# A general frame work based on fast and full simulation for CEPC analysis

**Gang LI**
**IHEP**
**CEPC physics & software workshop 2018.06.27-29**

# Outline

- Motivation
- Class design/Objects in fast and full simulation
- Examples
- Summary

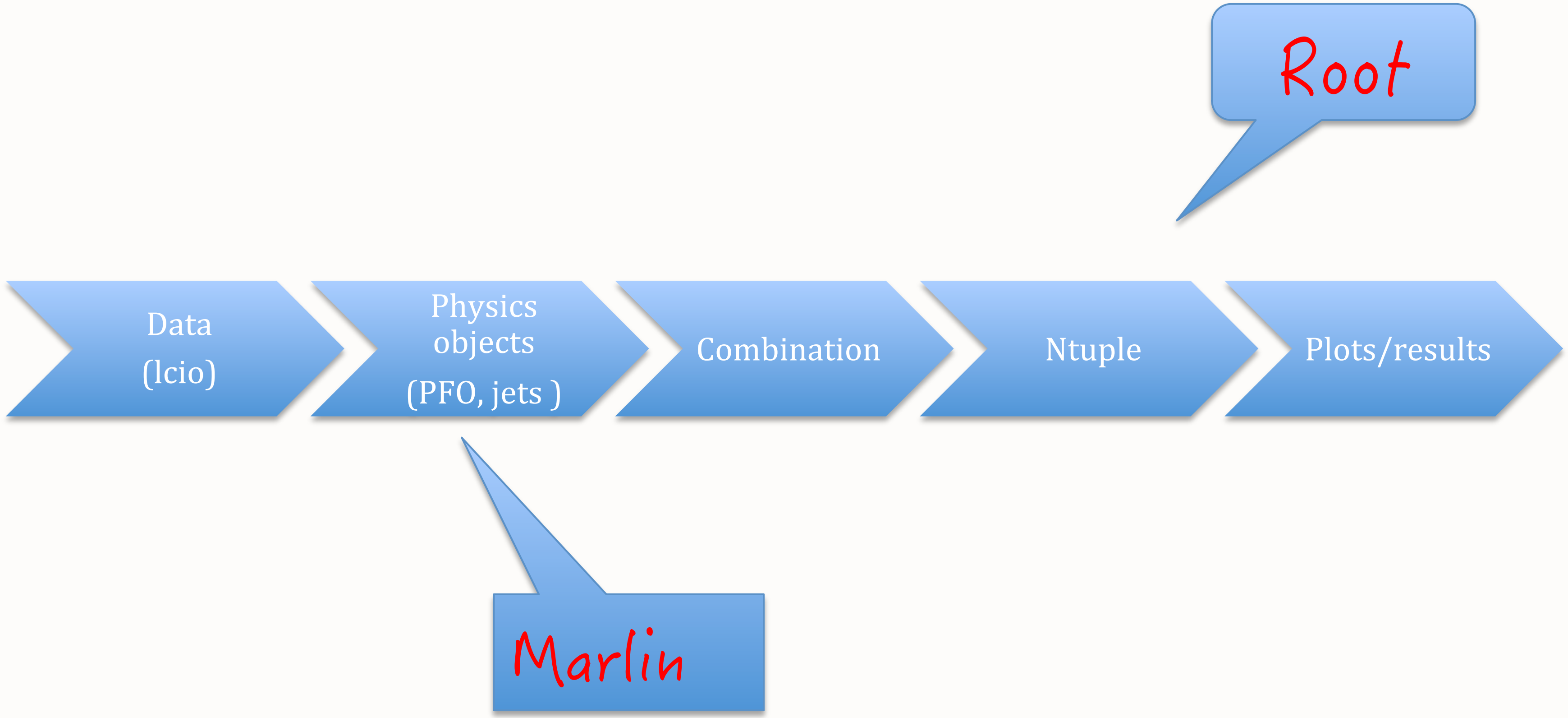
# Motivation

- **C++ is object-oriented**
- **Analysis procedures can be abstracted as some well designed classes in c++**
- **Well top-design has lots of advantages:**
  - ◆**Simple**
  - ◆**Generic**
  - ◆**Reusable**
  - ◆**Easy to standardize**
  - ◆**Easy to debug**
  - ◆**Easy for further work --- combination**
  - ◆ **… …**
  - ◆**Not need coding any more …**

# What we do in data-analysis?

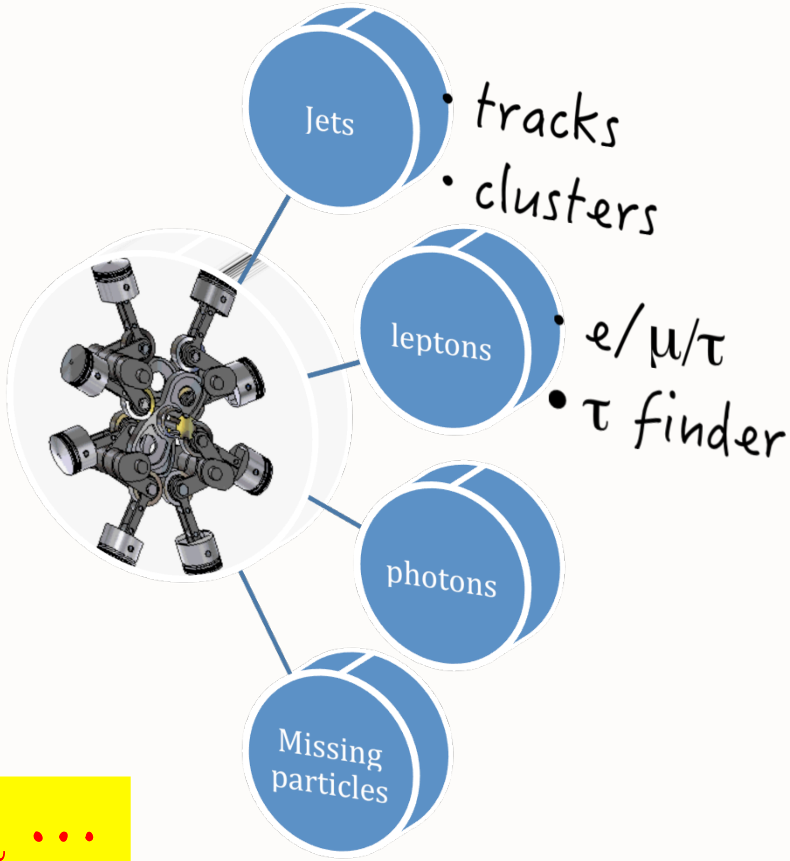
- Analysis
  - Some specific final state as our signal
  - Backgrounds
    - Same final states – keep it for further treatment  and avoid bias
    - Different final states – try to reject
- Three stages:
  - Particle level: track, neutral clusters, PFO objects, tau, jet …
  - Event level: combination of the particles-kinematic constraints
  - Save information: Ntuple & Tree
- Useful utilities
  - MC truth: tag event type/topology, check resolution, validate your selection
  - Kinematic fit
  - General variables: yij, thrust, sphericity, …
  - …

# Data-analysis (cont'd )



# Data-analysis (cont'd )

Feed all types of particle lists to the **combination engine** for further processing




$ee+X$, $\mu\mu+X$, $jj+ee$, $jj+\mu\mu$ ⋯

# Abstracting

- Class FSParticle → particles & lists
- Class FSinfo → all kinds of combination
- Class NTupleHelper → Ntuple service
- Class MCTruthHelper → MC truth service
- Class FSCut → simple cuts

# FSPaticle  -- physics objects

- **Data**
  - PID/Mass/charge/ 4-momentum, p, pT  …
  - Flavor/vertex
  - Matched MC object

```
ReconstructedParticle *          m_pfo;
MCParticle *                     m_mcp;
//JetFitObject*                    m_JetFitObject;


string   m_name;
int      m_type;
int      m_pdgid;
bool     m_missed;
bool     m_fast;
double   m_mass;
double   m_recmass;
double   m_charge;
double   m_pT;
double   m_pZ;
double   m_Energy;
double   m_Rapidity;
double   m_CosTheta;
double   m_btag;
double   m_ctag;
double   m_bctag;
double   m_flavor;

TLorentzVector       m_rawFourMomentum;
TLorentzVector       m_fitFourMomentum;

vector<int>          m_trackId;
vector<int>          m_showerId;
```

# FSInfo– event candidate the combination of particles

- Data
  - Daughter particles
  - Matched MC truth
    - Matched Parton
  - Cuts
    - Reduce root file size
  - Kinematic fit
    - Improve resolutions
  - Tree
    - All useful info. Saved

```
private:
   string m_FSName;
   vector<string> m_particleNames;
   vector<int>    m_particleStatus;
   int            m_nChargedParticles;
   int            m_nMissingParticles;
   NTupleHelper*  m_NT;
   NTupleHelper*  m_NTGen;

   int m_decayCode1;
   int m_decayCode2;

   bool   m_fast;

   bool   m_Constrain4Mom ;
   bool   m_missingMassFit;
   double m_missingMassValue;
   string m_missedParticle;

   vector< vector<unsigned int> >& submodeIndices(const string& submodeN

   vector<FSCut*>                                  m_FSCuts;
   vector<vector<FSParticle*> >                    m_particleCombinations;
   map<string, vector< vector <unsigned int> > > m_submodeIndices;
```

# How to use it?
# First – Marlin and LCIO

# Marlin-LCIO

Modular Analysis & Reconstruction for the LINear Collider

- Simple logic in processors
  - Each processor has similar interface: data collection in LCIO
  - Each processor dose simple job: Isolated lepton/photon finding, jet-clustering, flavor tagging, etc. …
  - Use processors to realize what you want

# Example --- fast simulation

- Simple smearing used for fast simulation
  - Tracks
  - Photons
  - Neutral hadrons
- According to the detector performance from full simulation, easy to change the parameters
- But very, very fast, easy to get some fast conclusion
- Start from generator data …

  - Not vertex — so no flavor tagging
  - Not photon conversion

# Example---fast simulation

```xml
<execute>
   <processor name="RootFileProcessor"/>
   <!-- read generator sample-->
   <processor name="MyStdHepReader"/>
   <!-- fast smearing -->
   <processor name="MyLGFastMCProcessor"/>
   <!-- fast jet and simple lepton isolation -->
   <processor name="MyLGFastJetClustering"/>
   <!-- analysis -->
   <processor name="FSClasserProcessor"/>
   <!-- save fast simulation to disk for further usage -->
   <processor name="MyLCIOOutputProcessor"/>
</execute>
```

# Example

# e+e- → ZH→$\mu^+\mu^-$ jet jet

- Select muon pair: energetic, isolated, Z-mass, …
- The rest of partices: vertex finder+jet-clustering
- Combine muon pair and jet pair into what you want
- Most processor are ready, just get firmilar with and use them

```
<execute>
  <processor name="RootFileProcessor"/>
  <processor name="MyISOlatedLeptonFinderProcessor"/>
  <processor name="VertexFinder"/>
  <processor name="JetClusteringAndFlavorTag"/>
  <processor name="FSClasserProcessor"/>
</execute>
```

# First processor

```
<processor name="MyISOlatedLeptonFinderProcessor" type="ISOlatedLeptonFinderProcessor">
    <parameter name="InputCollection" type="string" lcioInType="ReconstructedParticle">ArborPFOs </parameter>
    <parameter name="OutputCollectionIsolatedLeptons" type="string" lcioOutType="ReconstructedParticle">IsoLeps </parameter>
    <parameter name="OutputCollectionWithoutIsolatedLepton" type="string" lcioOutType="ReconstructedParticle">RemainPFOs </parameter>
    <parameter name="OutputCollectionIsolatedZLeptons" type="string" lcioOutType="ReconstructedParticle">IsoZLeps </parameter>
    <parameter name="OutputCollectionWithoutIsolatedZLepton" type="string" lcioOutType="ReconstructedParticle">RemainZPFOs </parameter>

    <parameter name="UsePID" type="bool">true </parameter>
    <parameter name="ElectronMaxEcalToHcalFraction" type="float">1.5 </parameter>
    <parameter name="MuonMinEnergyDepositByMomentum" type="float">0 </parameter>
         ... ...

    <parameter name="UseImpactParameter" type="bool"> false </parameter>
    <parameter name="ImpactParameterMin3D" type="float">0 </parameter>
    <parameter name="ImpactParameterMax3D" type="float">0.01 </parameter>
         ... ...

    <parameter name="UseImpactParameterSignificance" type="bool">false  </parameter>
    <parameter name="ImpactParameterMin3DSignificance" type="float">0 </parameter>
    <parameter name="ImpactParameterMaxZ0Significance" type="float">1e+20 </parameter>
         ... ...

    <parameter name="UseRectangularIsolation" type="bool">true   </parameter>
    <parameter name="CosConeAngle" type="float">0.98 </parameter>
    <parameter name="IsolationMaximumConeEnergy" type="float">1e+20 </parameter>
         ... ...

    <parameter name="UsePolynomialIsolation" type="bool"> false  </parameter>
    <parameter name="IsolationPolynomialCutA" type="float"> 0.0  </parameter>
    <parameter name="IsolationPolynomialCutB" type="float"> 4.0 </parameter>
         ... ...

    <parameter name="UseJetIsolation" type="bool">false </parameter>
    <parameter name="JetCollection" type="string" lcioInType="ReconstructedParticle">JetsForIsolation </parameter>
    <parameter name="JetIsolationVetoMaximumXt" type="float">0.25 </parameter>
         ... ...
</processor>
```

# Second, third …
# Use the output of first (second)

```
<processor name="VertexFinder" type="LcfiplusProcessor">

  <!-- run primary and secondary vertex finders -->
  <parameter name="Algorithms" type="stringVec"> PrimaryVertexFinder BuildUpVertex </parameter>
  <parameter name="ReadSubdetectorEnergies" type="int" value="1"/> <!-- true for ILD -->
  <parameter name="UpdateVertexRPDaughters" type="int" value="0"/> <!-- false for non-updative PandoraPFOs -->
  <parameter name="TrackHitOrdering" type="int" value="1"/> <!-- Track hit ordering: 0=ILD-LOI,SID-DBD, 1=ILD-DBD -->
  <parameter name="PrintEventNumber" type="int" value="0"/> <!-- 0 for not printing event number, n for printing every n events -->

  <!-- specify input collection names -->
  <parameter name="PFOCollection" type="string" value="RemainZPFOs" />
  <parameter name="PrimaryVertexCollectionName" type="string" value="PrimaryVertex" />
  <parameter name="BuildUpVertexCollectionName" type="string" value="BuildUpVertex" />
  <parameter name="BuildUpVertex.V0VertexCollectionName" type="string" value="BuildUpVertex_V0" />
```

```
<processor name="JetClusteringAndFlavorTag" type="LcfiplusProcessor">

  <!-- run primary and secondary vertex finders -->
  <parameter name="Algorithms" type="stringVec"> JetClustering JetVertexRefiner FlavorTag ReadMVA</parameter>

  <!-- general parameters -->
  <parameter name="PFOCollection" type="string" value="RemainZPFOs" /> <!-- input PFO collection -->
  <parameter name="UseMCP" type="int" value="0" /> <!-- MC info not used -->
  <parameter name="MCPCollection" type="string" value="" /> <!-- not used -->
  <parameter name="MCPFORelation" type="string" value="" /> <!-- not used -->
  <parameter name="ReadSubdetectorEnergies" type="int" value="1"/> <!-- true for ILD -->
  <parameter name="UpdateVertexRPDaughters" type="int" value="0"/> <!-- false for non-updative PandoraPFOs -->
  <parameter name="TrackHitOrdering" type="int" value="1"/> <!-- Track hit ordering: 0=ILD-LOI,SID-DBD, 1=ILD-DBD -->

  <!-- jet clustering parameters -->
  <parameter name="JetClustering.InputVertexCollectionName" type="string" value="BuildUpVertex" /> <!-- vertex collections to be used in JC -->
  <parameter name="JetClustering.OutputJetCollectionName" type="stringVec" value="VertexJets" /> <!-- output collection name, may be multiple -->
  <parameter name="JetClustering.NJetsRequested" type="intVec" value="2" /> <!-- Multiple NJets can be specified -->
  <parameter name="JetVertexRefiner.OutputVertexCollectionName" type="string" value="RefinedVertex" />
```

# Power of Processor group

- 1+1+1>>3
- Small blocks build great building …

- Do not invent wheel
- Contribute blocks to APODIS

- Move to FSClasser …
  - Final state classification

# Two types of analyses

- Inclusive: u+u- recoil, only select the muon pair, not requirement on 4-P conservation
- Exclusive: all final state particle need to be detected, usually requires 4-P conservation — kinematic fit

- Physics objects ready : list of e+/e-/mu+/mu-/tau+/tau-/photon/jet …
- Combine them into event--- exclusively or inclusively
  - Multi-entry problem
  - Save matched MC truth
  - Kinematic fit
  - Save tree: all necessary variables
  - Simultaneous event selection

# Example

# ee→ $\mu^+\mu^-$ Higgs(anything)

- Tell the FSClasser processor what you want

$\tau+$ $\mu+$ e+

- INC 0 _ 0 0 0 1 1 0 0

jet $\gamma$ $\tau-$ $\mu-$ e-

8 digits: the numbers of particles your want

# Cut on μμ masses

- Invariant mass: Muu>50&&Muu<120
  - INC0_0001100 0_1100 RawMass 50 120
- Invariant mass: Mrec>110&&Mrec<160
  - INC0_0001100 0_1100 RecoilMass 110 160
- Reduce root file size to save space and CPU time

# Multi-entry

- If there are more than one mu+ and/or mu-
- There would be more than entries per event
- All the combination will be saved to ntuple to avoid bias, which can be further carefully analyzed in root

# Example
# ee→ $\mu^+\mu^-$ Higgs→ $\mu^+\mu^-$ jet jet

- Tell the FSClasser processor what you want
- EXC 2 _ 0 0 0 1 1 0 0

jet γ τ+ τ− μ+ μ− e+ e-

8 digits: the numbers of particles your want

# Simultaneous event selection
# number of channels < 300

```
<!-- -->
<parameter name="FS100"    type="string"    >  INC0_0001100 </parameter>
<parameter name="FS102"    type="string"    >  INC0_0000011 </parameter>
<parameter name="FS111"    type="string"    >  EXC2_0001100 </parameter>
<parameter name="FS112"    type="string"    >  EXC2_0000011 </parameter>
<parameter name="FS121"    type="string"    >  EXC0_2001100 </parameter>
<parameter name="FS122"    type="string"    >  EXC0_2000011 </parameter>
<parameter name="FS132"    type="string"    >  EXC0_0001111 </parameter>
<parameter name="FS142"    type="string"    >  EXC0_0002200 </parameter>
<!--
```

Each channel has its own tree

```
KEY: TTree    ntINC0_0001100;1
KEY: TTree    ntEXC2_0001100;1
KEY: TTree    ntGEN2_0001100;1
KEY: TTree    ntEXC0_2001100;1
KEY: TTree    ntGEN0_2001100;1
KEY: TTree    ntEXC0_0001100;1
KEY: TTree    ntGEN0_0001100;1
```

# Satisfys most analyses

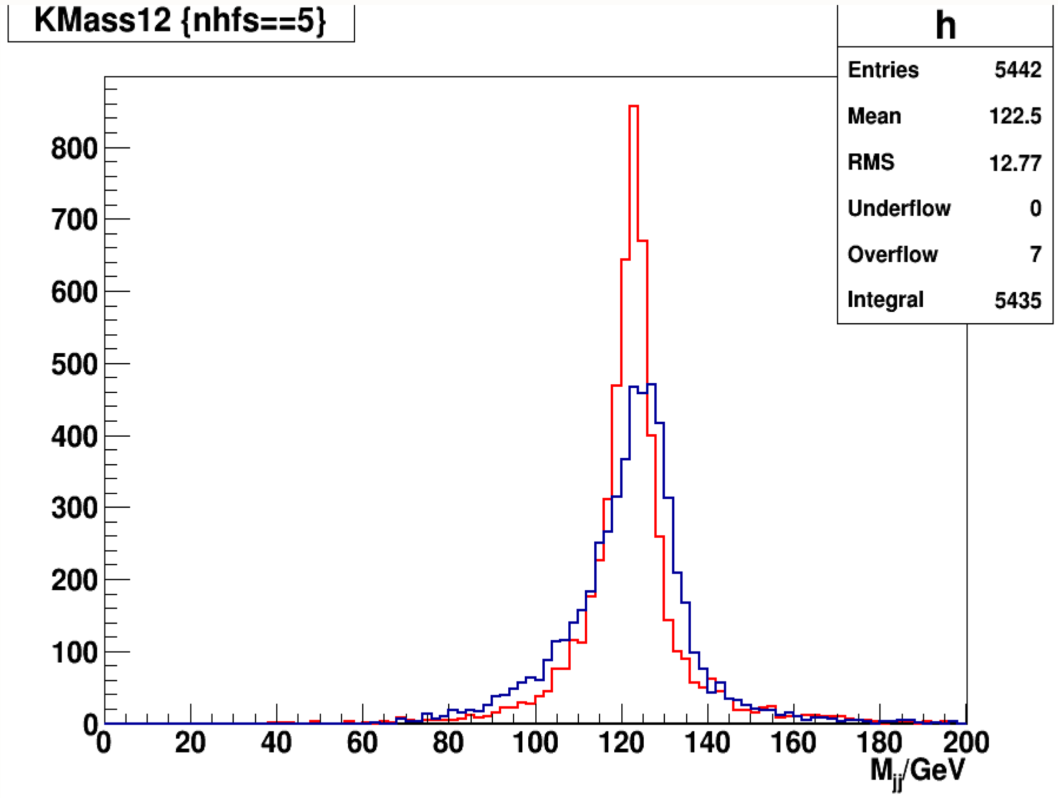
- Variables in a tree
  - Event level:

    Run Event Weight ntrks nclus nPFOs Emax njets ntaus nElec nMuon nIsoLep nGamma VisEn VisPx VisPy VisPz VisMass $y_{ij}$ Sphericity Aplanarity Thrust Major Minor ThrustTheta ThrustPhi ThrustEDM MajorTheta nhfs VisEnMC MisEnMC MissingMass2 Chi2 prob
  - Particle level

    PfontrkP1 PfoncluP1 PfoVtxRP1 PfoVtxZP1 PfoecalP1 PfohcalP1 PfonHitsP1 PfototCalEnP1 PfochargeP1 PfomassP1 PfoEnP1 PfoPxP1 PfoPyP1 PfoPzP1 PfoPtP1 PfoPtotP1 PfoLepTypeP1 PfoRapidityP1 PfocosThetaP1 PfoPDGIDP1 PfoMCTENP1 PfoMCTPXP1 PfoMCTPYP1 PfoMCTPZP1
  - Masses, angles
    - Invariant masses of all 2 or 3 particle combinations
    - Recoil masses of all 1, 2, 3 particle combinations
    - The angles among any two particles

# Kinematic fit

- Four constraints kinematic fit applied by default for exclusive final states to improve the resolution, useful for some analyses with jets
- All final states correlated, need to estimate its systematic uncertainty
- Chi^2 requirement improves more



# Other utilities

- Automatic validation plots: mulitiplicity, visible energy&mass, etc. …
  - switch on in xml file
- Event shape: thrust, spherity, Fractal dimension, …
- MC-truth matching:
  - In a jet, the parton mother of each track is traced, and the mother of majority is taken, which avoids mis-matching of some other methods, like angle method.

# Summary and discussion

- An general analysis frame work of full and fast simulation
- It is easy to use and you can get Ntuple without coding any more
- To be improved
  - Reduce the Ntuples
  - Limit number of stdhep files: multi files
  - More on MC truth: essential complication in hadronization
  - … …
  - Something else according to user requests