Study of Electron Id Performance in Collision Data

Kalanand Mishra

Fermilab

Egamma meeting
(January 20, 2011)

Electron Id performance metric

✦Evaluate the general performance of three classes of electron Id currently available in the market:

- •Egamma/VBTF Working Points (WP),
- •Egamma Cuts-in-Categories (CiC)
- •Egamma likelihood (LH)

For ECAL-seeded electrons

✦ Just for fun I also show how much improvement one can possibly get if one sacrifices simplicity and uses multi-variate (a.k.a "kitchen sink") approach

- •used boosted decision tree for this purpose (plots labeled as "KM")

✦Use the following signal and background samples to evaluate the signal efficiency and fake rates:

Signal:
Probe leg of $Z \rightarrow ee$ when tag≡WP80 and $80 < m_Z < 100$ GeV and pfmet<20 GeV
Signal purity >98%

Background:
Probe leg of Z when tag fails WP95 and $m_Z < 80$ GeV
[also tried fake electrons in QCD jet triggered events in data, the results remain same]
Signal contamination ~ 10^{-3} – 10^{-4}

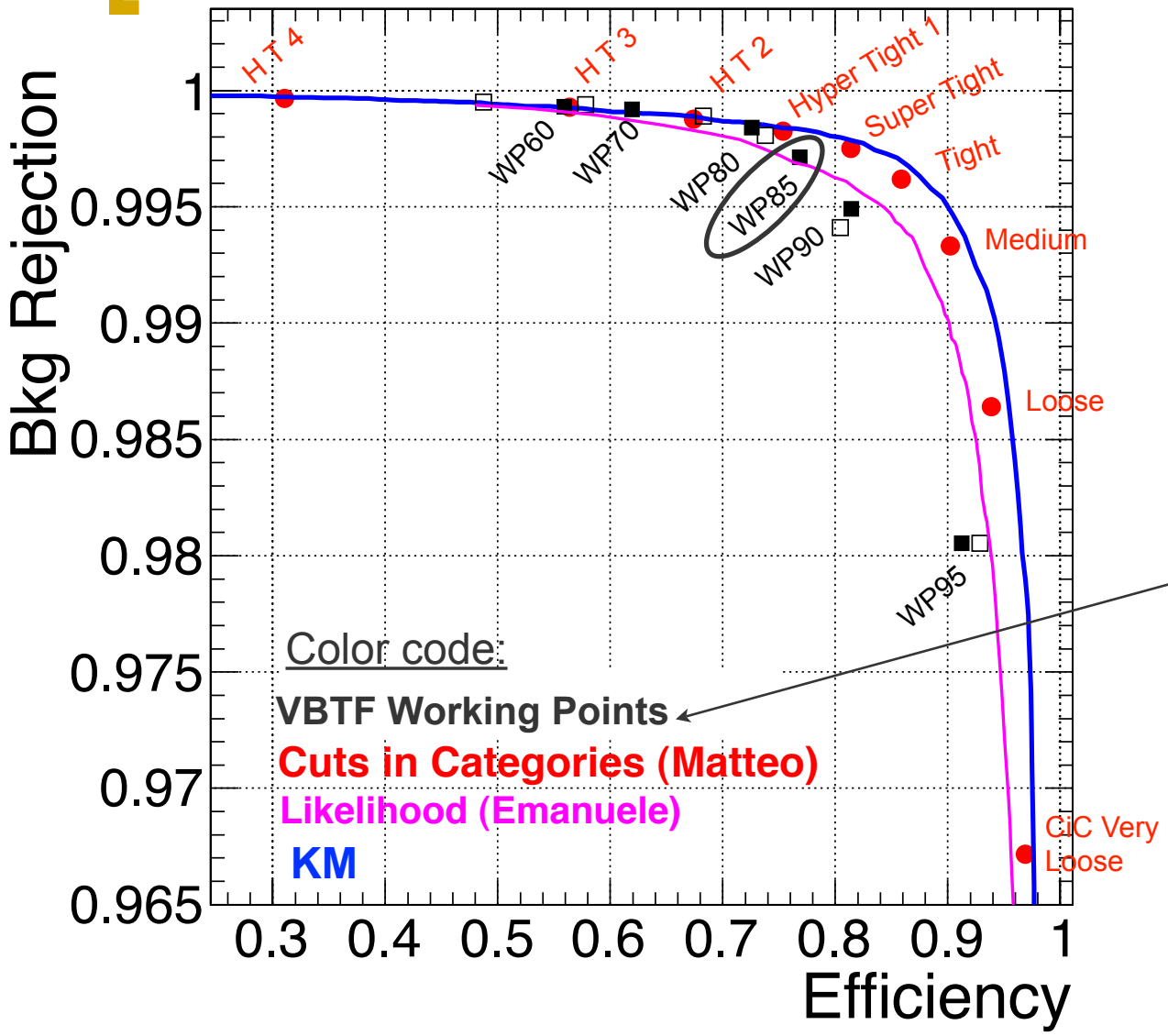
Some details on comparison

- All electron Id selections have been treated on equal footing.
  - efficiency computed exact same way in all cases
  - Apply selections as advertised by developers on twiki/hypernews
  - In case of CiC select events which pass all isolation, id, conversion, and impact parameter flags (i.e., bit ==15)

- The following caveats apply
  - In general the performance depends on the choice of background sample and kinematics
  - In the comparison on next slide the background sample is dominated by generic QCD events (almost no heavy flavor), electron E_T>20 GeV

- To conclude on the performance of each Id we need to look at
  - missing E_T plot for W→ev candidates
    - Bkg in that sample has more heavy flavor
  - Tag+Failed super cluster invariant mass distribution in Z→ee candidates for various electron Id on tag.

Performance: signal efficiency vs bkg rejection

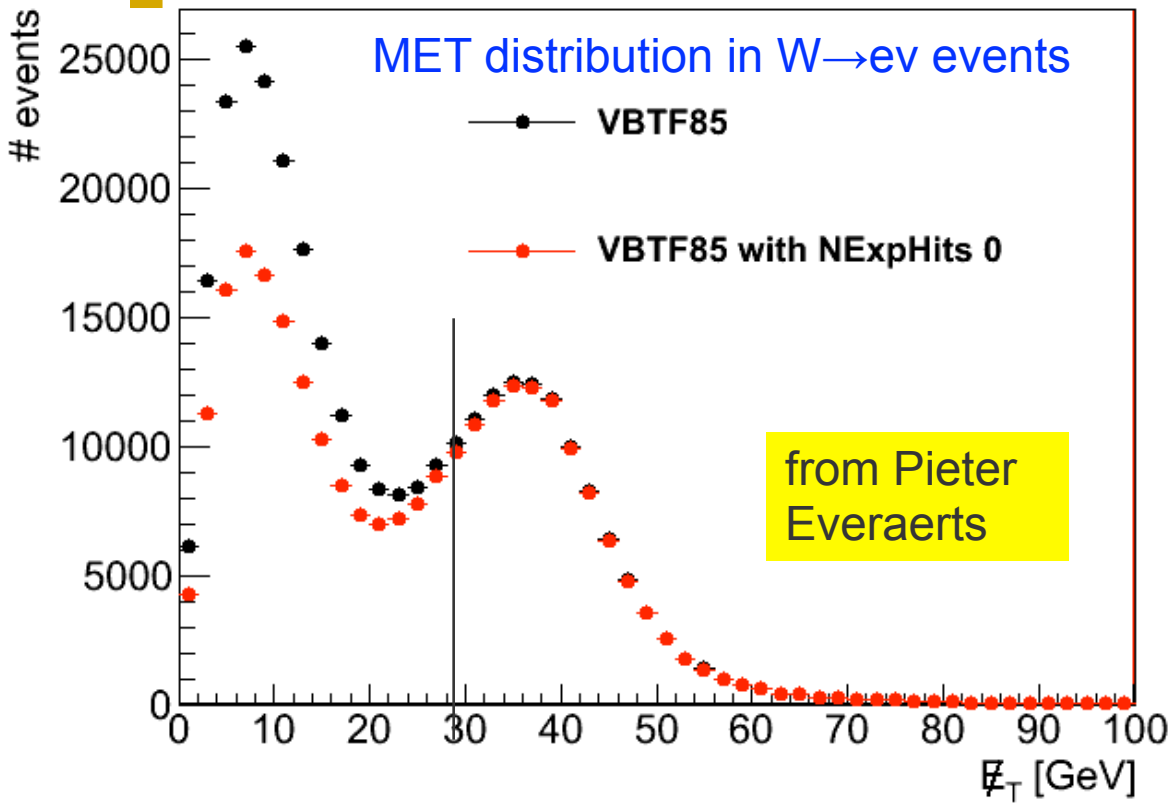


- The higher the curve/ point, the better performing it is
- Bkg rejection = 1 – bkg efficiency

Also, graph with empty squares: WP tuned on data (privately provided by N Rompotis)

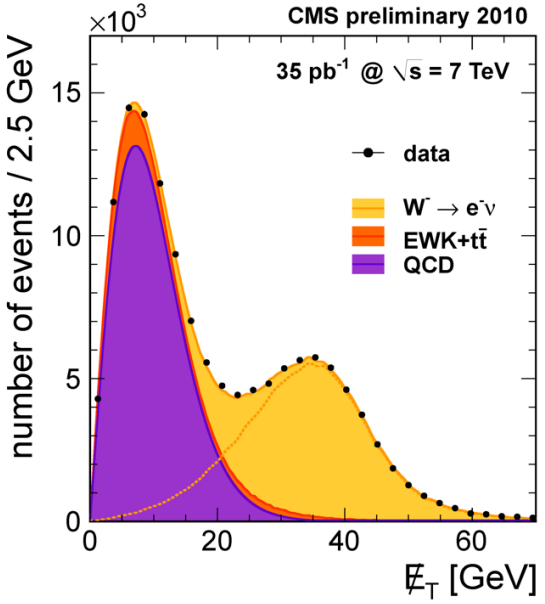
Many thanks to Matteo Sani and Nikos Rompotis for providing me the tuned parameters and feedback

Likely improvement in WP85 performance



from Pieter Everaerts

In CMSSW 3.9 reconstruction NExpectedHits==0 is more optimal for WP85. VBTF will move to WP85 (with $E_T > 25$ GeV) for inclusive W,Z analysis.



✦Currently WP85 requires NExpectedHits==1.
✦Just changing this one requirement (i.e., w/o touching other cuts) will improve the background rejection rate substantially while signal efficiency will remain same.

Validation of the above study

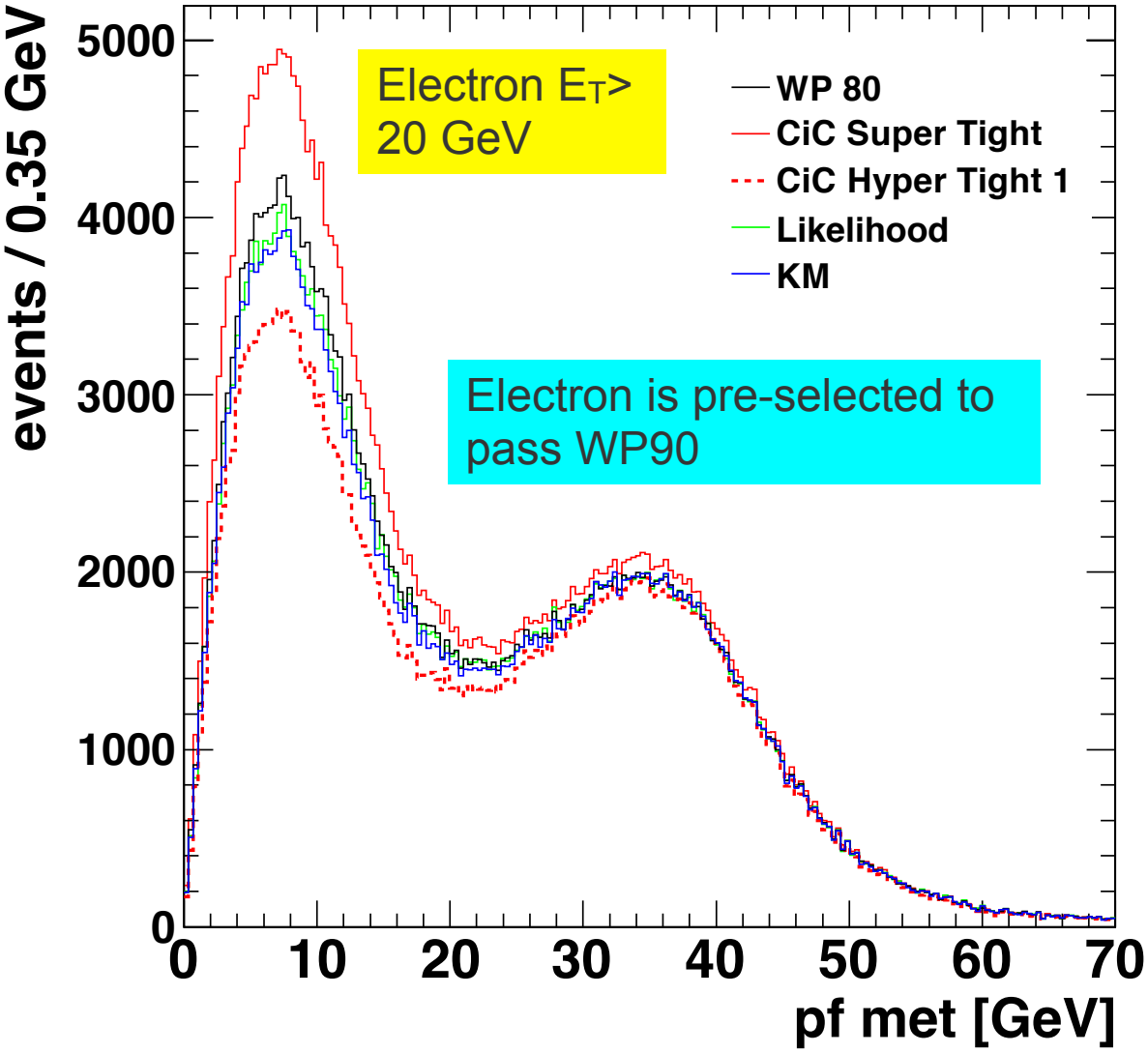
Further, evaluate performance of “80% efficiency point” in two samples:

W→ev: Missing E_T distribution.

- pick “WP80” as reference for comparison, i.e., compare performance of other Ids with respect to WP80
- for eid likelihood and “KM” the strictness cuts are not defined
  - -choose the cut so that it has same efficiency as WP80
  - -this way we can compare the background rejection rate

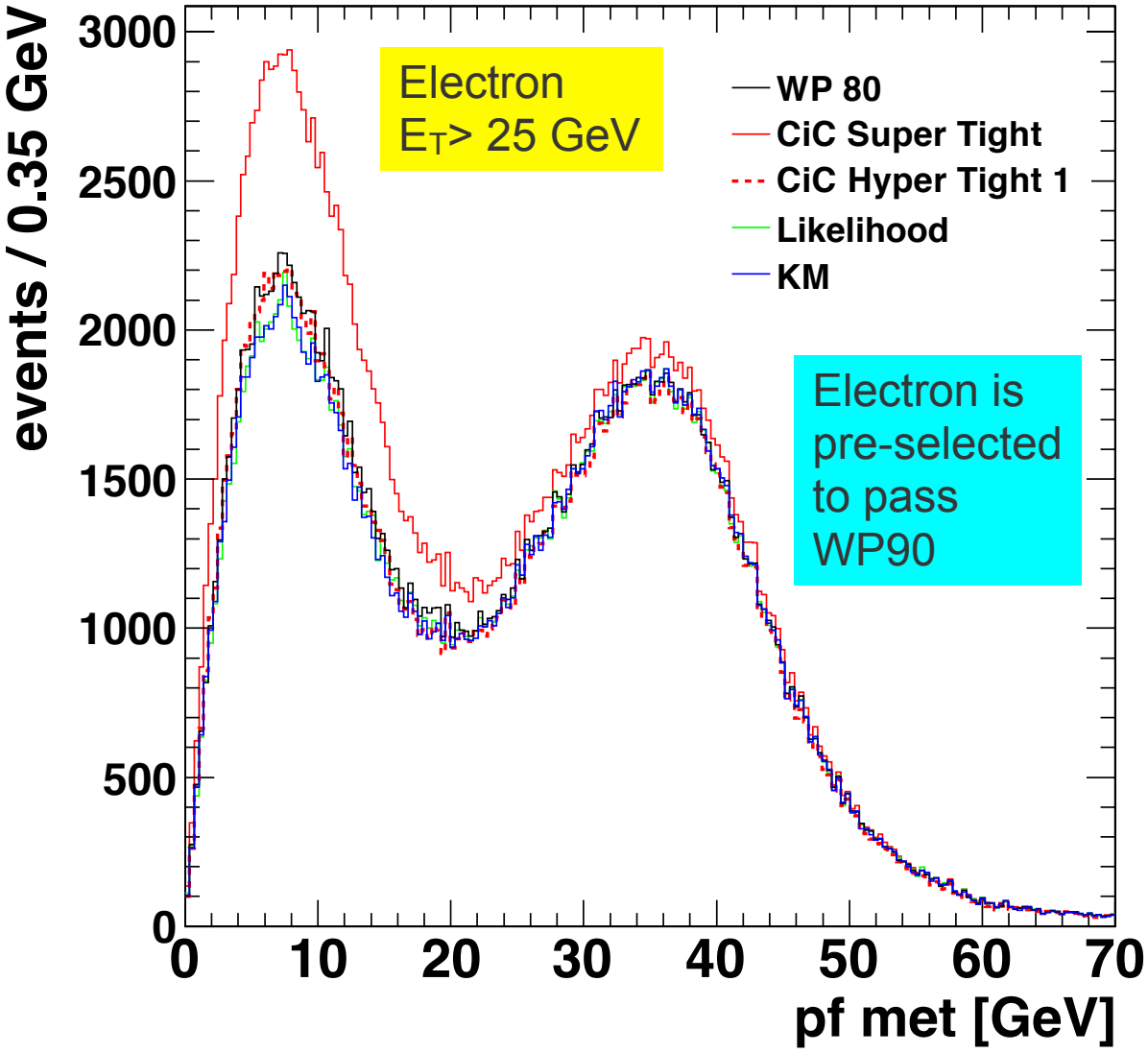
Z→ee: Invariant mass distribution in Tag+Fail sample (e.g., with probe GsfElectrons which fail WP80 selection) which is a major source of systematics in efficiency measurement − and therefore in all precision electroweak analyses.

Id performance: W→ev distribution (E_T>20)



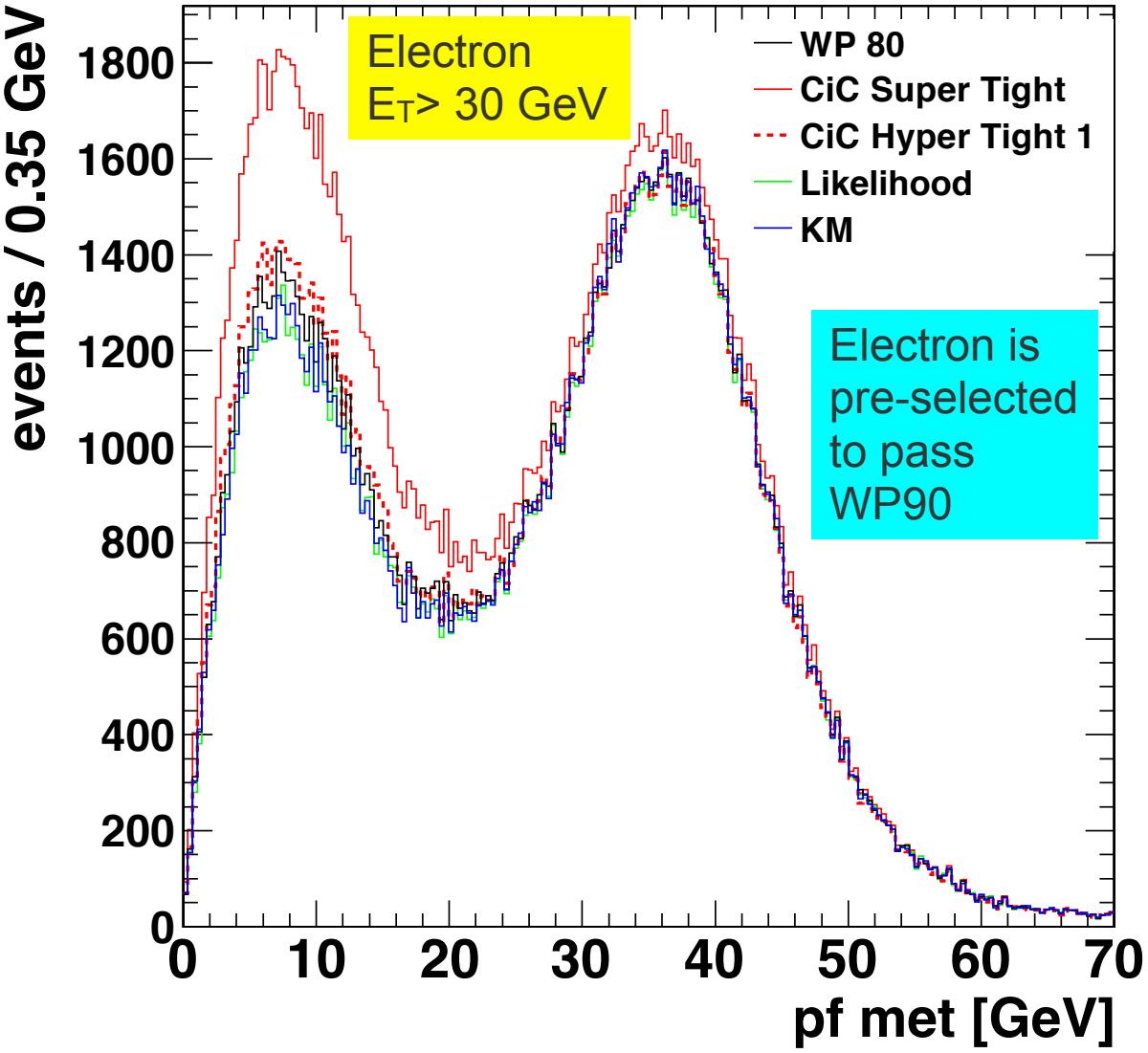
- Among the three Ids WP80 and CiC are performing about the same:
  - CiC Super Tight is significantly looser than WP80 whereas CiC Hyper Tight1 is significantly tighter
- Likelihood has slightly better performance than WP80.

Id performance: W→ev distribution (E_T>25)



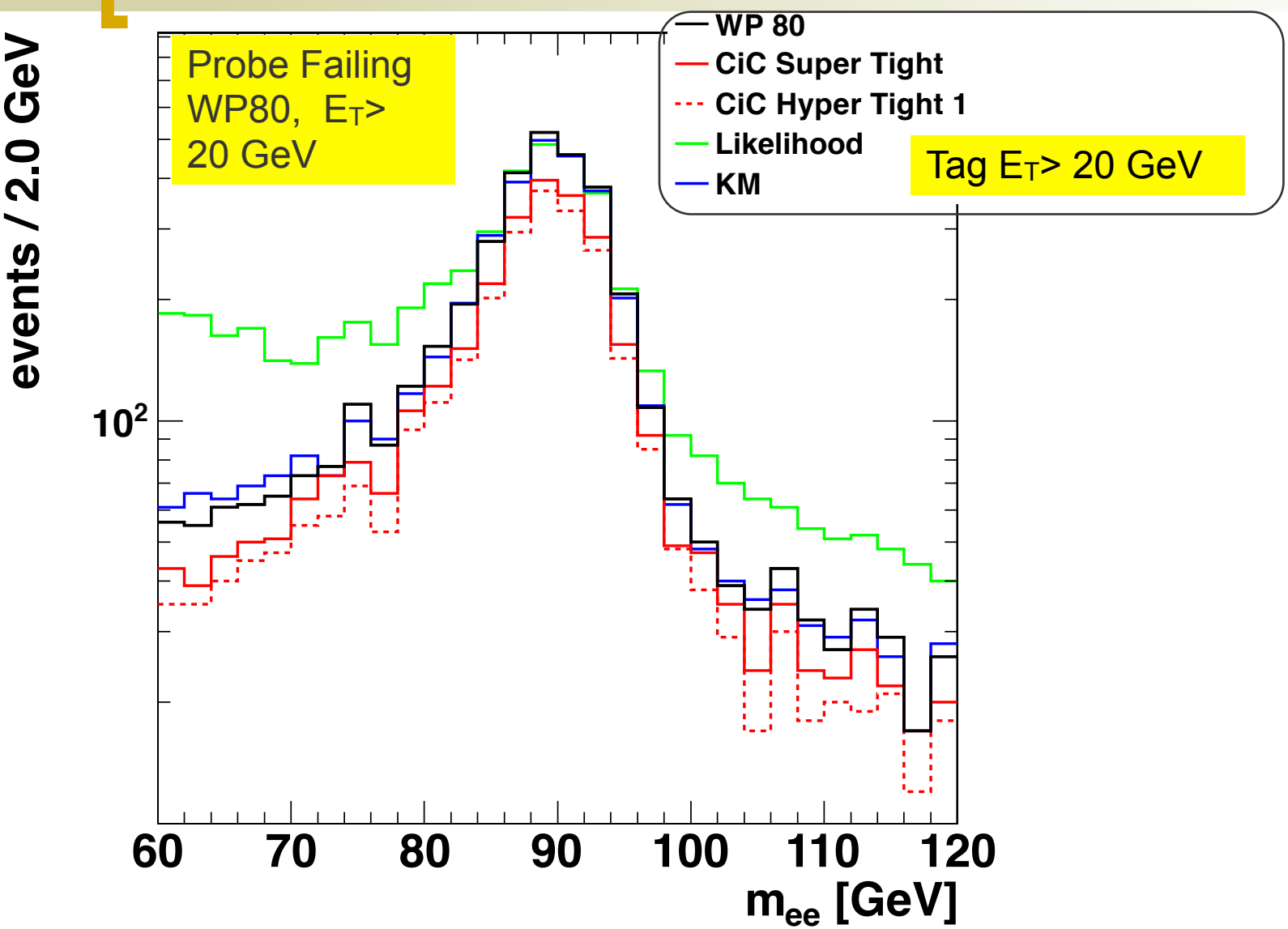
Same conclusions as last slide although higher kinematic cut has improved the signal purity

Id performance: W→ev distribution (E_T>30)

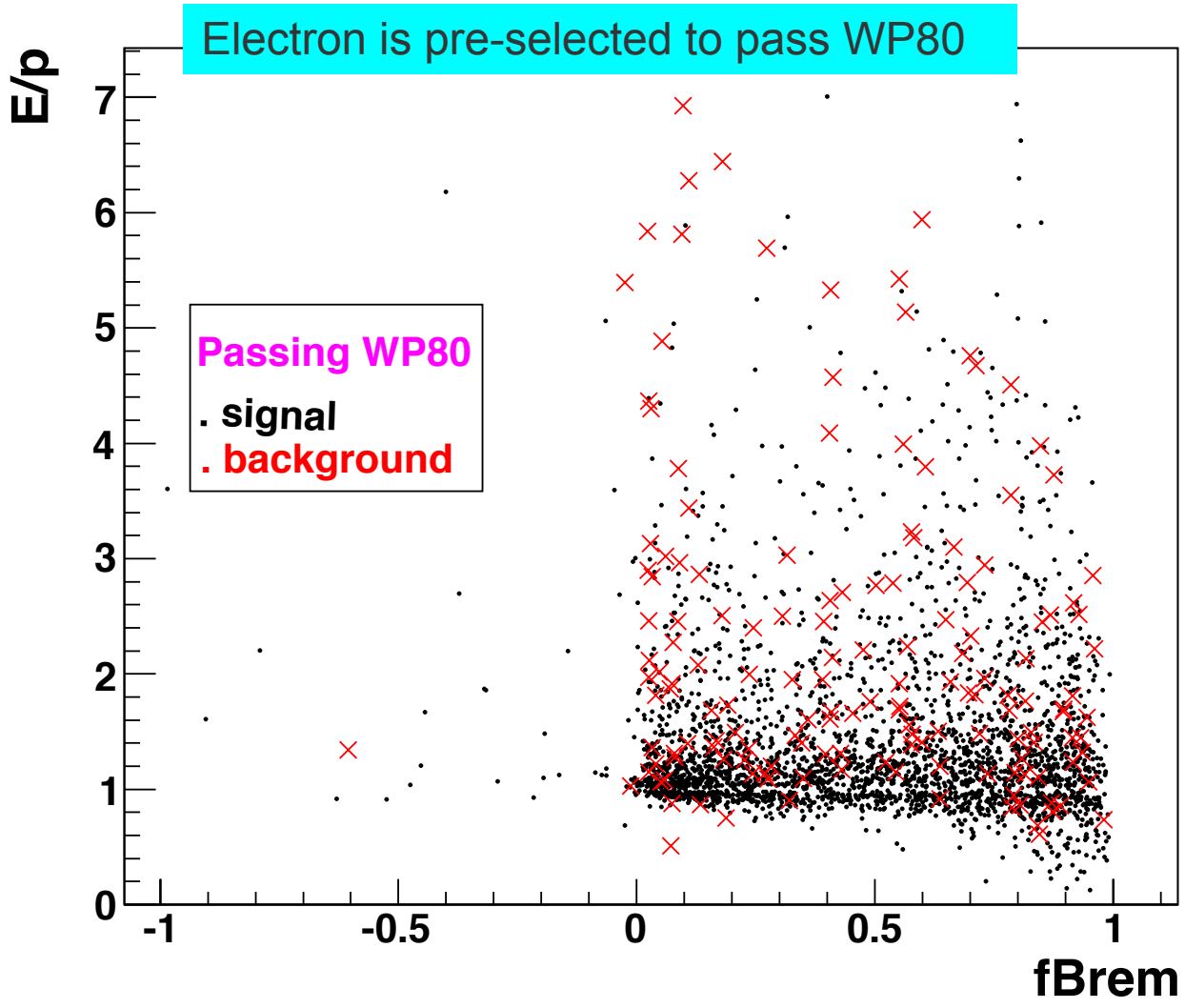


- Now all Ids are performing about the same.
- It seems that for electrons in W,Z sample the E_T-dependent cut in CiC is not any more effective than applying higher E_T threshold and then applying simple cut on isolation, shower shape, and track-cluster matching (+conversion, TIP).

Id performance: TF mass distribution (E_T>20)



Why dividing in categories helps little for tight select.



- There is some structure to exploit here
  - high E/P low fBrem looks background enriched
  - but not nearly enough.
- Applying cuts in categories does not help much for already quite pure electrons
  - no low-hanging fruit beyond WP80

However, in high efficiency and low purity region the categorization helps. This is where CiC is performing better than WP.

BACKUP SLIDES

Recipe used to get Egamma likelihood value

From slide 2 of Emanuele's following presentation:

http://indico.cern.ch/getFile.py/access?contribId=2&resId=0&materialId=slides&confId=111019

Likelihood is not run on standard RECO sequence: need to be re-run on GsfElectrons

Extra tags needed:

- V00-03-14-01 RecoEgamma/ElectronIdentification

I used more recent tag V00-03-21 which also has updated Egamma CiC cuts tuned on data

Sequence to run (likelihood_cff.py):

```
from RecoEgamma.ElectronIdentification.electronIdLikelihoodExt_cfi import *
import RecoEgamma.ElectronIdentification.electronIdLikelihoodExt_cfi
egammaIDLikelihood = RecoEgamma.ElectronIdentification.electronIdLikelihoodExt_cfi.eidLikelihoodExt.clone()
electronIDLH = cms.Sequence( egammaIDLikelihood )
```

How to access the likelihood output in an analyzer:

```
Handle<GsfElectronCollection> electronCollection;
if ( iEvent.getByLabel(electronCollection_, electronCollection) ) {

  std::vector<edm::Handle<edm::ValueMap<float> > > eIDValueMap(1);
  if ( iEvent.getByLabel( electronIdLikelihoodLabel_ , eIDValueMap[0] ) ) {
    const edm::ValueMap<float> & eIDmapLikelihood = * eIDValueMap[0] ;

    for(int index = 0; index < (int)electronCollection->size(); index++) {
      edm::Ref<reco::GsfElectronCollection> electronRef(electronCollection,index);

float result = eIDmapLikelihood[electronRef]
```

One needs to apply conversion rejection (dist, dcot, misHits) and isolation cuts separately. These are not included in LH computation. At least this is what I have learned.

Recipe used to get Egamma CiC bit values

Check out: V00-03-21 RecoEgamma/ElectronIdentification

In python config:

```
process.load
("RecoEgamma.ElectronIdentification.cutsInCategoriesElectronIdentificationV06_DataTuning_cfi")
```

Include all the modules in the path and then access the value map as described on the previous slide

Recipe used for Egamma/VBTF "working points"

Follow instructions at:

https://twiki.cern.ch/twiki/bin/view/CMS/SimpleCutBasedEleID#Selections_and_How_to_use_them

Use separate relative isolations in all cases.

Details on "KM" selection

Described in the following presentation:

http://indico.cern.ch/getFile.py/access?contribId=3&resId=0&materialId=slides&confId=108328

A few pointers:

-Use boosted decision tree as classifier
-Following are the 23 input variables (the classifier can deal with correlated variables)

deltaEta, deltaPhi, sigmaIetaIeta, E/p, fBrem, H/E, charge, eta, ecalIso, hcalIso, trackIso, NExpectedHits, convDist, convDcot, deltaEtaOut, deltaPhiOut, e1x5/e5x5, e2x5/e5x5, eSeedClusterOverP, classification, track impact parameter, track d0, number of primary vertex in the event