Current Status of LCFIPlus

Masakazu Kurata, Tomohiko Tanabe
The University of Tokyo
Taikan Suehara
Kyushu University
Jan Strube
PNNL

LCWS16, 12/05/2016-12/09/2016

INTRODUCTION

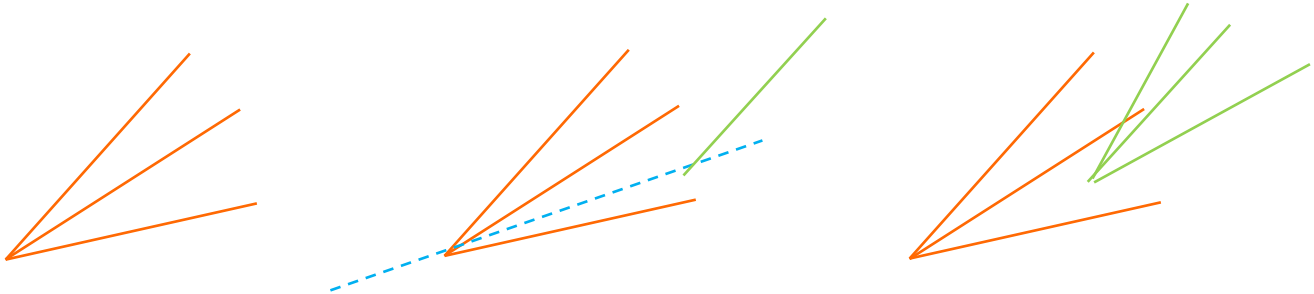
- Recent issues and treatments
  - Problems in ROOT6 environment
  - Main problem: Flavor tagging efficiency degrades significantly

- For flavor tagging improvement
  - Updated PID and vertex mass recovery
  - Vertex finding and related issue
  - Flavor tagging efficiency

Recent issues and treatmenets

Flavor classification in LCFIPlus

- Vertex categorization
  - Vtx: no, 1vtx, 1vtx+1single track, and 2vtx



- Jet flavor determination
  - Using multivariate analysis with TMVA
  - Classifier: Gradient BDT with multiclass
  - b-likeness, c-likeness, uds-likeness (3-class)
  - Sum of three likenesses should be 1
- Combination of 4-caterogies
  - Assuming the average x-likeness of each category equals the fraction of events
- Flavor tagging efficiency degrades drastically in ROOT6

A TMVA modification

MethodBDT.cxx

```
if (!DoRegression()){
    Log() << kINFO << "<InitEventSample> For classification trees, "<< Endl;
    Log() << kINFO << " the effective number of backgrounds is scaled to match "<<Endl;
    Log() << kINFO << " the signal. Othersise the first boosting step would do 'just that'!"<<Endl;
    // it does not make sense in decision trees to start with unequal number of signal/background
    // events (weights) .. hence normalize them now (happens atherwise in first 'boosting step'
    // anyway..
(nip)
    if (sumSigW && sumBkgW){
      Double_t normSig = nevents/((1+fSigToBkgFraction)*sumSigW)*fSigToBkgFraction;
      Double_t normBkg = nevents/((1+fSigToBkgFraction)*sumBkgW); ;
      Log() << kINFO << "re-normlise events such that Sig and Bkg have respective sum of weights = "
          << fSigToBkgFraction << Endl;
      Log() << kINFO << "  sig->sig*"<<normSig << "ev. bkg->bkg*"<<normBkg << "ev." <<Endl;
      Log() << kINFO << "#events: (reweighted) sig: "<< sumSigW*normSig << " bkg: " << sumBkgW*normBkg << E
      Log() << kINFO << "#events: (unweighted) sig: "<< sumSig << " bkg: " << sumBkg << Endl;
      for (Long64_t ievt=0; ievt<nevents; ievt++) {
        if ((DataInfo().IsSignal(fEventSample[ievt])) ) fEventSample[ievt]->SetBoostWeight(normSig);
        else                                            fEventSample[ievt]->SetBoostWeight(normBkg);
      }
(nip)
```

Renormalize event weights here

This violates our expectation of average x-likeness!

Treatment to it

- We found that if we comment out this feature the performance is recovered.
- The comment says this normalization do not have side effects, which is not true.
- We submitted a patch to have an option called "SkipNormalization" to BDTG to skip the feature
- The patch was accepted to the official ROOT repository (16 Sep. 2016)
- The option is automatically added inside latest LCFIPlus (still not in the official release)
- With older version you can explicitly specify "SkipNormalization" in "TrainMVA.BookOptions" in train.xml (new ROOT necessary, >6.06)

Another issue: crash in ROOT6

- LCFIPlus is suddenly crashed at somewhere in event creation
- Might be some different behavior between ROOT 5 and 6
- So far, this crash does not affect flavor tagging itself
- Under investigation…

For improvement

LCFIPlus improvement

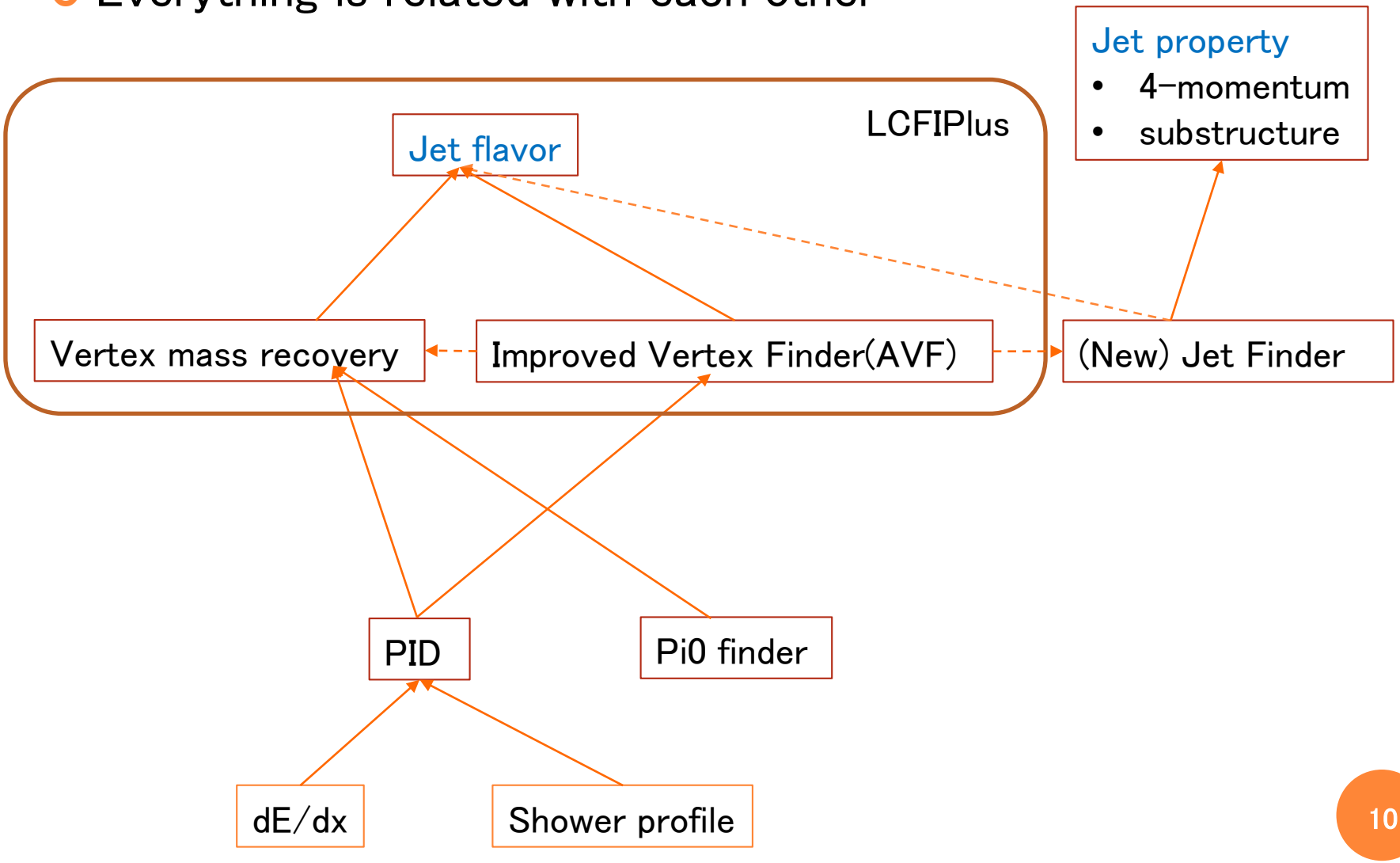
- For better flavor tagger, we need
  - Find secondary/tertiary vertices as many as possible
  - Perfect reconstruction of B/D meson mass
    - Perfect attachment of charged particles
    - Recover lost component(especially neutrals)
  - Better variables which can separate jet flavors

- We found that we can acquire flavor tagging improvement:
  - Vertex finding efficiency improvement → introduce a new algorithm
  - Vertex Mass Recovery → using escaping π^0s
  - Better flavor separation for jets of 0 vtx

- Much help is necessary!:
  - Particle ID is one of the key to flavor tagging improvement
    - To classify vertices
  - Pi0 reconstruction(γ pairing) is other key for vertex mass recovery
    - First of all, pi0 is necessary!

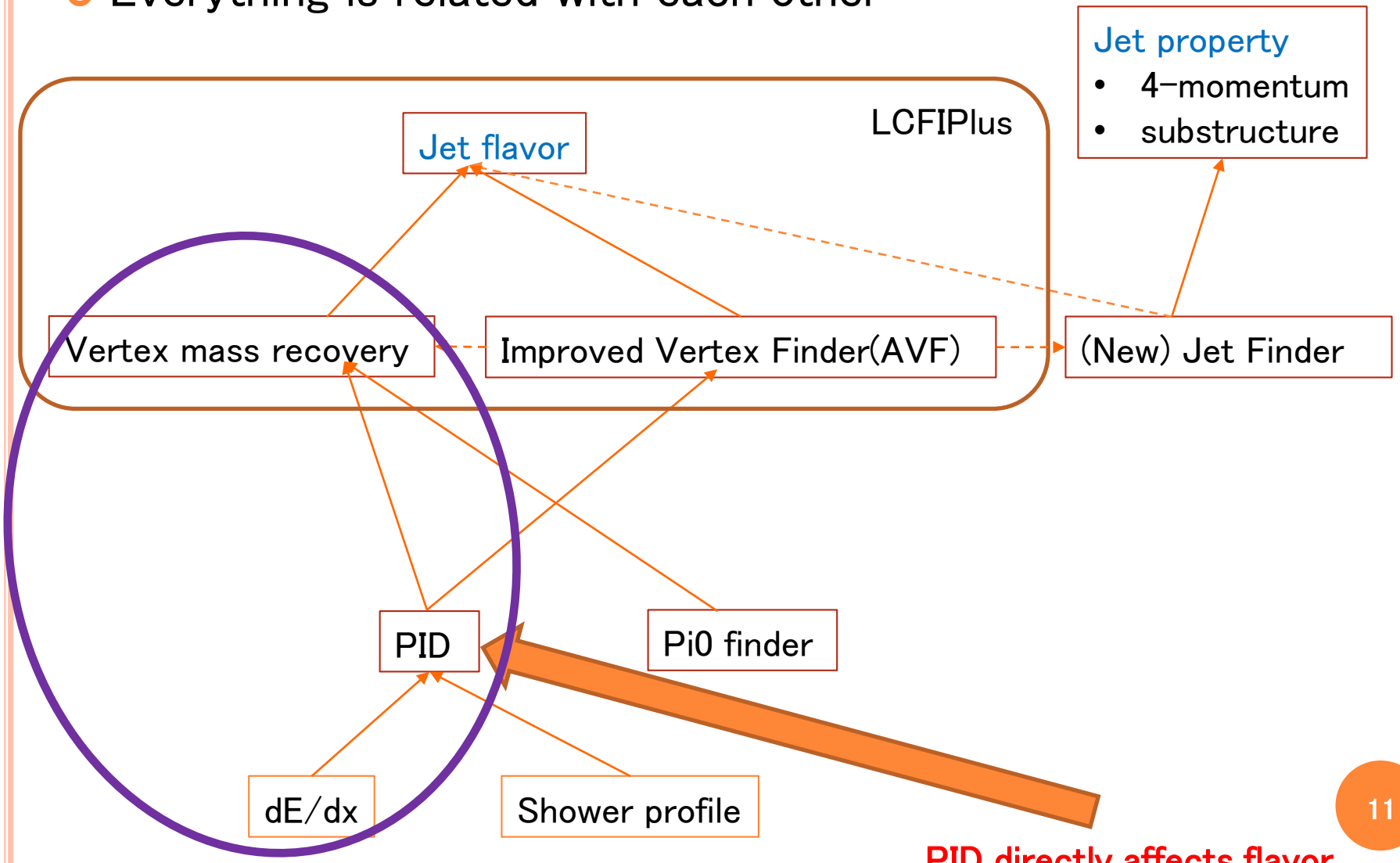
RELATIONSHIPS

- Everything is related with each other



Relationships

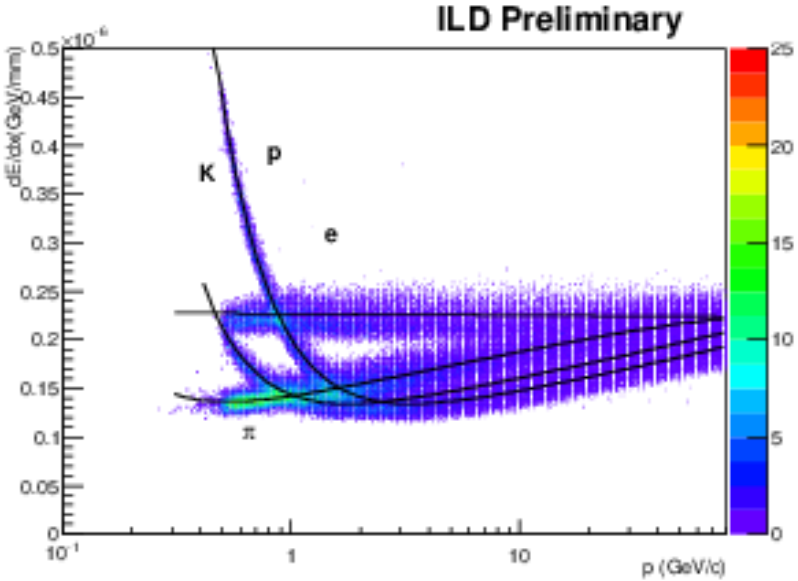
- Everything is related with each other

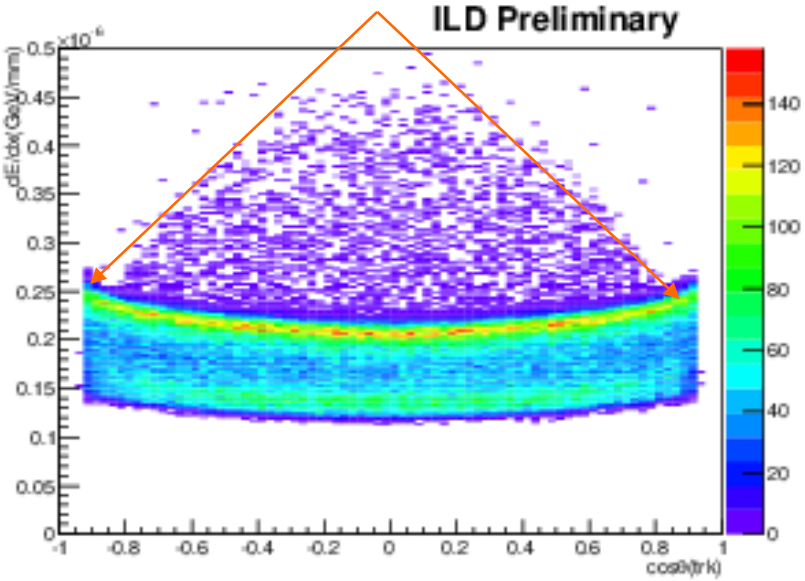


PID directly affects flavor tagging improvement

Angular dependence of dE/dx

- There is angular dependence of dE/dx value(before correction)





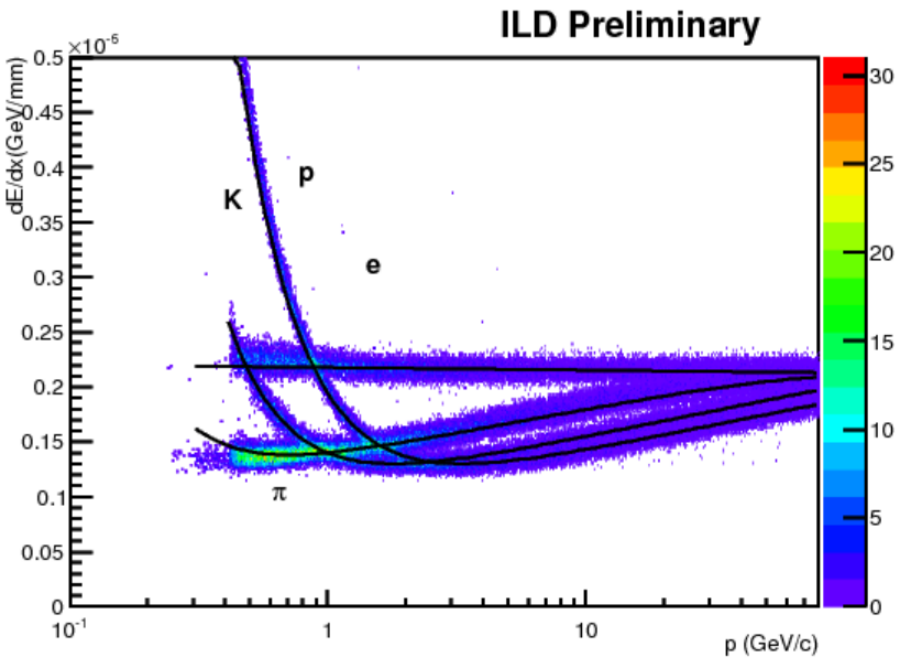
- Include angular correction
  - Tried some functions for fitting
  - Best parameterization:

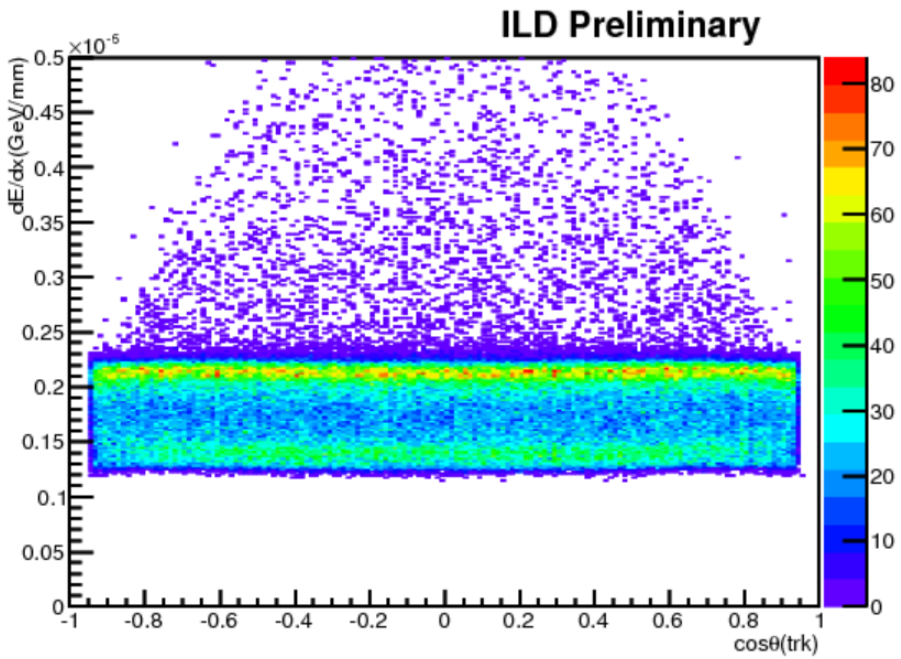
$$\frac{dE}{dx}_{corr} = \frac{dE}{dx} \cdot \min(\theta,\ \pi - \theta)^{0.07}$$

  - This parameterization is best especially very forward region

After correction

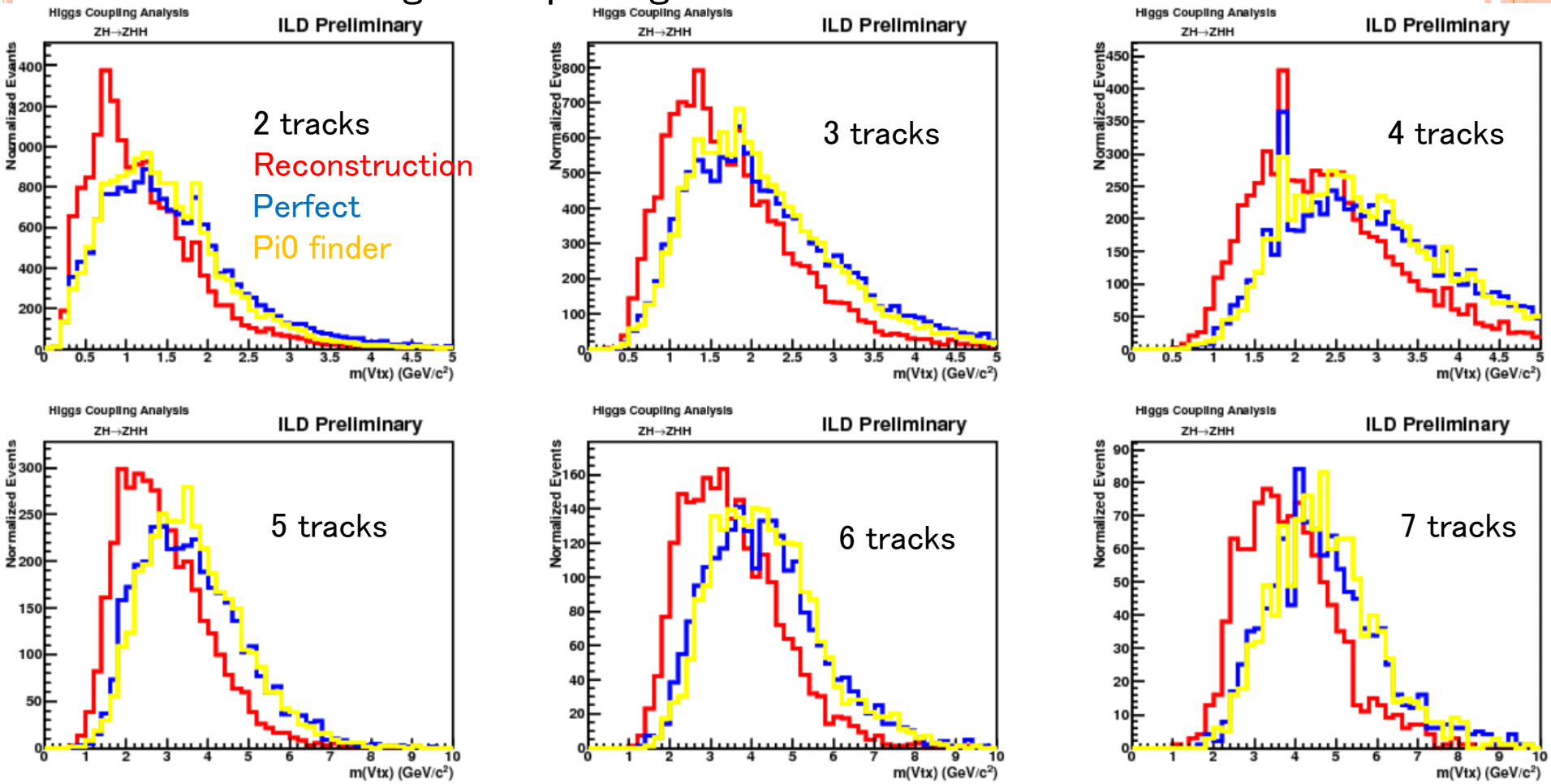
- Very good improvement of dE/dx distribution
  - Forward region very nice!





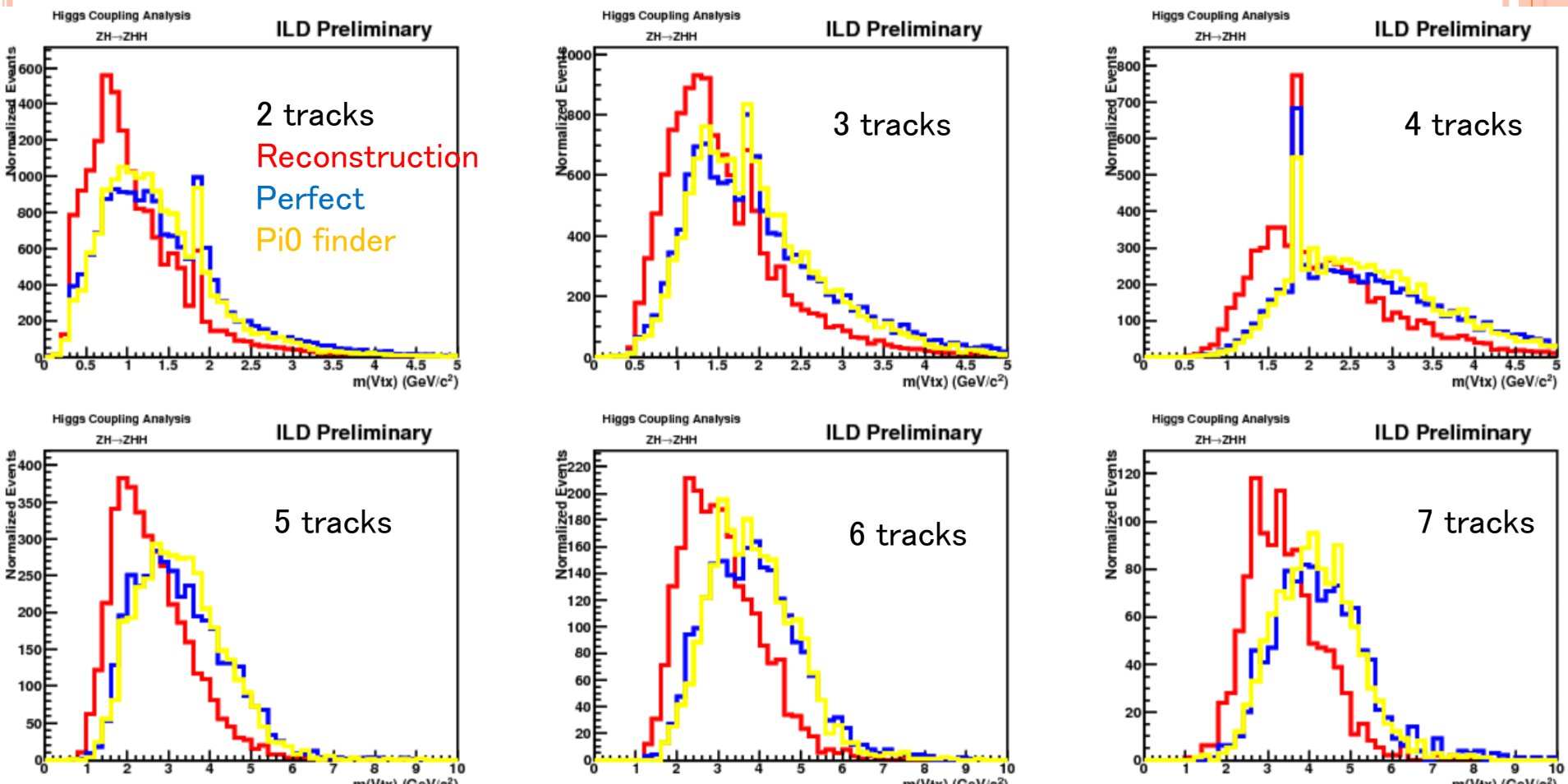
Vtx masses of bjets in double-Higgs process

- Vtx mass distributions for each vertex pattern(ntrk)
  - bjets with 1vtx
  - Difference is limited by mis-pairing of gammas(eff. ～50%) and mis-attachment of pi0s
  - Need better gamma pairing!



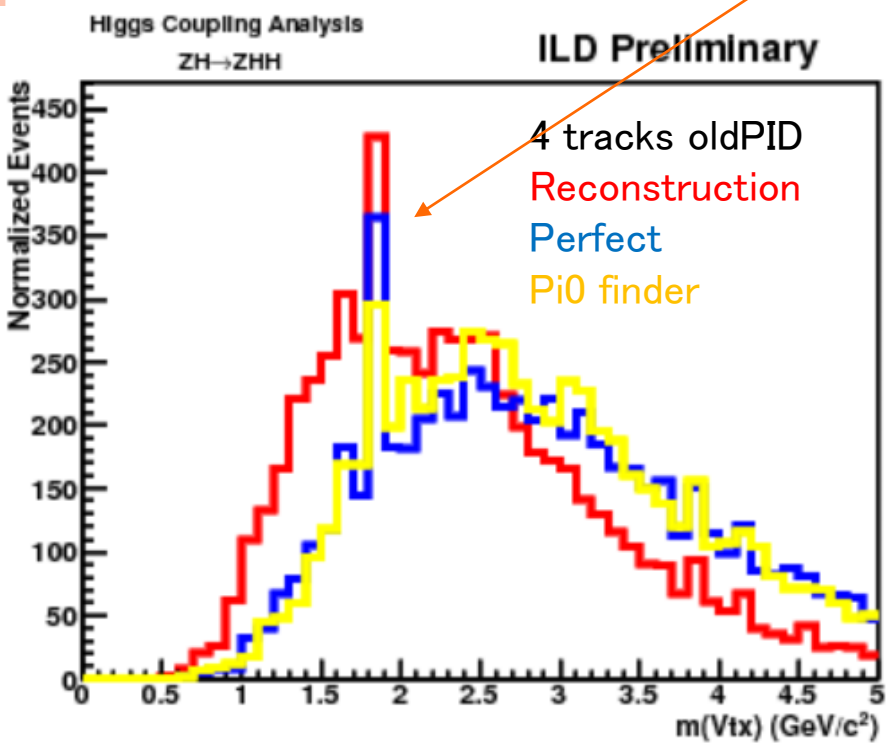
Vtx masses of bjets in double-Higgs process

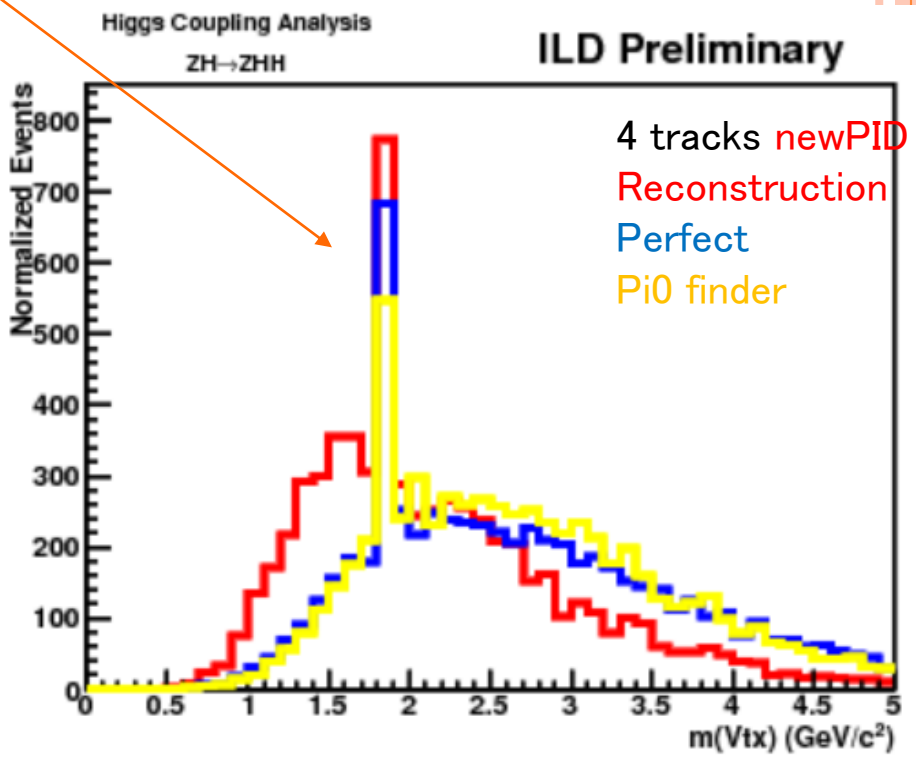
- Vtx mass distributions for each vertex pattern(ntrk)
  - Updated PID + ROOT6 environment
  - Vertex mass recovery is reasonable at same degree



COMPARISON

- Better reconstruction of D meson!





Adaptive Vertex Fitting

- To introduce the effect of multi-vertex fitting
  - Introduce weight function to estimate vertex which a track belongs to
  - Weight function definition: k-th track's weight on n-th vertex

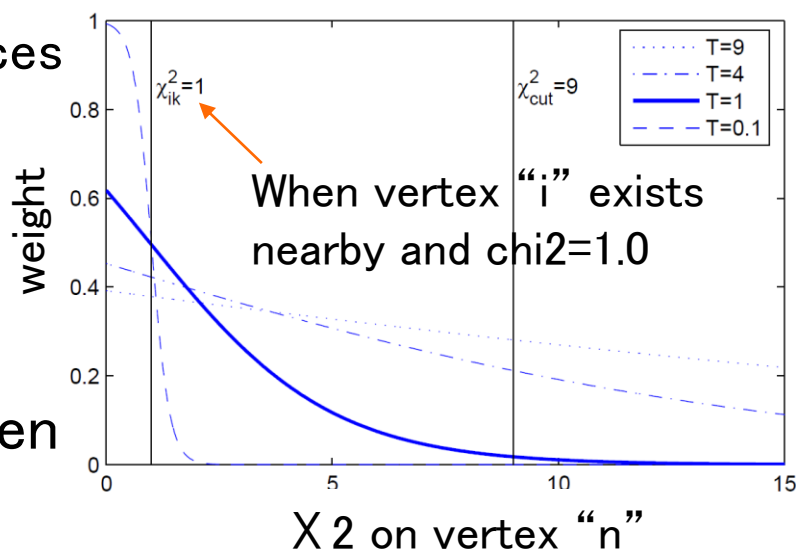
$$w_{nk} = \frac{e^{-\chi^2_{nk}/2T}}{e^{-\chi^2_{\mathrm{cut}}/2T} + \sum_{i=1}^{N} e^{-\chi^2_{ik}/2T}}$$

  - Parameter: temperature T
    - If T very small, decision is like χ 2 minimization(almost same as DBD LCFIPlus)
    - If T large, multi-vertex effect becomes large
  - In multi-vertex environment, weight on certain vertex will degrade

→becomes harder to attach tracks to vertices in multi-vertex environment

→can reject fake tracks well!

Weight of track "k" on vertex "n"



- Thanks to weight function, we can loosen the track quality selection

→vertex finding eff. will be improved!

Impact of Adaptive Vertex Fitting

- Common parameters are set at same values for comparison
- Same event sample(qqHH sample@500GeV) 19889 events
- 6 jet clustering, jet matching with MCtruth is performed
- Num. of jets with vertex:

method	bjet with 2vtx	bjet with 1+1vtx	bjet with 1vtx	total
DBD LCFIPlus	10581	9104	12847	32532
AVF	13190	6576	13233	32999

- Total jets with vtx: ～1.4% increased
  - Jets with 2vtx: ~22% increased → good for bjet ID!
  - Jets with 1vtx: ~3% increased → good for uds jet separartion!
- Fake track rate per vtx: how many fake tracks contaminate on vertices?
  - Almost same – slightly better!

method	bjet with 2vtx	bjet with 1+1vtx	bjet with 1vtx
DBD LCFIPlus	0.029±0.001	0.013±0.0012	0.055±0.002
AVF	0.025±0.001	0.012±0.0013	0.055±0.002

Related issue: vertex charge assignment

- For better vertex charge assignment, it is necessary to find as many vertices as possible
- So far, we are trying to suppress fake tracks on vertices
- For vertex charge study, trying to create as many vertices as possible
  - Loosen vertex quality condition inside the VertexFinder
- For more detail, see S. Bilokin's talks at several workshops& meetings

VERTEX FINDING EFF.

- Num. of vertices(using ～20000 of qqHH sample)

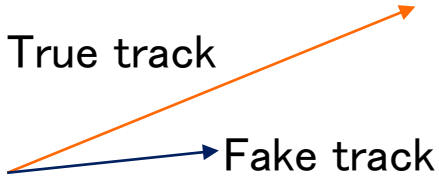
method	bjet with 2vtx	bjet with 1+1vtx	bjet with 1vtx	total
DBD LCFIPlus	10581	9104	12847	32532
AVF	13190	6576	13233	32999
Trial 1	13154	6346	13873	33373

  - AVF: Num. of vertices increase 1.4%
  - Trial 1: Try to obtain as many vertices as possible
    - Bjet with 1vtx wil be increased

- Fake rate per vertices:

method	bjet with 2vtx	bjet with 1+1vtx	bjet with 1vtx
DBD LCFIPlus	0.029±0.001	0.013±0.0012	0.055±0.002
AVF	0.025±0.001	0.012±0.0013	0.055±0.002
Trial 1	0.030±0.001	0.013±0.0013	**0.078±0.002**

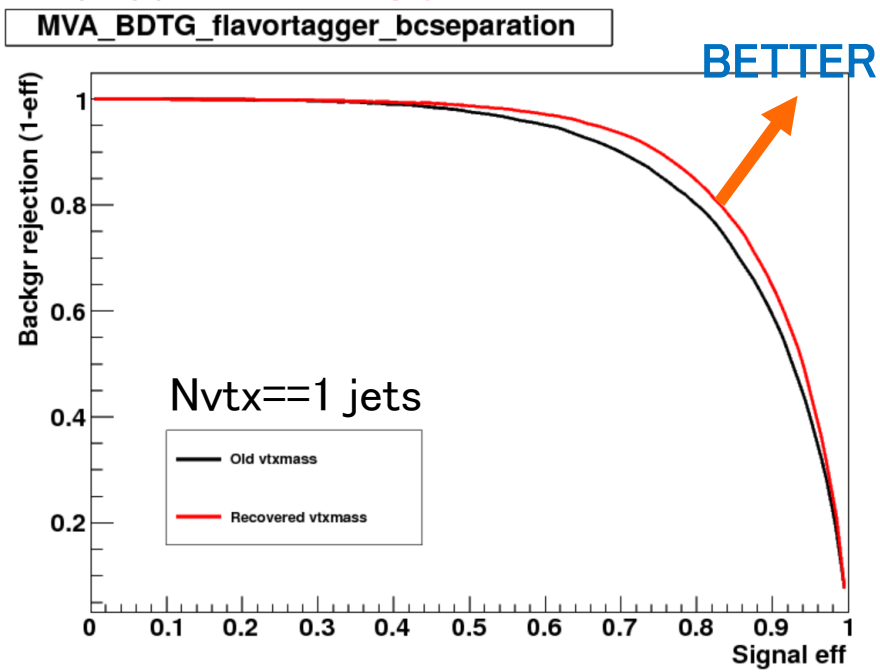
  - Fake rate of 1vtx case will be increased drastically
  - Vtx with 2tracks will be increased

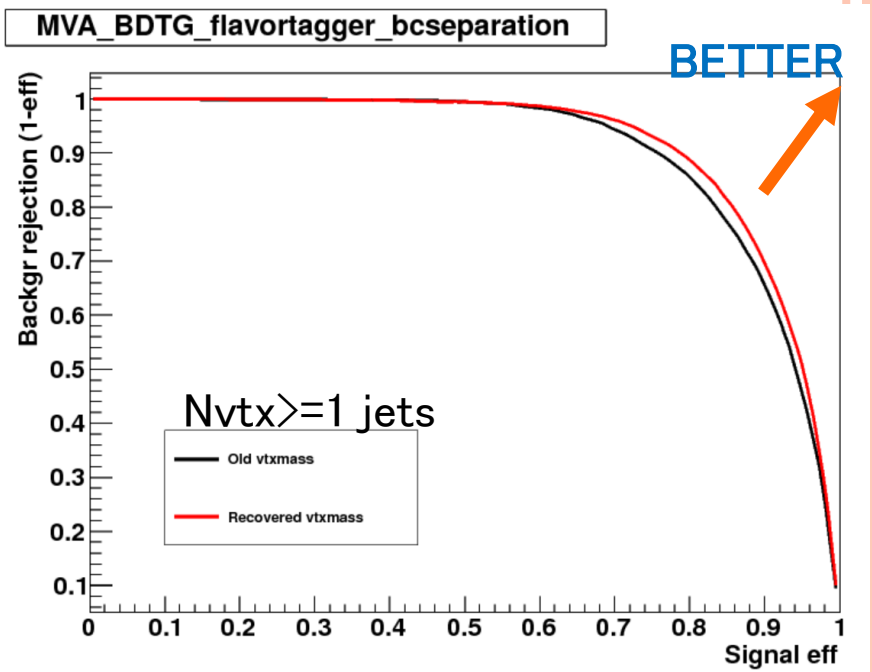


Such kind of vertices is increased??

Vertex mass recovery effect on flavor tagging

- Construct a "toy" flavor tagger
  - Input variables are obtained from LCFIPlus
  - Input variable selection is too primitive!
  - Only vertex mass is replaced to recovered vertex mass
  - Compare with ROC curve
- Vertex: DBD ver.





- Using binary classifier…
- Result is very primitive: How about more realistic case?

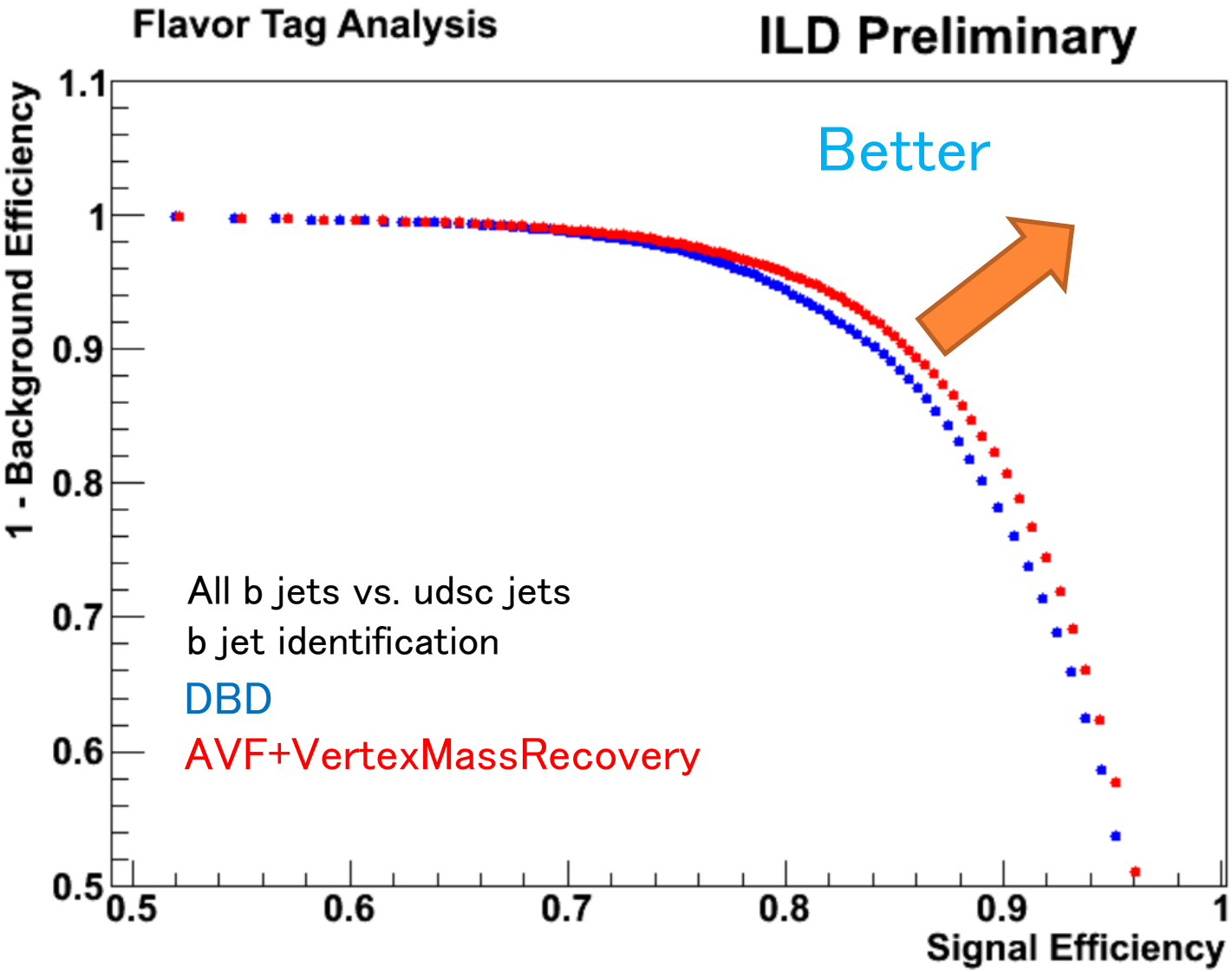
Impact on Flavor tagging efficiency

- Need to check final flavor tagging efficiency!
- Using REALISTIC environment of LCFIPlus
  - Classifier: Gradient BDT with multiclass
  - b-likeness, c-likeness, uds-likeness (3-class)
  - 4 categorization with vertices in jets

- Train MVA & check b-likeliness
  - Samples: 6 fermion samples coming from ZZZ events@500GeV
  - Use AVF & VertexMassRecovery
  - Input variables are add or replaced to their corresponding variables coming from AVF & VertexMassRecovery
  - Check b-likeliness of each flavor jet
  - Compare ROC curve with DBD ver. LCFIPlus

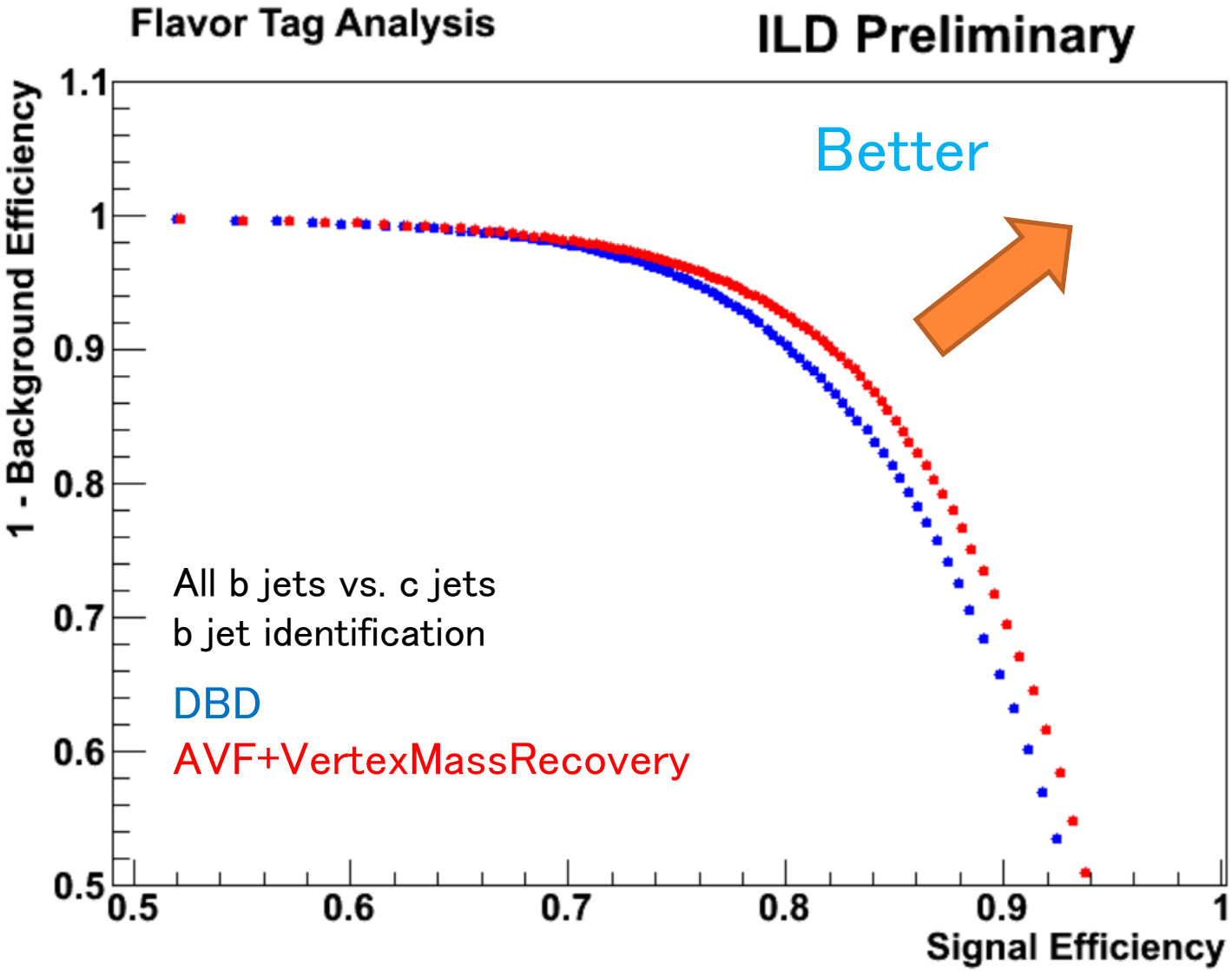
Impact on Flavor Tagging Efficiency

- 6f samples coming from ZZZ events@500GeV
- Compare with ROC curve



Impact on Flavor Tagging Efficiency

- 6f samples coming from ZZZ events@500GeV
- Compare with ROC curve



Summary and prospects

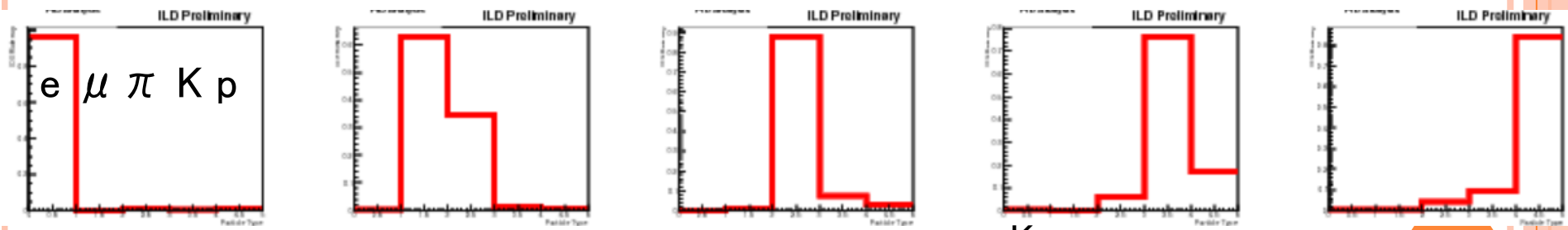
- We are going to adjust LCFIPlus to ROOT6 environment
  - We found the cause of degradation of flavor tagging efficiency in ROOT6 and treat it
  - We still have a problem of sudden crash in ROOT6
    - Under investigation
- For flavor tagging improvement:
  - Updated PID provides better reconstruction of (charged)vertex mass
  - Vertex mass recovery is still reasonable
  - Vertex finding is clearly trade-off with fake vertex creation…
- Estimate flavor tagging efficiency using multiclass classifier
  - Efficiency improvement can be seen using AVF+ VertexMassRecovery!
  - Need idea for 0vtx jet case
- We need to tackle jet clustering improvement
  - It is crucial for multi-jet event analyses!

Back ups

PID

- Use dE/dx, Shower shapes in Calorimeters, E/P, etc.
- Construct Bayesian classifier
  - Estimate posterior probability of each particle hypothesis
- μ / π separation for low momentum tracks included
  - Developed For degenerated Higgsino study
  - Use special shower shape information of μ / π at forward calorimeter
  - Using TMVA for separation
  - Effective <2GeV/c tracks

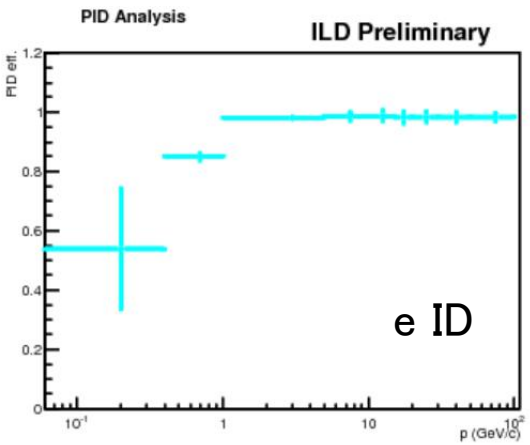
- Simple example: Overall ID & mis-ID efficiency:

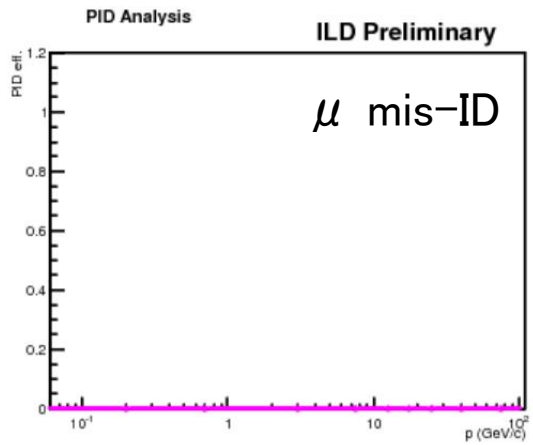


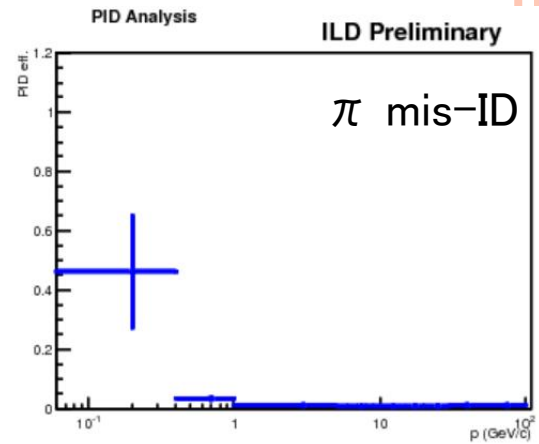
e μ π K p

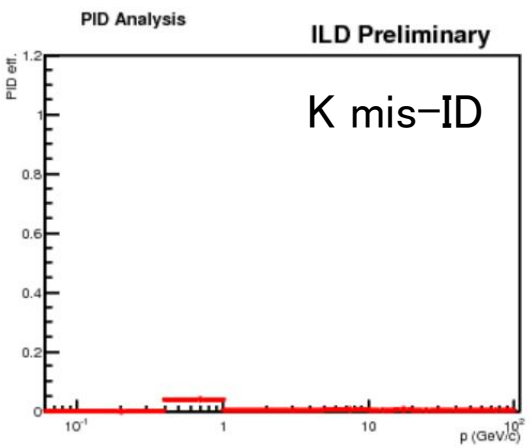
PERFORMANCE EXAMPLE

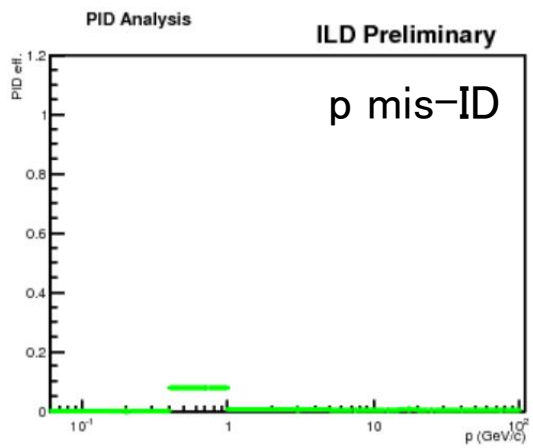
- Single track electron
- Momentum dependence of ID & mis-ID efficiency





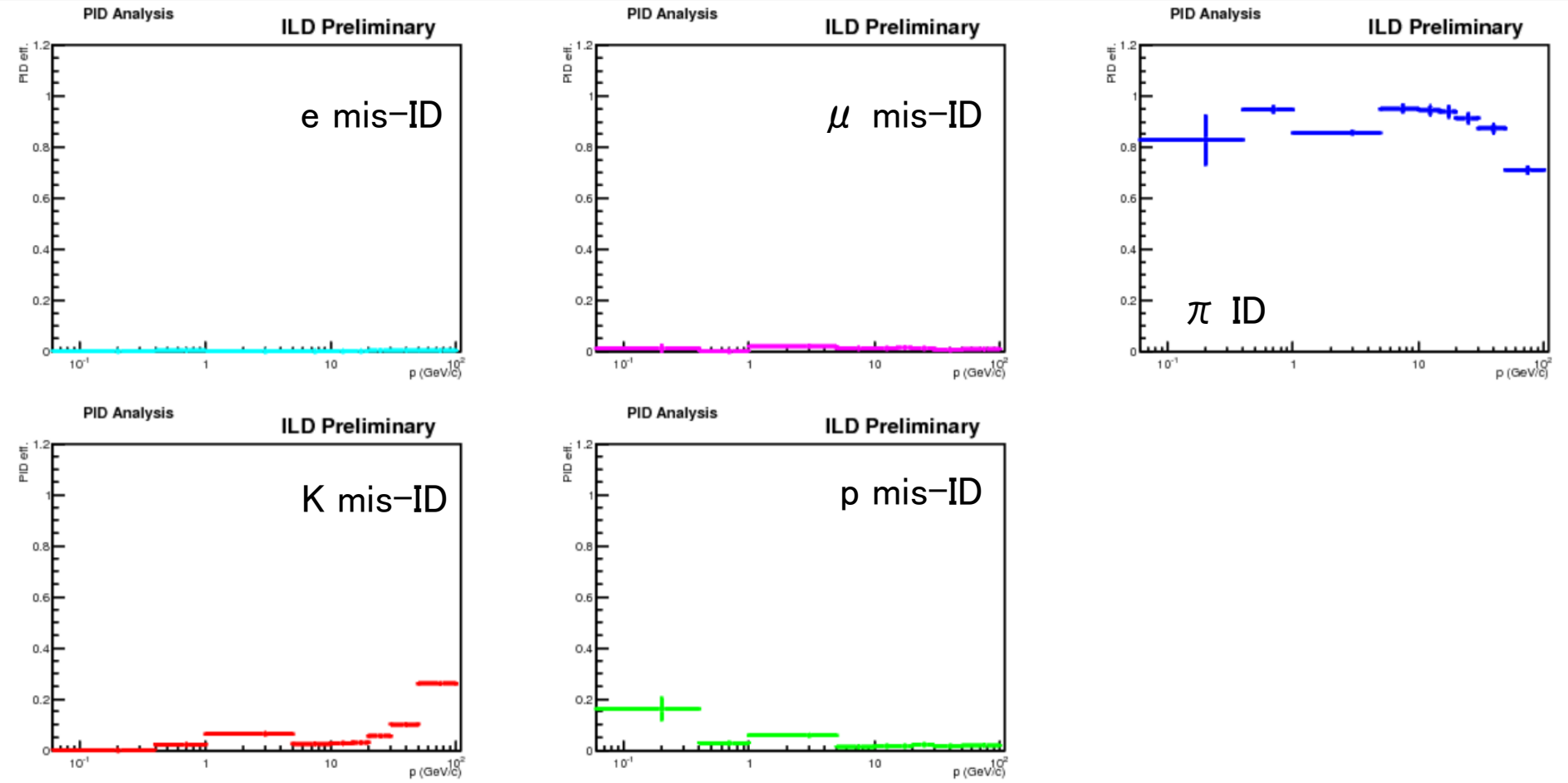






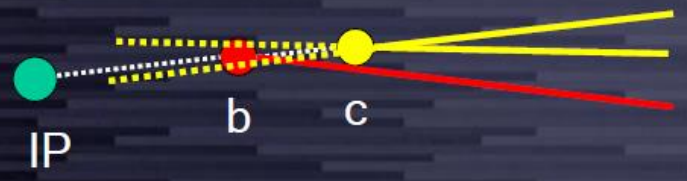
Performance example

- Single track pion
- Momentum dependence of ID & mis-ID efficiency



Issues / Vertex finder

- Apply recent developments
- Not robust against background
  - $\gamma\gamma \rightarrow$ hadrons (CLIC performance very bad)
  - pairs



- b/c separation
  - More efficient finder $\leftarrow\rightarrow$ worse b/c separation
  - Should be treated with different vertex finder
- Association of low-energy tracks

Issues / Vertex finder (cont.)

- Refitting tracks
  - may improve the vertex separation
  - Need tracker hits (not available in DSTs)
  - Kalman filter or ...
- Speed of vertex finder
  - Vertex fitter is slow
    - 2 loops of Minuit minimization
  - Vertex finder is also slow
    - trying every pair of tracks

Issues / Jets, Leptons

- Jet clustering
  - Interface to external? eg. Fastjet?
  - Color-singlet? kinematic constraint? More?
- Lepton finder
  - Apply LikelihoodPID instead of simple one
    - Function is already available,
      need training and checking performance
  - Tau finder

Issues / flavor tagging

- Treatment of multiple jets inside (Jet substructure ID etc.)
- Treatment of vertices
  - "Concrete" or "Doubtful" vertices
    - based on probability, # of tracks etc.

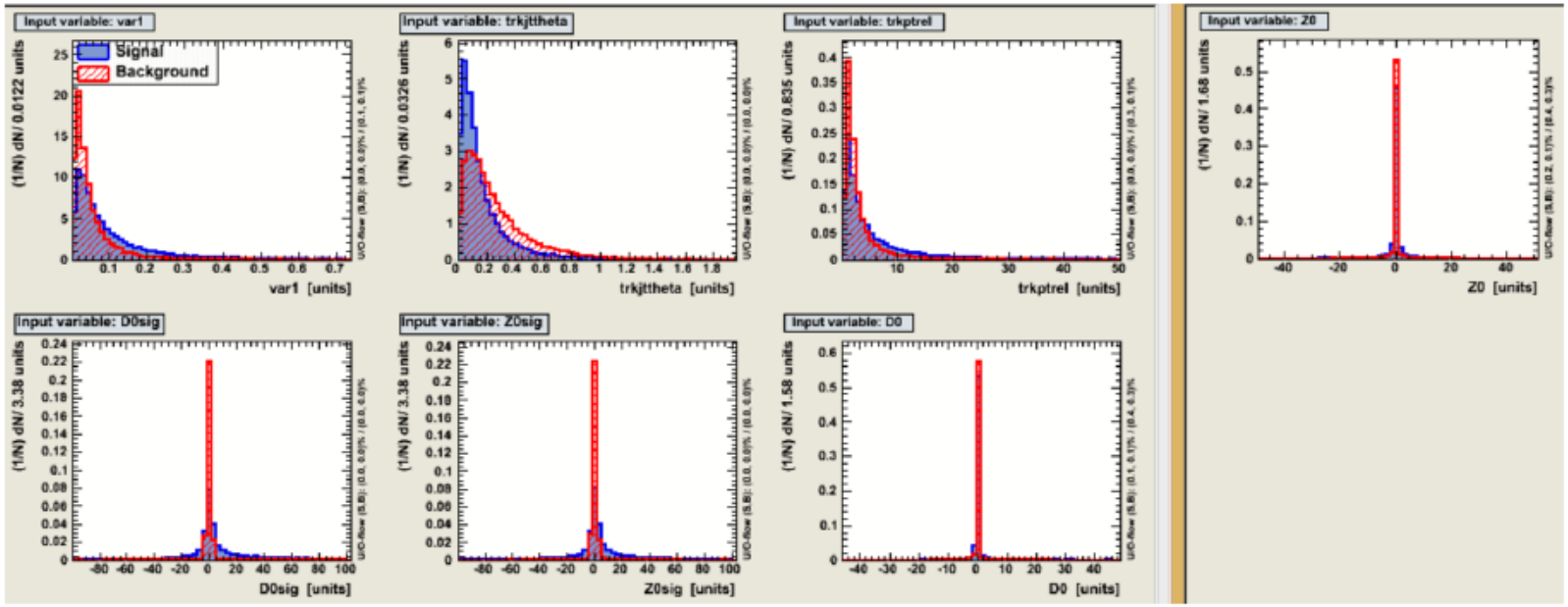
Issues / others

- Systematic errors from flavor tagging
  - Only 'ballpark estimate' available
  - Should be studied in control samples
  - Application to each physics analysis

- Interface to LCIO/Marlin
  - Some problem from multiple PFO collection
  - External jet clustering
- Documentation!

Track MVA(BNess)

- To identify track which comes from heavy flavor particle →using MVA
  - Signal: tracks which come from B mesons or B baryons
  - Background: tracks produced in hadronization process
- Most significant tracks with both plus and minus signed impact parameters in a jet are collected
  - Significance: $sig = \sqrt{(\frac{d_0}{\sigma})^2 + (\frac{z_0}{\sigma})^2}$



BNESS TAGGER FOR FAKE TRACK REJECTION

- Loosen the track selection to try to attach as many tracks as possible to vertices
  - Fake track rate will be increased
- To reject fakes, BNess tagger is used
  - So far, just use BNess(bl)
- So far, only BNess is checked

→some bias for D meson tracks?

Example: looking for single track vertex