CEPC Jet@Clusters

Kaili Zhang

IHEP

zhangkl@ihep.ac.cn

Jet tasks to do

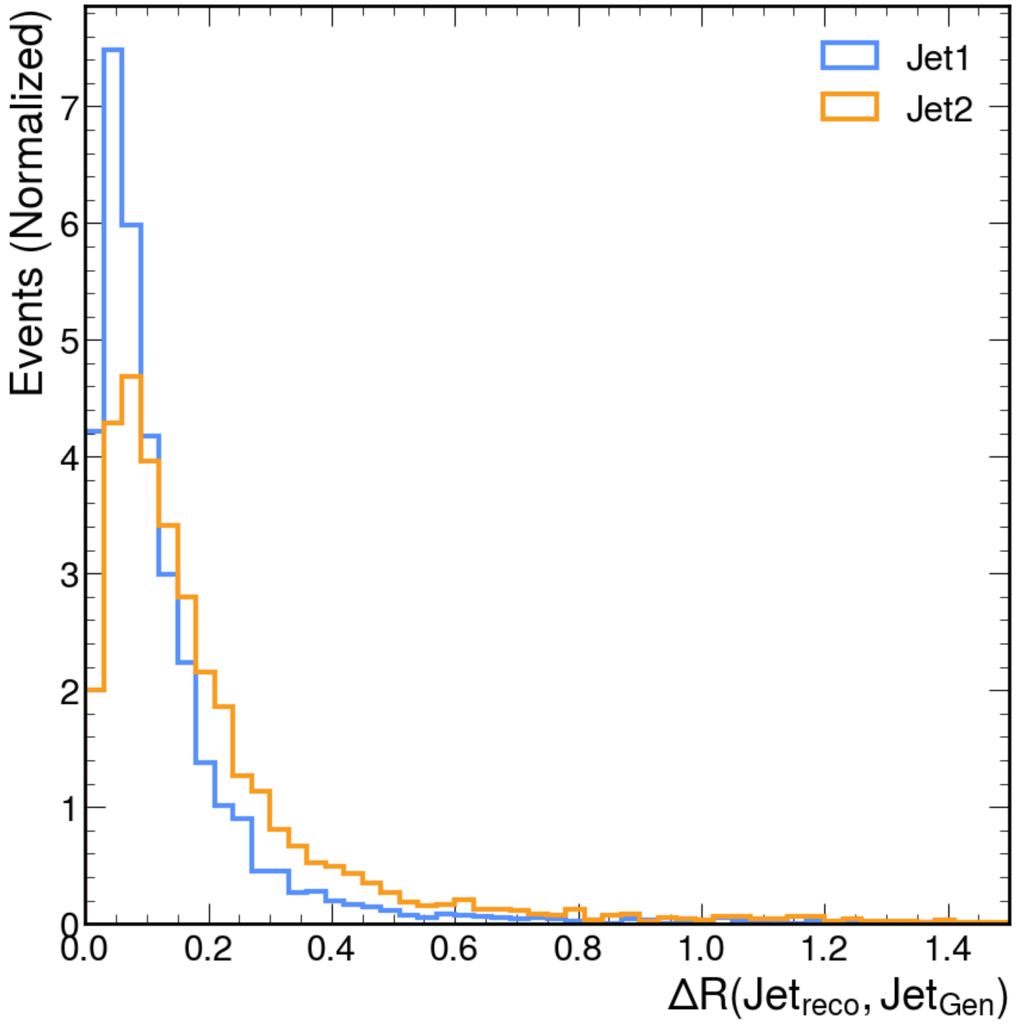
- Jet Sample Production
- Jet Gen Match
- JE, JA related plots @Yingqi
- BMR plots
- Neutral jet superclusters
- Particle gun one-type particle response See new MR.
- Remove isolated lepton/photon in PFO then jet clustering. @Danning
    - Need a quick PID

Tasks sub priority

- Jet Event display
- Validation ee-kt algorithm with others.
- Validation generator Whizard with others.
- Flavors/JOI
- Endcap jet performance
- Repeat Ecal/Hcal performance

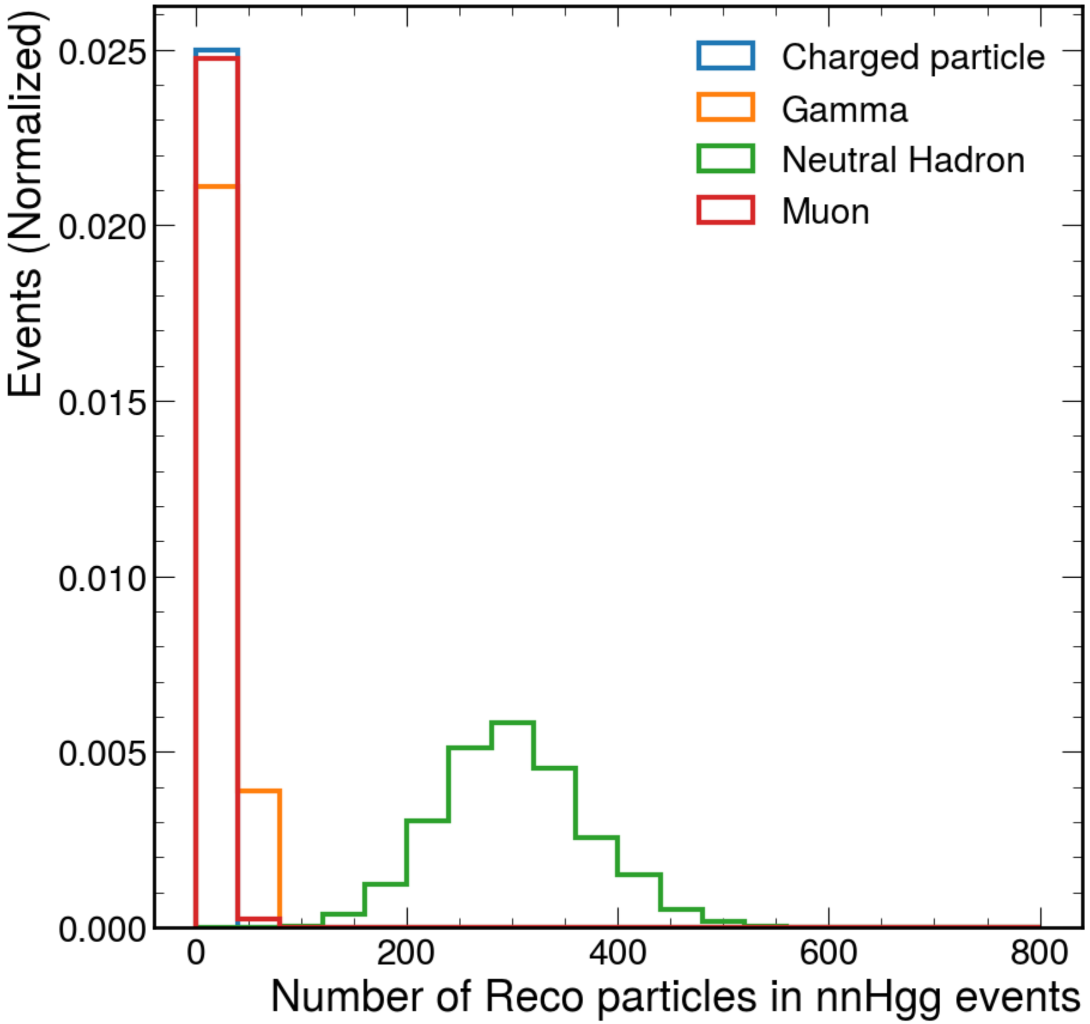
Current GenMatch

Why not peak at 0?



Neutral components

these neutral components in jets split to many parts.
Form one super cluster?



Jet samples

- /cefs/higgs/zhangkl/Production
  - 400k ZH->vvbb. Please feel free to play with them.
  - More under generation.

- Single Jet Gun available soon.

Machine Learnings on Jets

- P-CNN
  - https://scipost.org/10.21468/SciPostPhys.7.1.014
- Particle Flow Nerwork
  - https://arxiv.org/abs/1810.05165
  - CEPC@Xiaotian : https://arxiv.org/abs/2410.04465v2 @Next week.
- LundNet
  - https://doi.org/10.1007/jhep03(2021)052
- ParticleNet
  - Arxiv:1902.08570
  - https://github.com/hqucms/ParticleNet

ParticleTransformer

- https://arxiv.org/abs/2202.03772
- https://github.com/jet-universe/particle_transformer
- Platforms: https://github.com/hqucms/weaver-core
- Tutorial on CEPC: https://github.com/ZHUYFgit/CEPC-Jet-Origin-Identification
- Inputs from CEPCsoft: /cefs/higgs/zhangkl/AI/datasets
- Inputs from LHC, JetClass: /cefs/higgs/zhangkl/AI/jetclass
- Require higgsgpu group. Request on https://ccsinfo.ihep.ac.cn/
- Follow the tutorial, build the env if you are interested.

ParticleTransformer @ CEPC

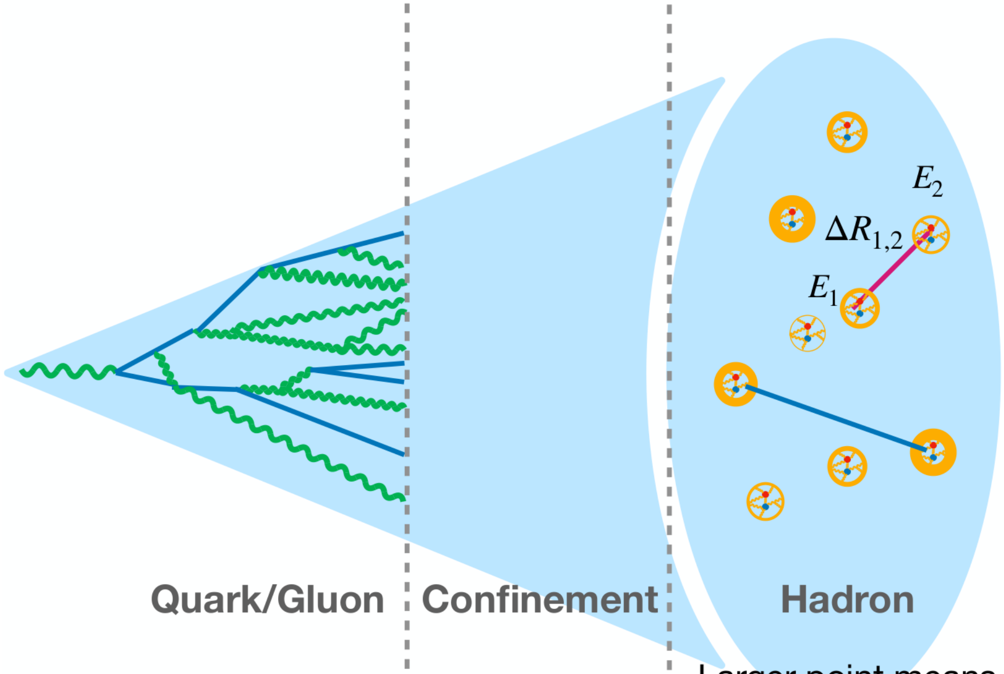
- Variable list in M11origin.cc
  - Under development to CEPCSW
  - Unit as one jet: 4 momentum, M11 id information……
- Train in Weaver: JetClass_full.yaml
- Submit jobs on IHEP: train_JetClass.sh
- Output: Pred.root: Label and score for each jets.
- Application: onnx format

New Inputs for machine learning

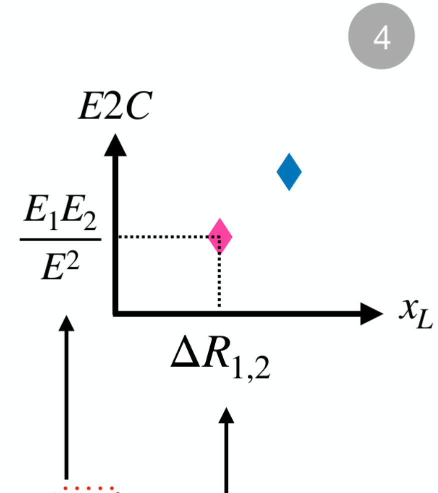
Energy Correlations:
https://arxiv.org/abs/1305.0007

[Xiao Meng's Slides]

Energy correlators: E2C



Larger point means higher energy



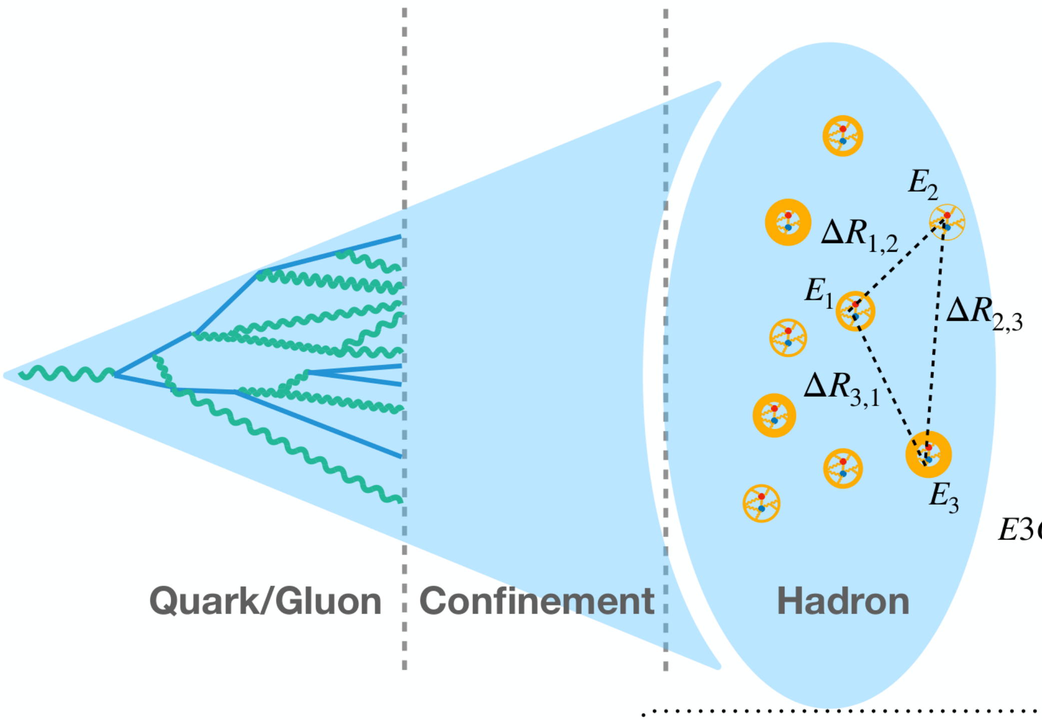
$$E2C = \frac{d\sigma^{[2]}}{dx_L} = \sum_{i,j}^{n} \int d\sigma \frac{E_i E_j}{E^2} \delta(x_L - \Delta R_{i,j})$$

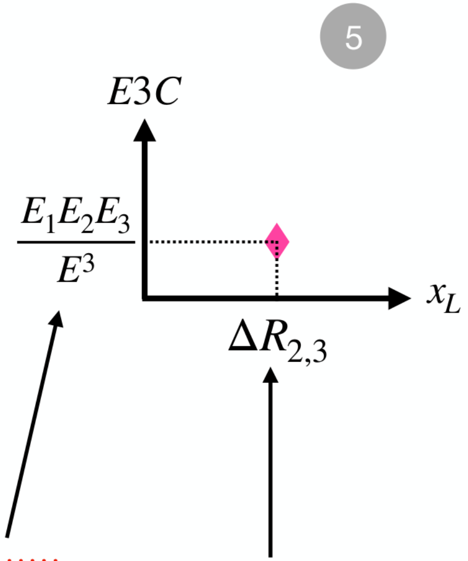
Collinear and infrared safe => calculable

New Inputs for machine learning

[Xiao Meng's Slides]

Energy correlators: E3C



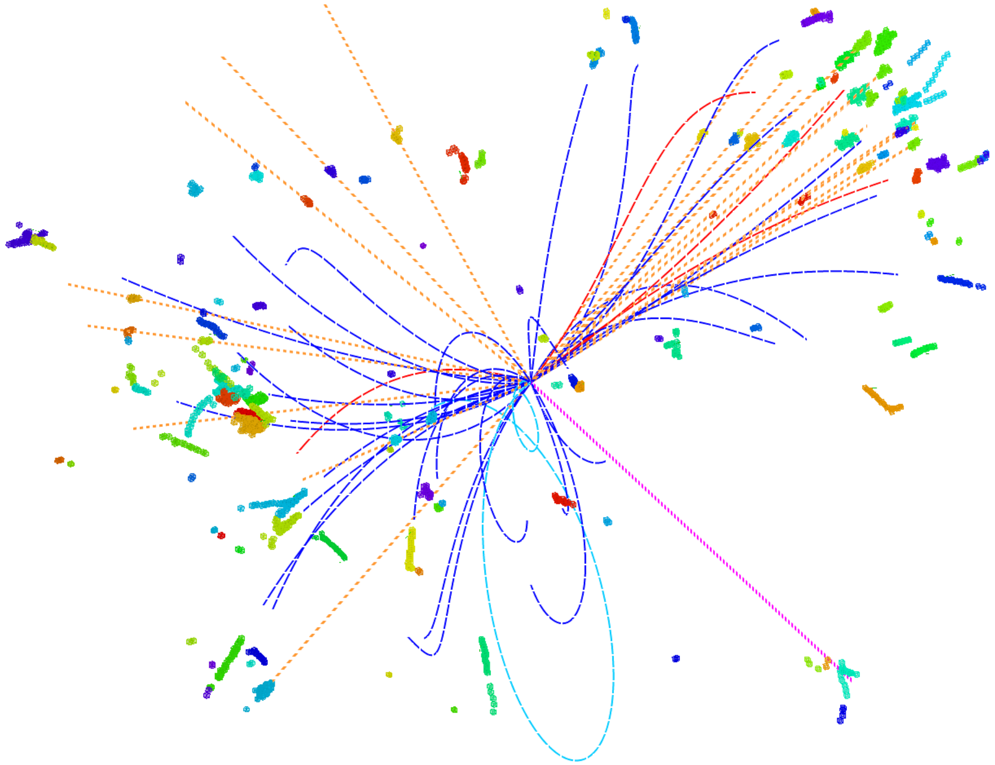


$$E3C = \frac{d\sigma^{[3]}}{dx_L} = \sum_{i,j,k}^{n} \int d\sigma \frac{E_iE_jE_k}{E^3} \delta(x_L - \max(\Delta R_{i,j}, \Delta R_{i,k}, \Delta R_{j,k}))$$

Initial proposal, Chen, Moult, Zhang, and Zhu, *arXiv:2004.11381*
NLO+NLL, Lee, Meçaj, and Moult, *arXiv:2205.03414*
NLO+NNLL$_{approx}$, Chen, Gao, Li, Xu, Zhang, and Zhu, *arXiv:2307.07510*

backups

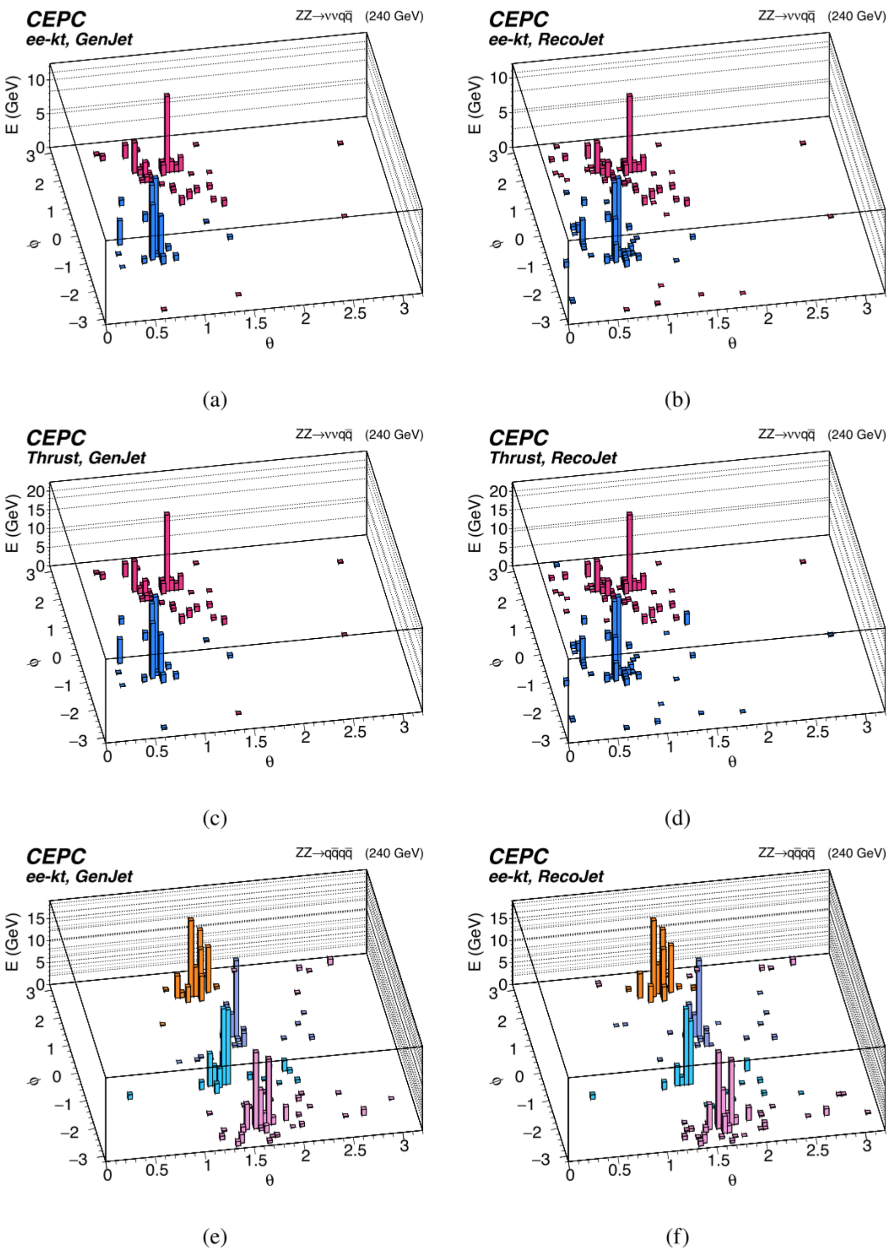
Jets 喷注



- Including varied components
- CEPC uses FastJet package to do jet clustering.
- Now, ee-kt/Durham algorithm used.
  - You need and only need to specify N_jets for Fastjet.
  - Generally, for all kt algos, 2 parameter: R and P can be adjusted.
  - ee-kt no R setting, so all clusters will be clustered.

$$d_{ij} = \min(E_i^{2p}, E_j^{2p}) \frac{(1-\cos\theta_{ij})}{(1-\cos R)},$$
$$d_{iB} = E_i^{2p},$$

Jet event display



(a) (b) (c) (d) (e) (f)

Jet @ CEPCSW

```
JetDefinition jet_def(ee_kt_algorithm);

ClusterSequence clust_seq(input_particles, jet_def);

vector<PseudoJet> jets = sorted_by_pt(clust_seq.exclusive_jets(nJets));
```

- Specify PFO container and njets.
- Not stored in final ntuple.
- In Zebing's Genmatch
  - Many variable stored.
  - Find jet_ntuple to extract informations

```
StatusCode GenMatch::initialize(){

  // Create a new TTree
  _file = TFile::Open(m_outputFile.value().c_str(), "RECREATE");
  _tree = new TTree("jets", "jets");
  _tree->Branch("jet1_px", &jet1_px, "jet1_px/D");
  _tree->Branch("jet1_py", &jet1_py, "jet1_py/D");
  _tree->Branch("jet1_pz", &jet1_pz, "jet1_pz/D");
  _tree->Branch("jet1_E", &jet1_E, "jet1_E/D");
  _tree->Branch("jet1_costheta", &jet1_costheta, "jet1_costheta/D");
  _tree->Branch("jet1_phi", &jet1_phi, "jet1_phi/D");
  _tree->Branch("jet1_pt", &jet1_pt, "jet1_pt/D");
  _tree->Branch("jet1_nconstituents", &jet1_nconstituents, "jet1_nconstituents/I");
  _tree->Branch("jet2_px", &jet2_px, "jet2_px/D");
  _tree->Branch("jet2_py", &jet2_py, "jet2_py/D");
  _tree->Branch("jet2_pz", &jet2_pz, "jet2_pz/D");
  _tree->Branch("jet2_E", &jet2_E, "jet2_E/D");
  _tree->Branch("jet2_costheta", &jet2_costheta, "jet2_costheta/D");
  _tree->Branch("jet2_phi", &jet2_phi, "jet2_phi/D");
  _tree->Branch("jet2_pt", &jet2_pt, "jet2_pt/D");
  _tree->Branch("jet2_nconstituents", &jet2_nconstituents, "jet2_nconstituents/I");
  _tree->Branch("constituents_E1tot", &constituents_E1tot, "constituents_E1tot/D");
  _tree->Branch("constituents_E2tot", &constituents_E2tot, "constituents_E2tot/D");
  _tree->Branch("mass", &mass, "mass/D");
  _tree->Branch("ymerge", ymerge, "ymerge[6]/D");
  _tree->Branch("nparticles", &nparticles, "nparticles/I");
  _tree->Branch("jet1_GENMatch_id", &jet1_GENMatch_id, "jet1_GENMatch_id/I");
  _tree->Branch("jet2_GENMatch_id", &jet2_GENMatch_id, "jet2_GENMatch_id/I");
  _tree->Branch("jet1_GENMatch_mindR", &jet1_GENMatch_mindR, "jet1_GENMatch_mindR/D");
  _tree->Branch("jet2_GENMatch_mindR", &jet2_GENMatch_mindR, "jet2_GENMatch_mindR/D");

  _tree->Branch("PFO_Energy_muon", &PFO_Energy_muon);
  _tree->Branch("PFO_Energy_muon_GENMatch_dR", &PFO_Energy_muon_GENMatch_dR);
  _tree->Branch("PFO_Energy_muon_GENMatch_ID", &PFO_Energy_muon_GENMatch_ID);
  _tree->Branch("PFO_Energy_muon_GENMatch_E", &PFO_Energy_muon_GENMatch_E);
  _tree->Branch("PFO_Energy_Charge", &PFO_Energy_Charge);
  _tree->Branch("PFO_Energy_Charge_Ecal", &PFO_Energy_Charge_Ecal);
  _tree->Branch("PFO_Energy_Charge_Hcal", &PFO_Energy_Charge_Hcal);
  _tree->Branch("PFO_Energy_Charge_GENMatch_dR", &PFO_Energy_Charge_GENMatch_dR);
  _tree->Branch("PFO_Energy_Charge_GENMatch_ID", &PFO_Energy_Charge_GENMatch_ID);
  _tree->Branch("PFO_Energy_Charge_GENMatch_E", &PFO_Energy_Charge_GENMatch_E);
  _tree->Branch("PFO_Hits_Charge_E", &PFO_Hits_Charge_E);
  _tree->Branch("PFO_Hits_Charge_R", &PFO_Hits_Charge_R);
  _tree->Branch("PFO_Hits_Charge_theta", &PFO_Hits_Charge_theta);
  _tree->Branch("PFO_Hits_Charge_phi", &PFO_Hits_Charge_phi);

  _tree->Branch("PFO_Energy_Neutral", &PFO_Energy_Neutral);
  _tree->Branch("PFO_Energy_Neutral_singleCluster", &PFO_Energy_Neutral_singleCluster);
  _tree->Branch("PFO_Energy_Neutral_singleCluster_R", &PFO_Energy_Neutral_singleCluster_R);
  _tree->Branch("PFO_Hits_Neutral_E", &PFO_Hits_Neutral_E);
  _tree->Branch("PFO_Hits_Neutral_R", &PFO_Hits_Neutral_R);
  _tree->Branch("PFO_Hits_Neutral_theta", &PFO_Hits_Neutral_theta);
  _tree->Branch("PFO_Hits_Neutral_phi", &PFO_Hits_Neutral_phi);
```

Jet performance parameters

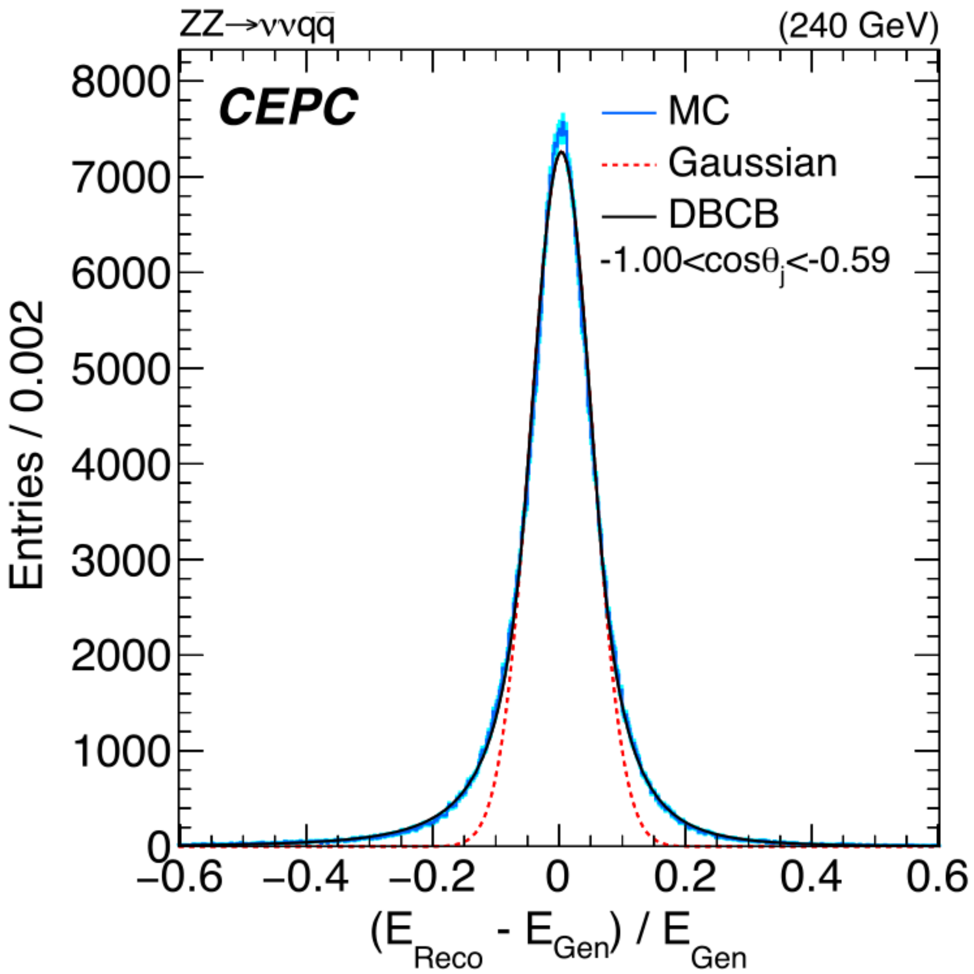
Peizhu, JINST

$$R_{R-G} = \frac{E_{\text{RecoJet}} - E_{\text{GenJet}}}{E_{\text{GenJet}}}$$

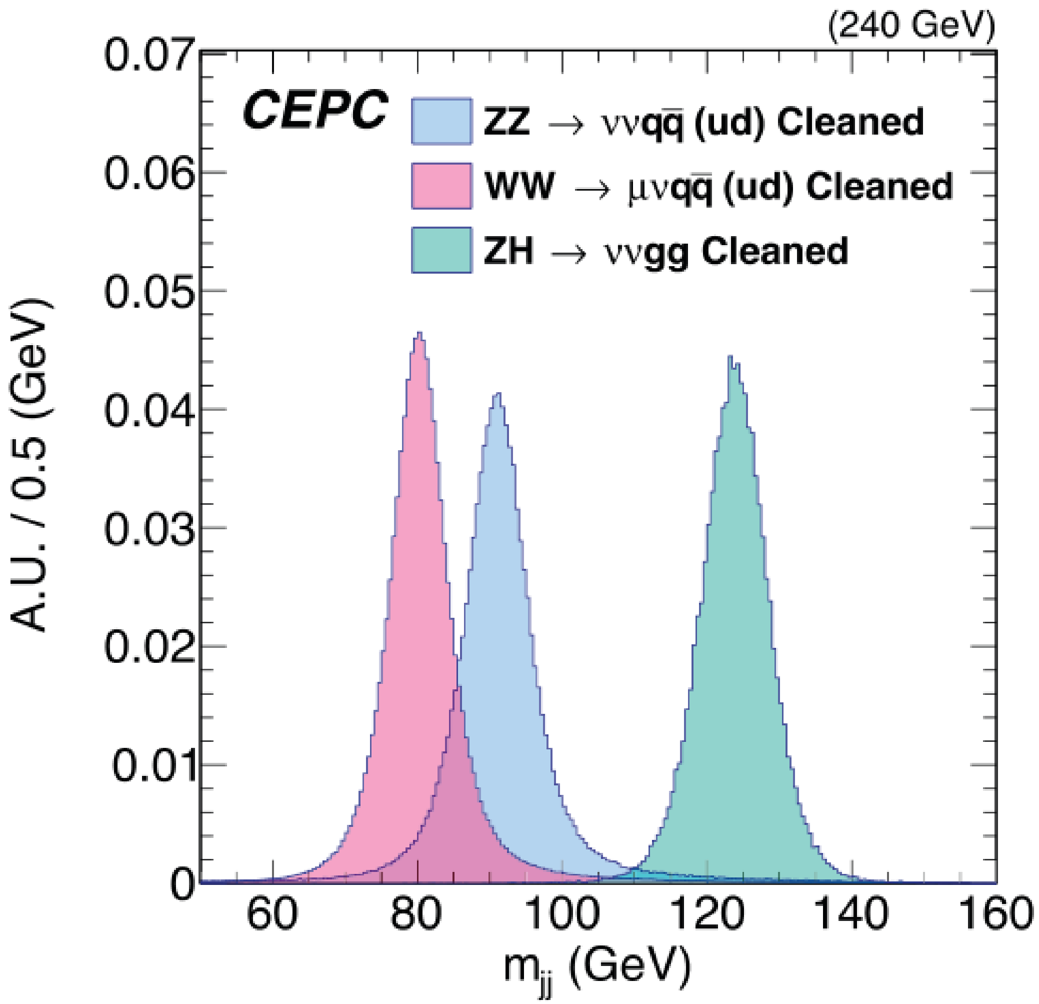
- After Reco-Gen matching,

In this difference plot (DSCB fit)

- JES Jet energy scale
  - Mean value shifted ($\bar{x}$)
- JER Jet energy resolution
  - Standard deviation (σ)



Jet performance parameters



- BMR(Boson mass resolution).
- ~Jet energy resolution.
- In CDR, when calculating BMR:
  - Veto total ISR components Pt > 1GeV;
  - Veto total neutrino Pt > 1GeV;
    - ISR and neutrino from single jet from Higgs.
  - Require Costheta_Jet;
- Current CEPCSW no endcap calo;
  - Require Costheta_Jet<0.65 (under tuning)

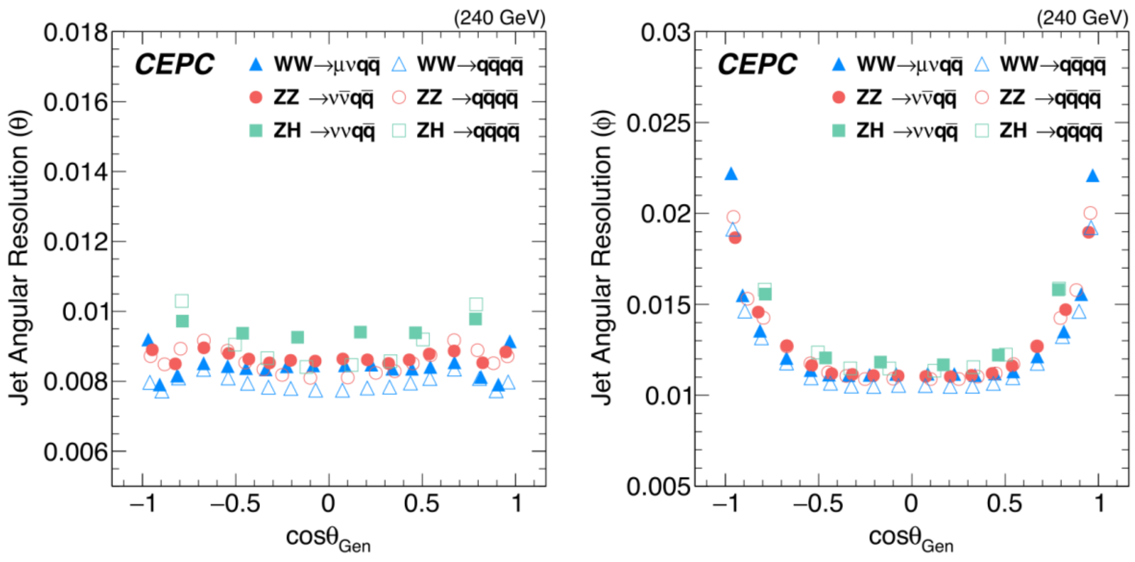
Table 1. Event cumulative efficiency for Higgs boson exclusive decay at the CEPC with $\sqrt{s} = 240$ GeV.

	gg(%)	bb(%)	cc(%)	WW^*(%)	ZZ^* (%)
Pt_ISR < 1 GeV	95.15	95.37	95.30	95.16	95.24
Pt_neutrino < 1 GeV	89.33	39.04	66.36	37.46	41.39
|Cos(Theta_Jet)| < 0.85	67.30	28.65	49.31	–	–

Jet angular performance

$$D_{R-G} = \theta_{\text{RecoJet}} - \theta_{\text{GenJet}} \quad \text{or} \quad \phi_{\text{RecoJet}} - \phi_{\text{GenJet}}$$

- JAR(θ, ϕ): Standard deviation (σ)
- JAS(θ, ϕ): Mean value shifted ($\bar{x}$)



(a) (b)

Most of the plots and performance need re-check under current CEPC ref-TDR.
Both for performance study and sanity check.

Flavor information

- Use traditional LCFIplus package, or ML training like ParticleTransformer, jet flavor information can be tagged.
- ML shows better performance
- Need migration.