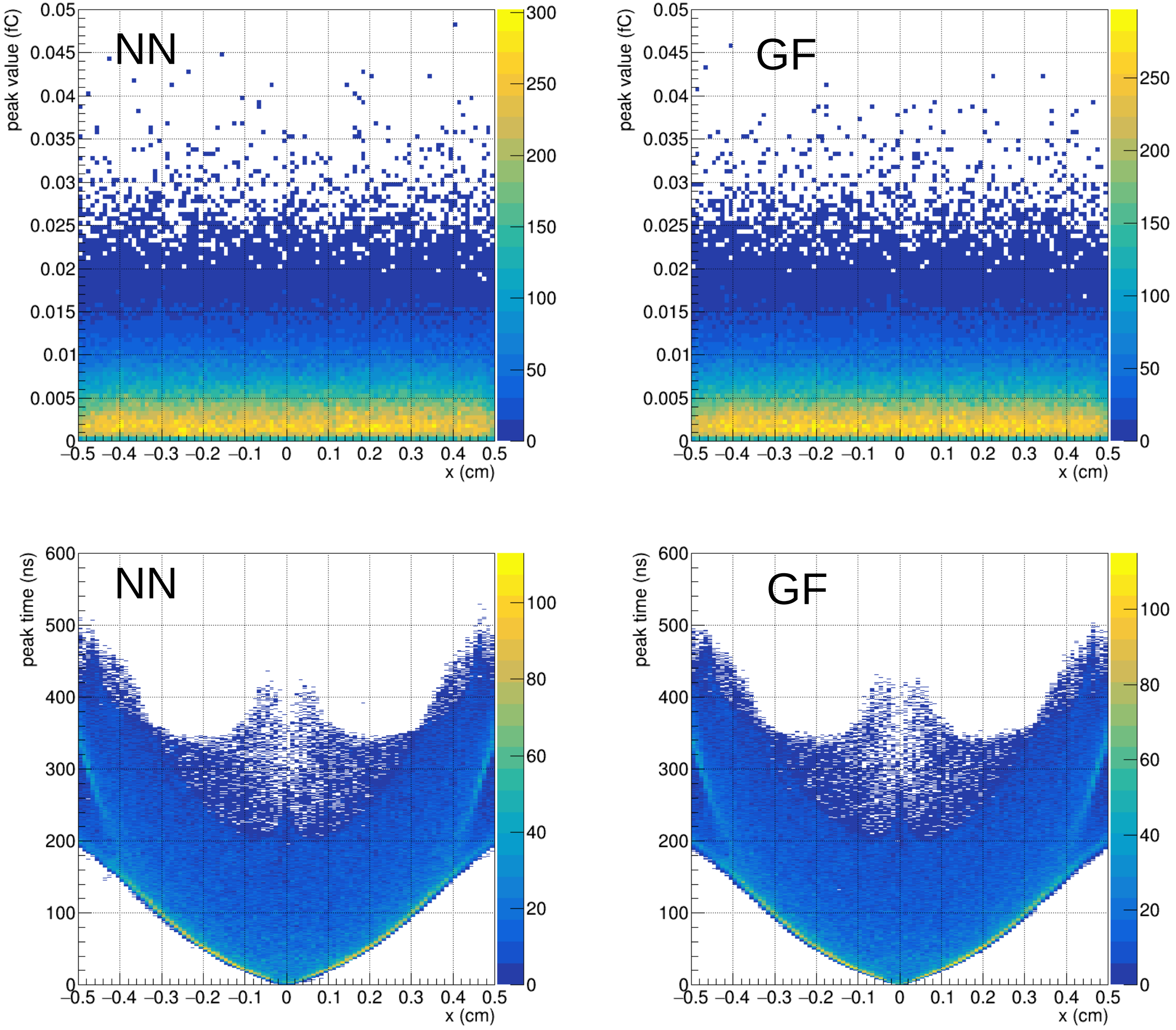
- Simulating single electron spectrum ≈ simulating (peak_time, peak_value) + using spectrum template

ML for peak time and value simulation



- Model: deep neutral network (DNN)
  - Consist of input, hidden, and output layers
- Input data:
  - Local x and y positions of ionized electrons
  - N(0,1) distribution noise
- Output: peak time and value of single electron spectrum
- Loss: two sample test statistics (SmoothKNN) between real data and generated data

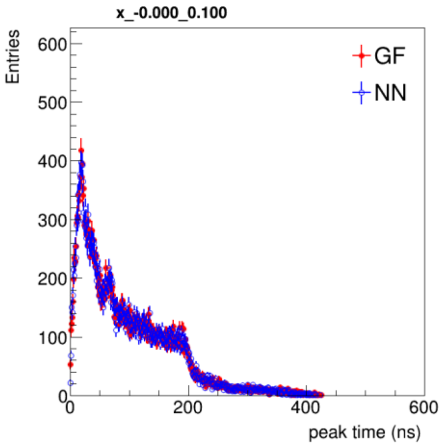
Garfield++ simulation

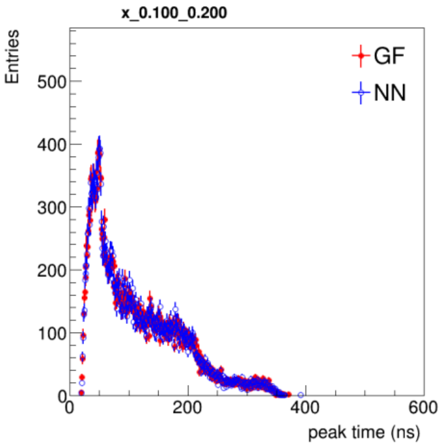


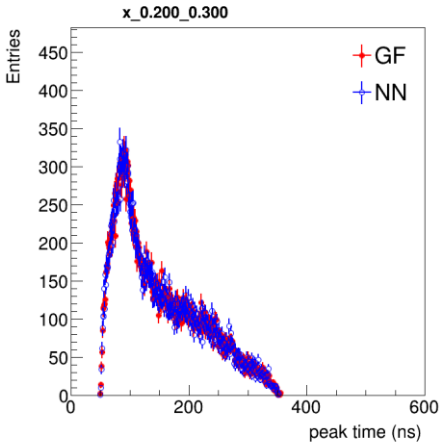
Peak value vs x

Peak time vs x

Peak time simulation



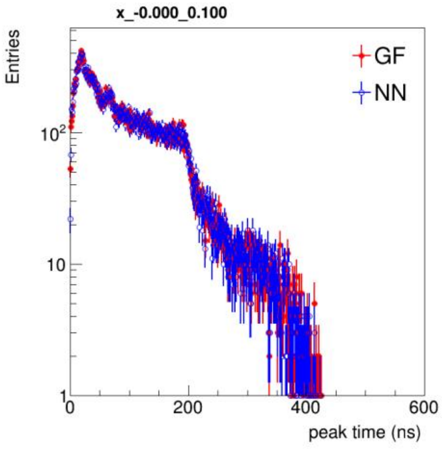


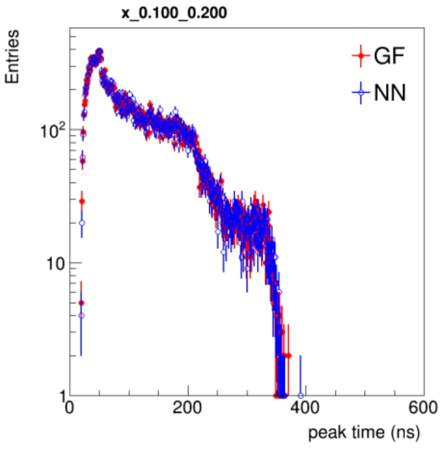


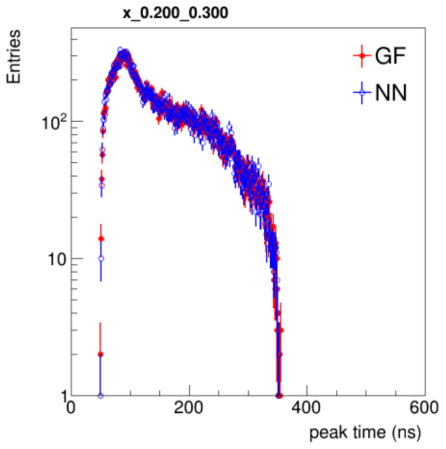
x(0,0.1) cm

x(0.1,0.2) cm

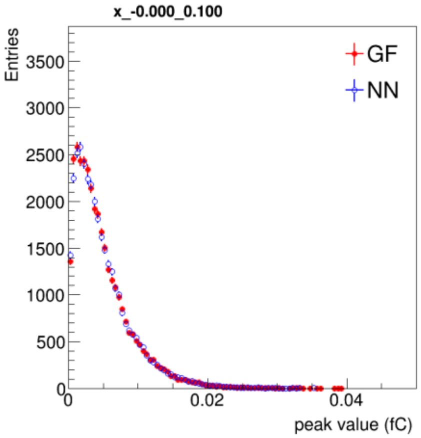
x(0.2,0.3) cm

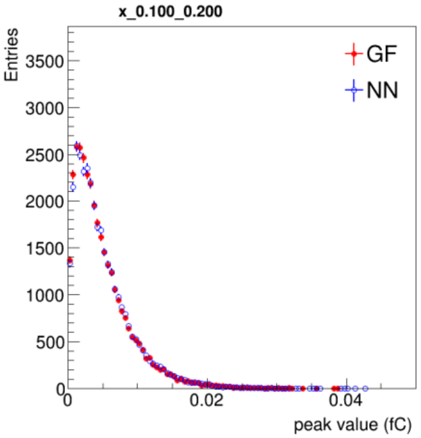


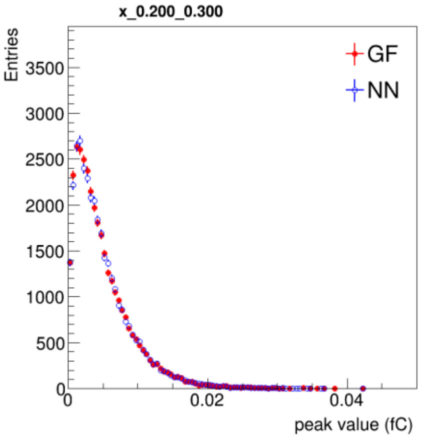




Peak value simulation



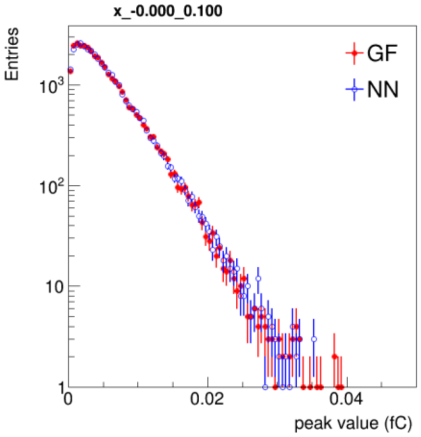


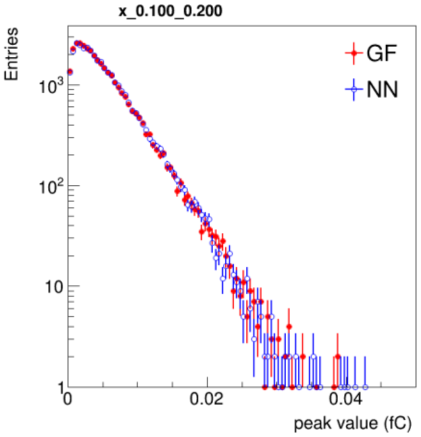


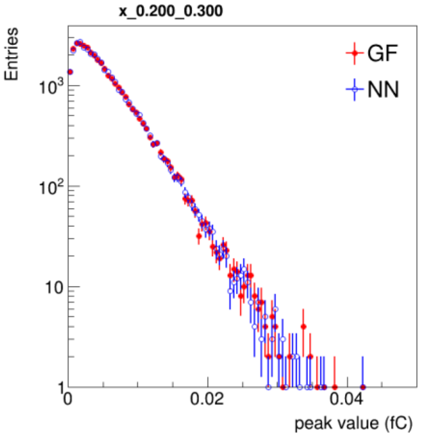
x(0,0.1) cm

x(0.1,0.2) cm

x(0.2,0.3) cm

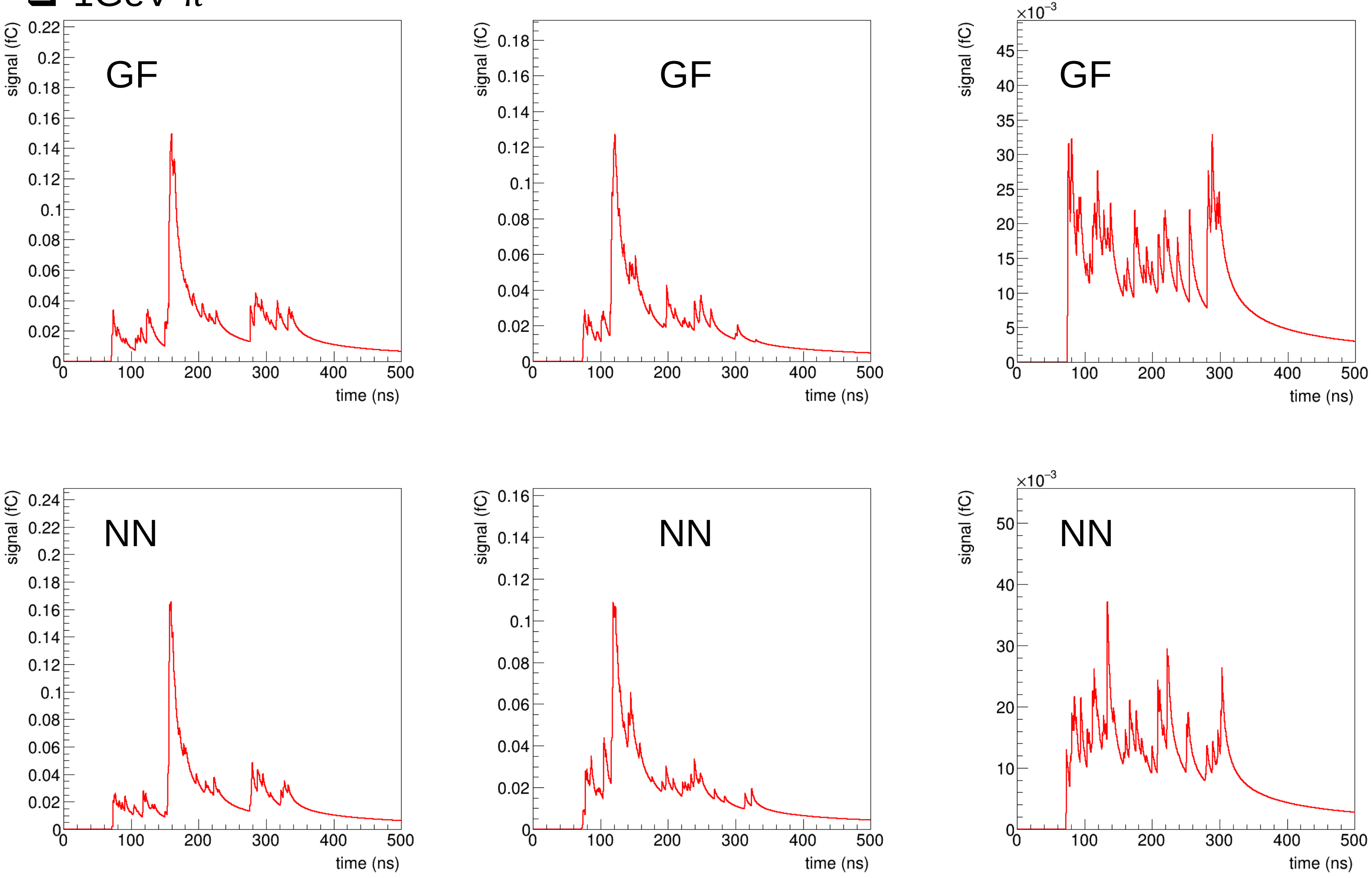






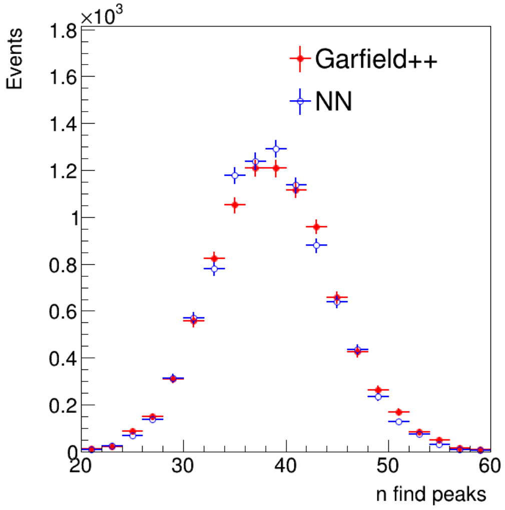
Signal waveform simulation

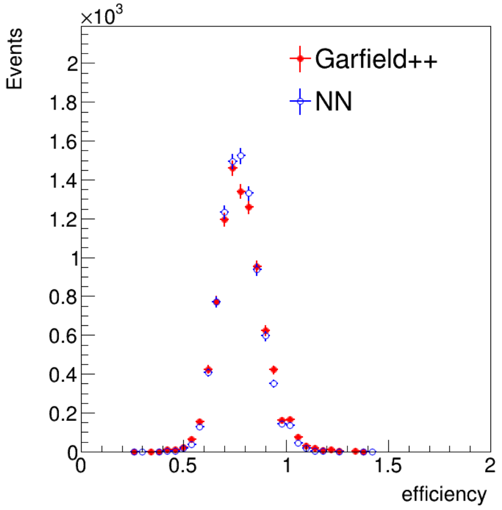
- 1GeV π^-

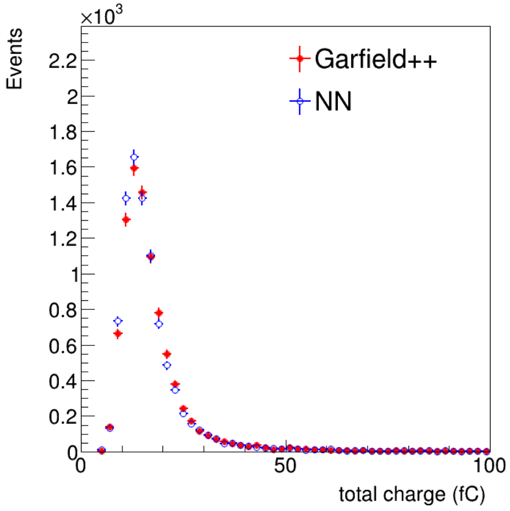


Signal waveform simulation

□ 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++

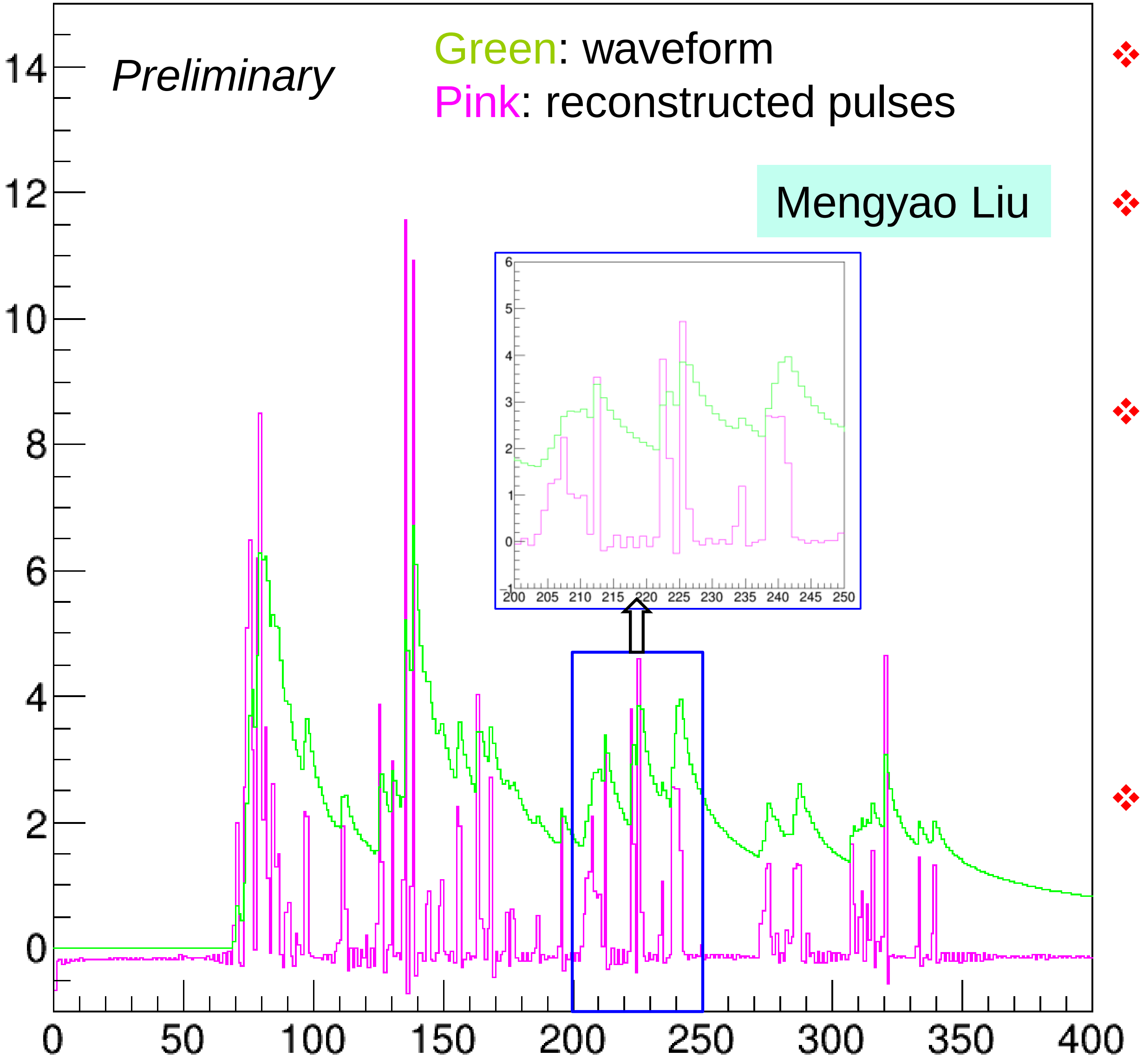
□ Average time for one cell simulation:

  □ Garfield++: ~250 s

  □ NN: ~1 s

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

Waveform reconstruction



- From waveform to ionized electrons
- Using waveform reconstruction method from JUNO experiment
- Idea is using Fourier transform to do the de-convolution(see backup) for the waveform and obtain pulse for each electron
- The preliminary results looks promising

Waveform reconstruction

- Using this method, we can analysis the waveform in frequency domain and filter high frequency electronic noise
- In CEPCSW, we have truth pulse information which is useful to study the performance of waveform reconstruction algorithm such as the fake rate of the reconstructed pulse
- Due to one primary ionization may produce more than one ionized electrons, it is needed to develop an algorithm to merge the reconstructed pulses which come from same primary ionization
- Getting the primary ionization counting efficiency
- Other primary ionization counting algorithms are welcome

dN/dx for a track

- Due to the production of δ electrons in the ionization process, the dN/dx from some cells could be effected by these δ electrons (very large dE/dx), this should be investigated
- For realistic case, each cell maybe different in background level (different layers), signal wire voltage, magnetic field and so on. The dN/dx calibration for each cell is needed
- Like dE/dx, we can use truncated mean method (remove some measurements with very small and large dN/dx) to get dN/dx for each track

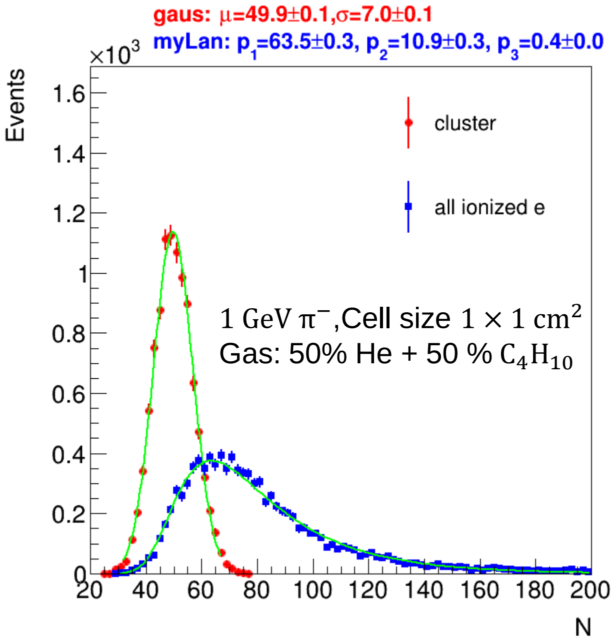
Summary and plan

- A new simulation scheme for drift chamber has been presented
- The ionization simulation using Geant4 combined with TrackHeed have been implemented in CEPCSW. Results are consistent with Garfield++ simulation
- In order to speed up the simulation of waveform, a parameterized method using ML has been described, which gives consistent results with Garfield++ simulation
- The de-convolution with Fourier transforms was introduced to waveform reconstruction, preliminary results are promising and further work is still in progress
- Future plan:
  - Studying waveform reconstruction by adding noise and electronics response
  - Development of the reconstruction algorithm for the dN/dx of a track
  - Development a data driven simulation method to get more realistic simulation if possible

Back up

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

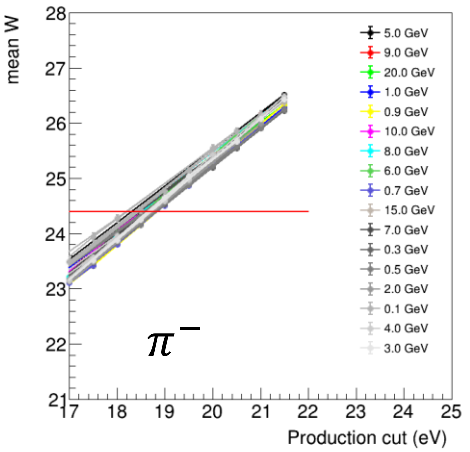


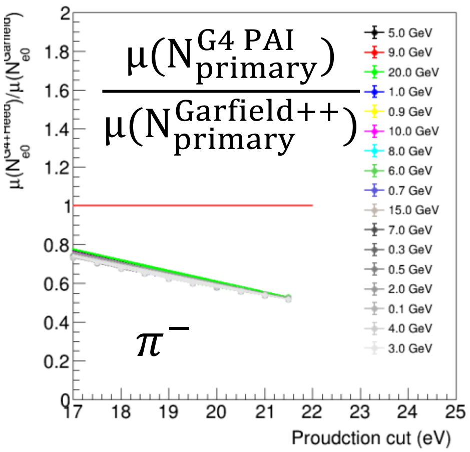
Ionization simulation in gas

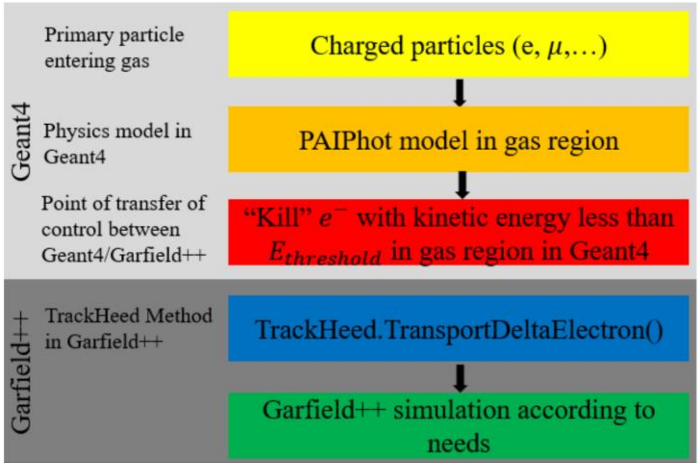
- Garfield++
  - Using Heed PAI model to simulate the ionization in gas precisely
  - Can simulate the drift and avalanche of electrons in gas
  - The drift of ions to cathode can be simulated
  - The induced current can be given
  - It is useful to study and characterize the properties of gas detector 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++.







Geant4

Primary particle entering gas — Charged particles (e, μ,…)

Physics model in Geant4 — PAIPhot model in gas region

Point of transfer of control between Geant4/Garfield++ — "Kill" e^- with kinetic energy less than $E_{threshold}$ in gas region in Geant4

Garfield++

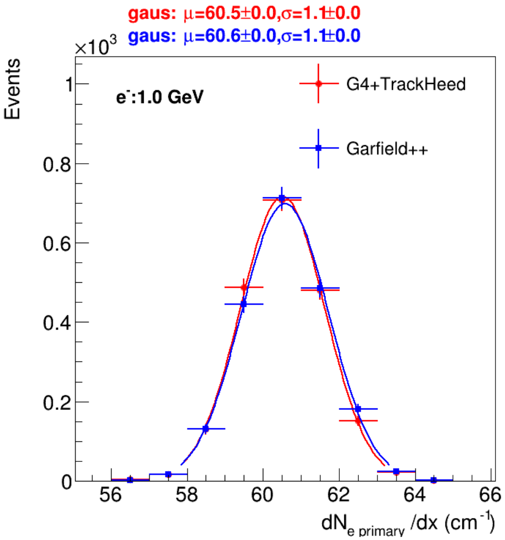
TrackHeed Method in Garfield++ — TrackHeed.TransportDeltaElectron()

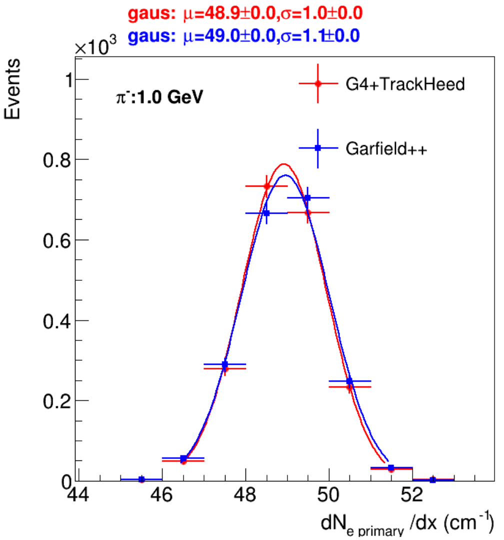
Garfield++ simulation according to needs

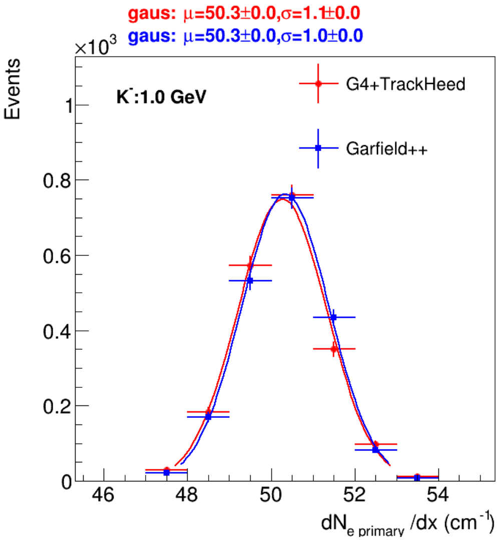
- Checking with authors:
  - This method designed to obtain correct energy deposition (or total ionizations)
  - It is true that this method will give less primary ionizations, so this method is obsoleted

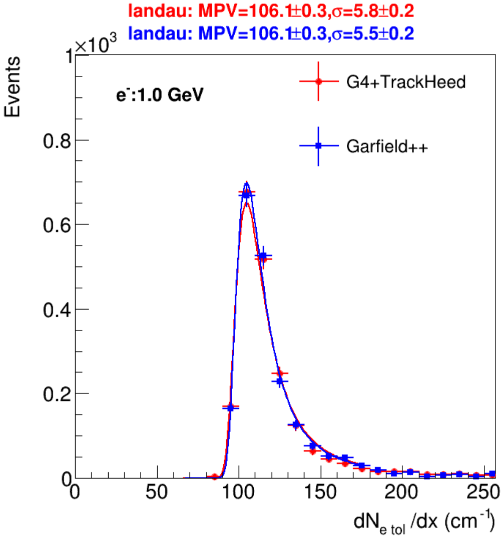
Ionization simulation performance

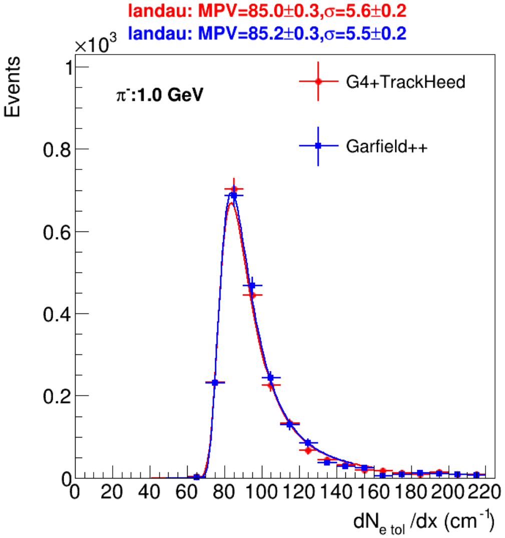
- Gas: 50% He + 50 % C_4H_{10}

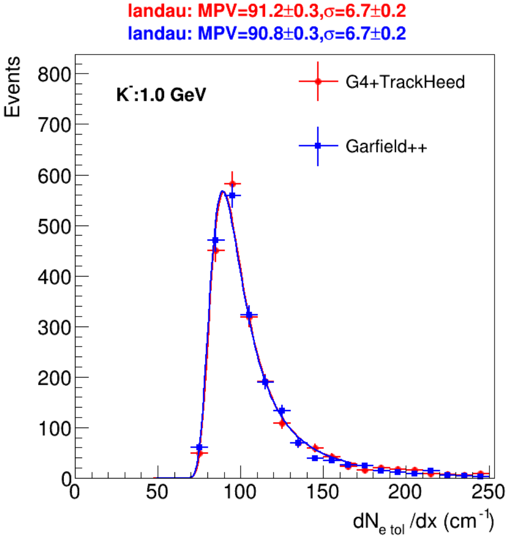








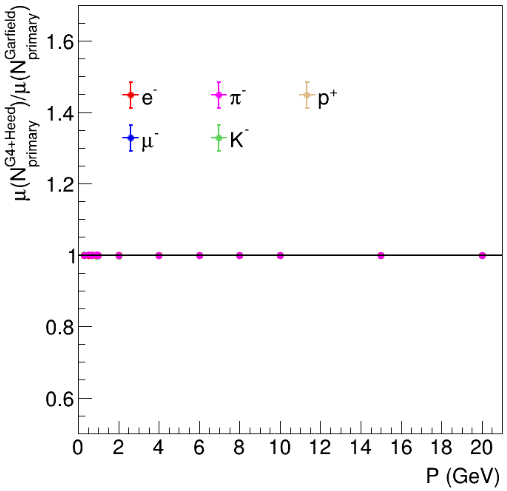


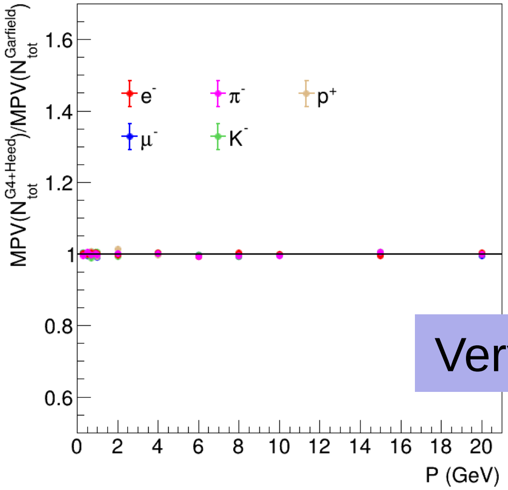


Ionization simulation performance

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

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

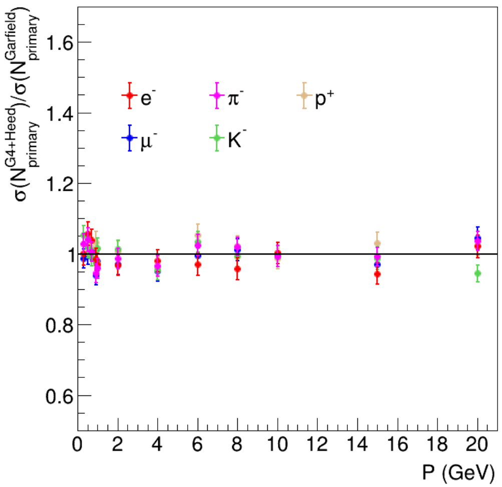


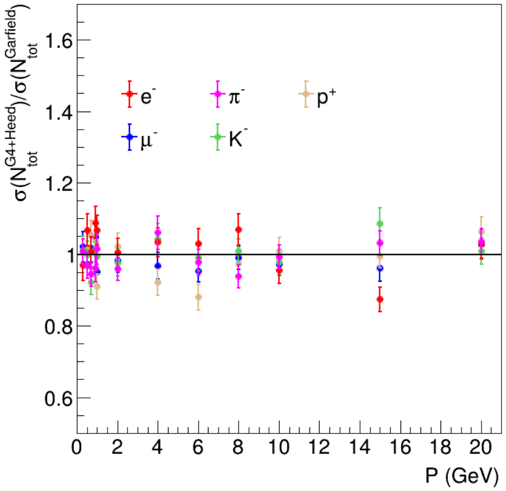


$$\frac{MPV(N_{tot}^{G4+TrackHeed})}{MPV(N_{tot}^{Garfield++})}$$

Very good agreement

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

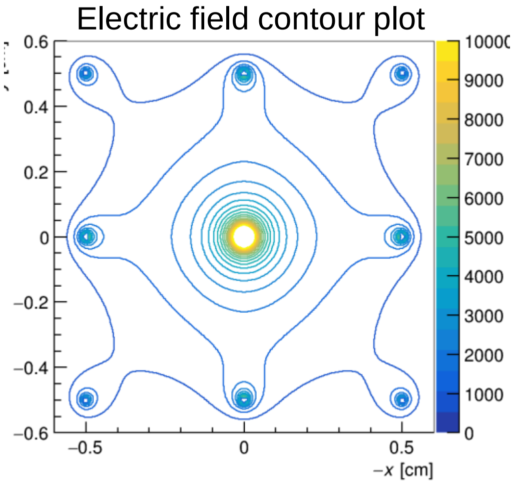


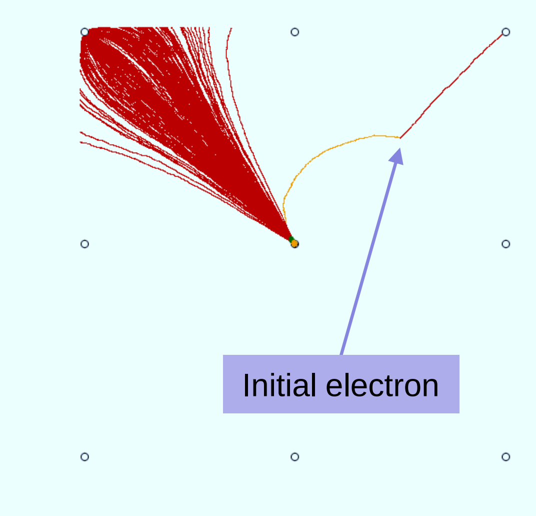


$$\frac{\sigma(N_{tot}^{G4+TrackHeed})}{\sigma(N_{tot}^{Garfield++})}$$

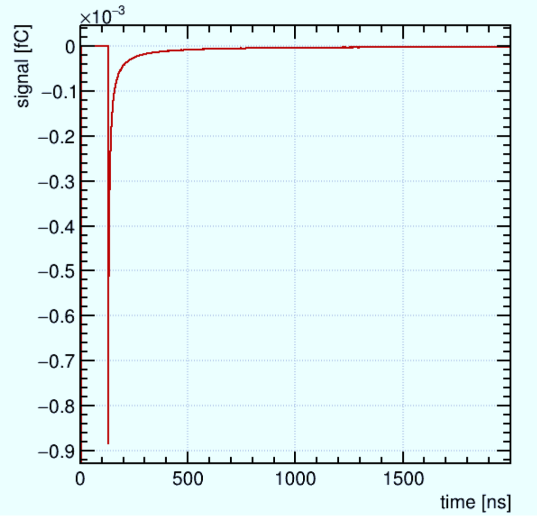
Getting parameters from Garfield++

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

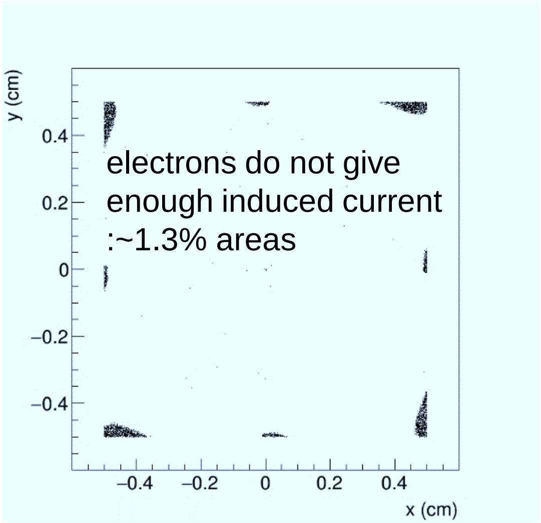

Electric field contour plot


Initial electron

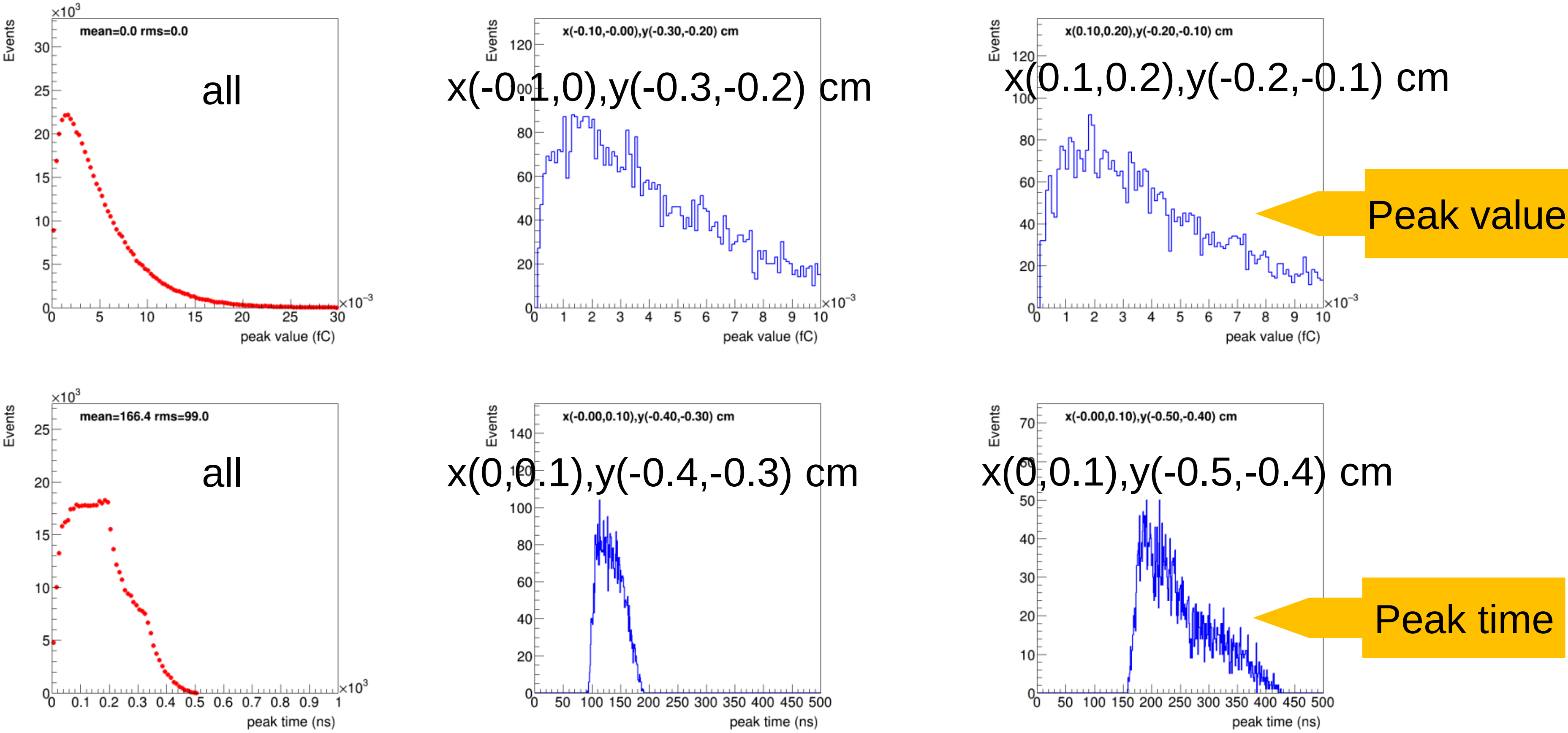
- One electron drift and avalanche
- Ions drift



Induced current


electrons do not give enough induced current :~1.3% areas

Garfield++ simulation



all

x(-0.1,0),y(-0.3,-0.2) cm

x(0.1,0.2),y(-0.2,-0.1) cm

Peak value

all

x(0,0.1),y(-0.4,-0.3) cm

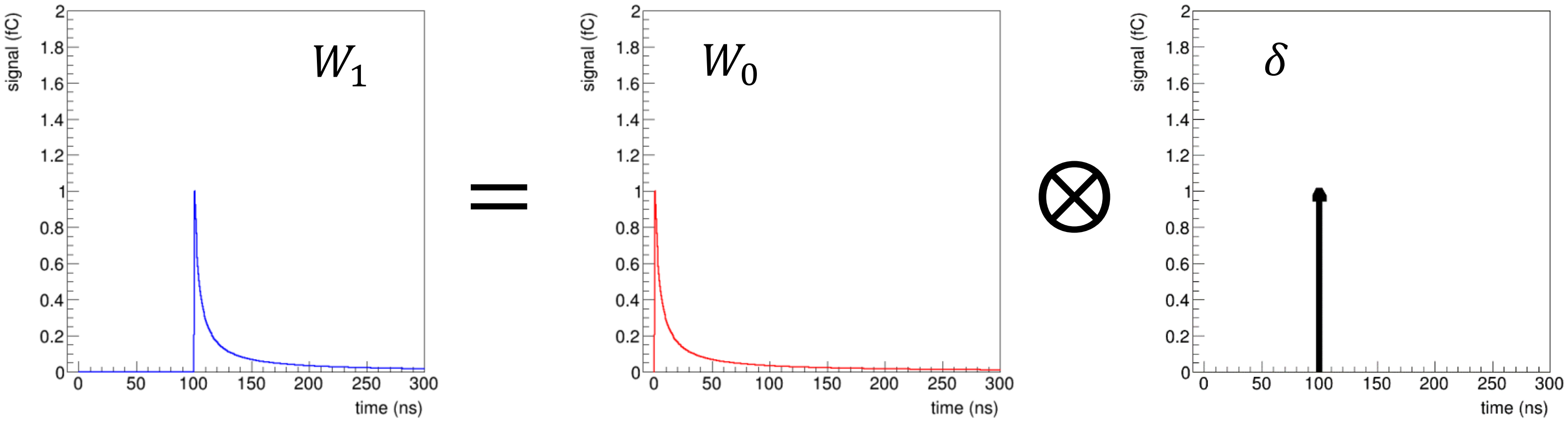
x(0,0.1),y(-0.5,-0.4) cm

Peak time

- 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

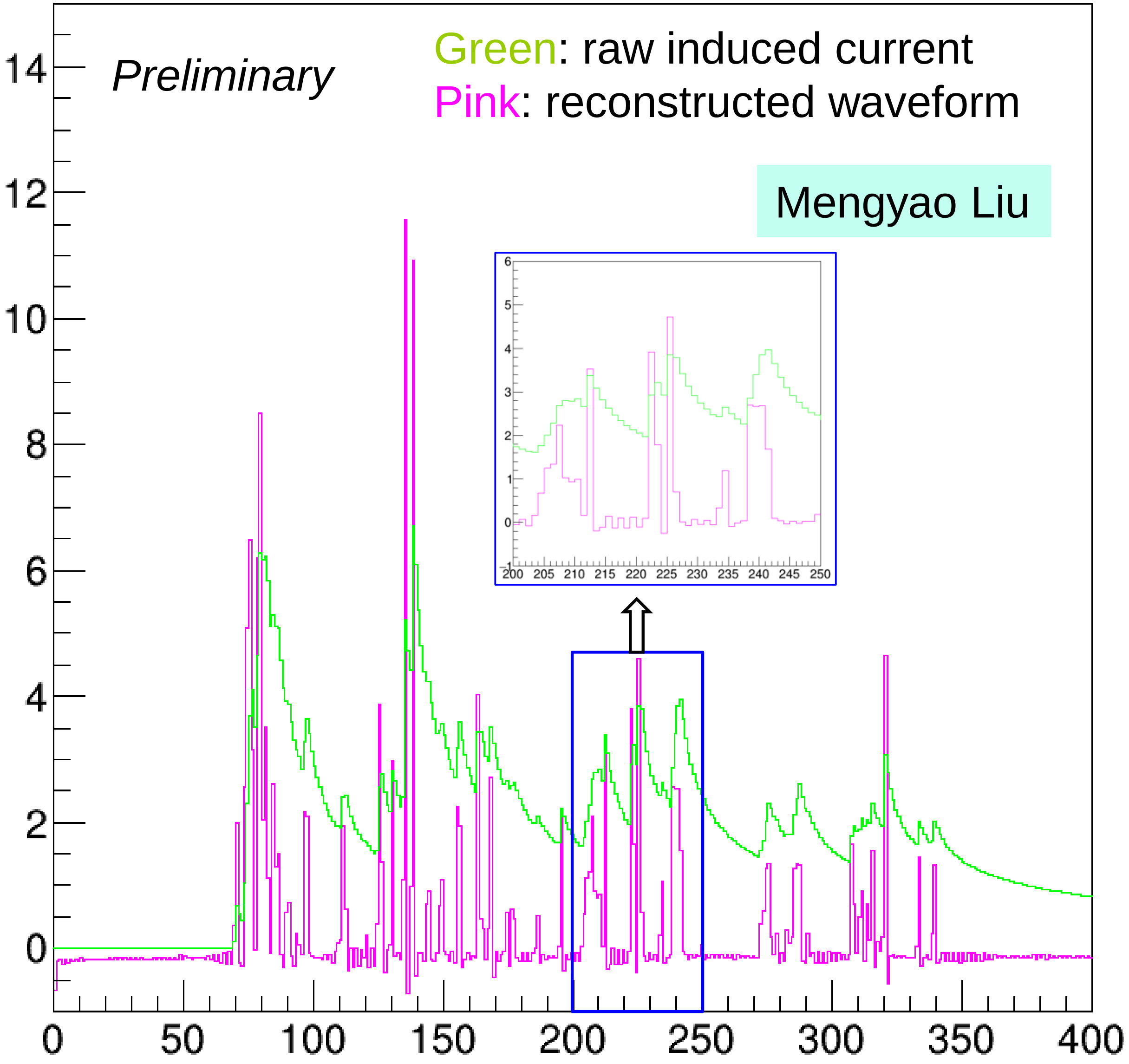
Waveform reconstruction

- Using the deconvolution method from JUNO experiment



- $F(W_1) = F(W_0 \otimes \delta) = F(W_0) * F(\delta)$
- $F(\delta) = F(W_1)/F(W_0)$
- $\delta = F^{-}(F(\delta))$, give information about peak time and value

Waveform reconstruction



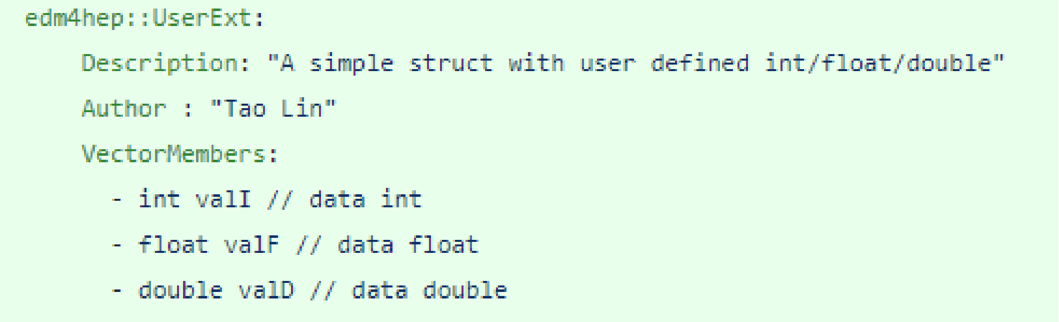
- The preliminary results looks promising
- Need further checks, such as cluster find efficiency, the purity of found clusters
- Using this method, we can analyse the components of the waveform and filter electronic noise
- Welcome for other algorithms

User extension data in EDM4hep

- As there is no waveform data format in EDM4hep yet, user extension data is a way to add additional data.
  - WIP: https://github.com/key4hep/EDM4hep/pull/117

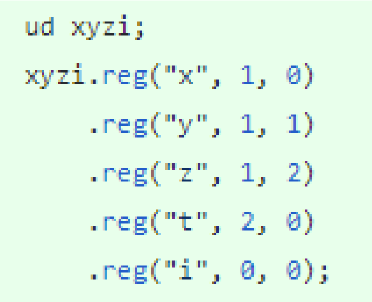
Tao Lin

The proposed underlying data structure:



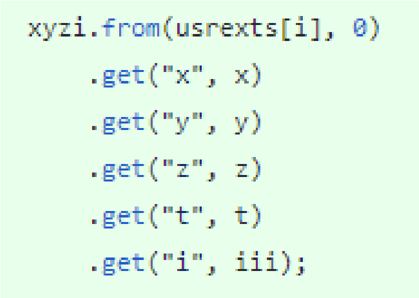
```
edm4hep::UserExt:
    Description: "A simple struct with user defined int/float/double"
    Author : "Tao Lin"
    VectorMembers:
      - int valI // data int
      - float valF // data float
      - double valD // data double
```

The proposed user APIs:



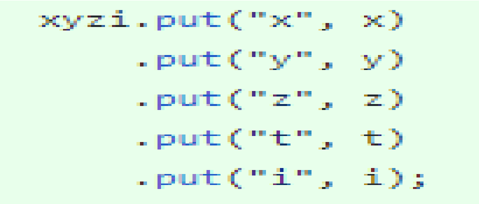
```
ud xyzi;
xyzi.reg("x", 1, 0)
    .reg("y", 1, 1)
    .reg("z", 1, 2)
    .reg("t", 2, 0)
    .reg("i", 0, 0);
```

Runtime Type definition

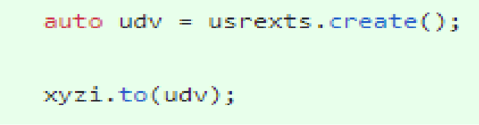


```
xyzi.from(usrexts[i], 0)
    .get("x", x)
    .get("y", y)
    .get("z", z)
    .get("t", t)
    .get("i", iii);
```

Getters



```
xyzi.put("x", x)
    .put("y", y)
    .put("z", z)
    .put("t", t)
    .put("i", i);
```



```
auto udv = usrexts.create();

xyzi.to(udv);
```

Setters