Particle Identification @ LHCb

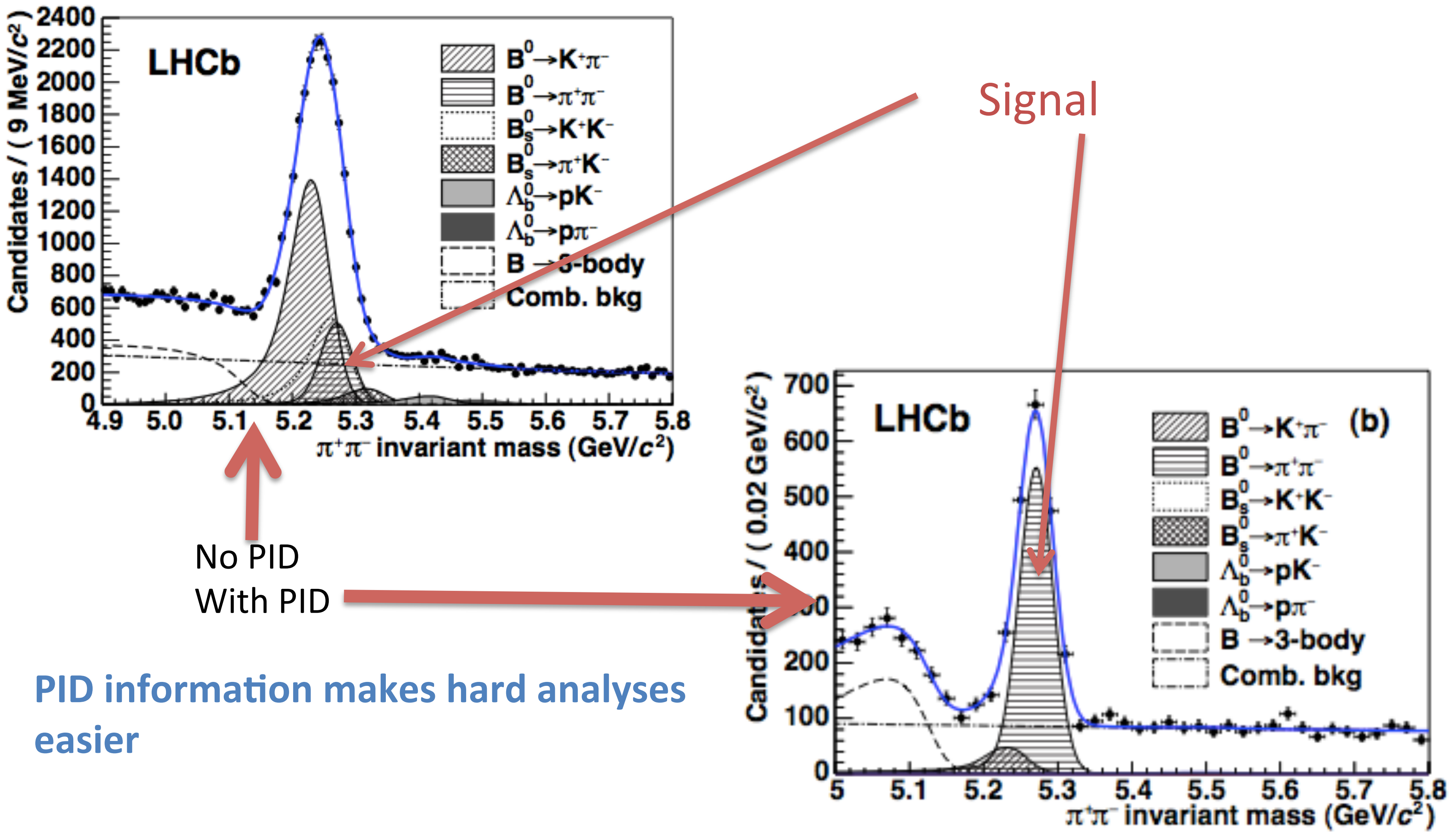
Sneha Malde

With a lot of credit to Andrew Powell

Outline

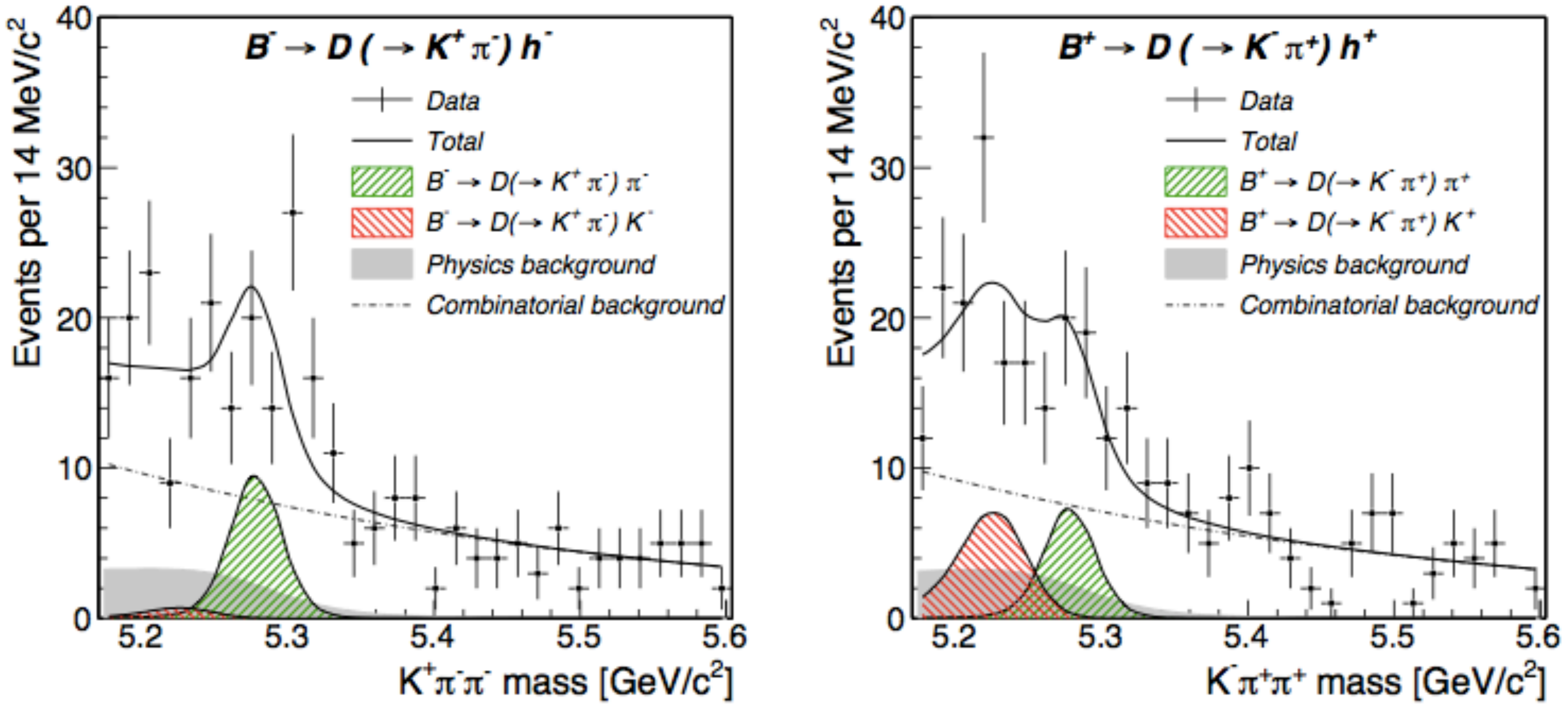
- Overview of PID at LHCb
- Focus on hadron identification
- Using PID information in an analysis
- Performance plots example
- PIDCalib Tutorial (if time)

Why does it matter?

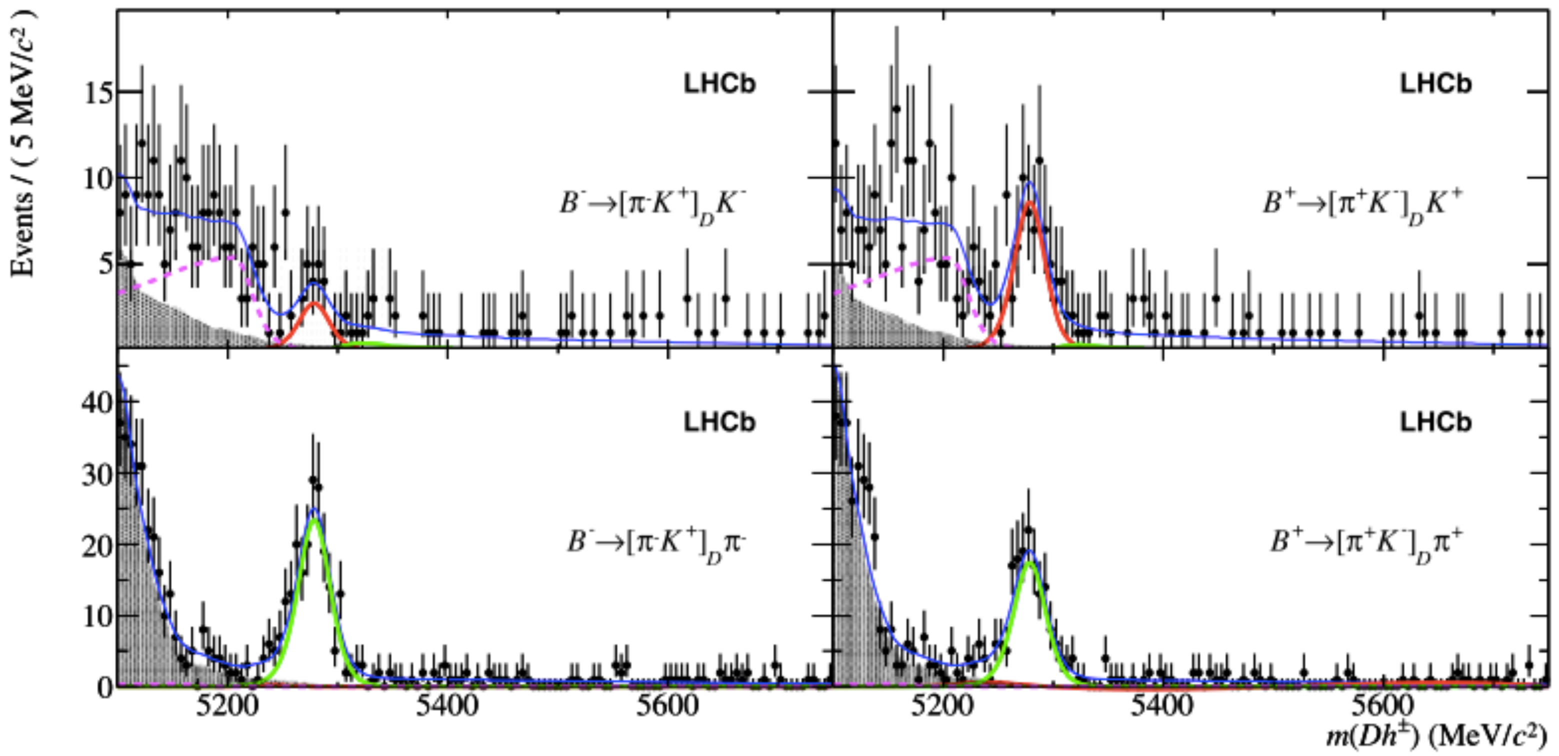


PID information makes hard analyses easier

Why does it matter (2)



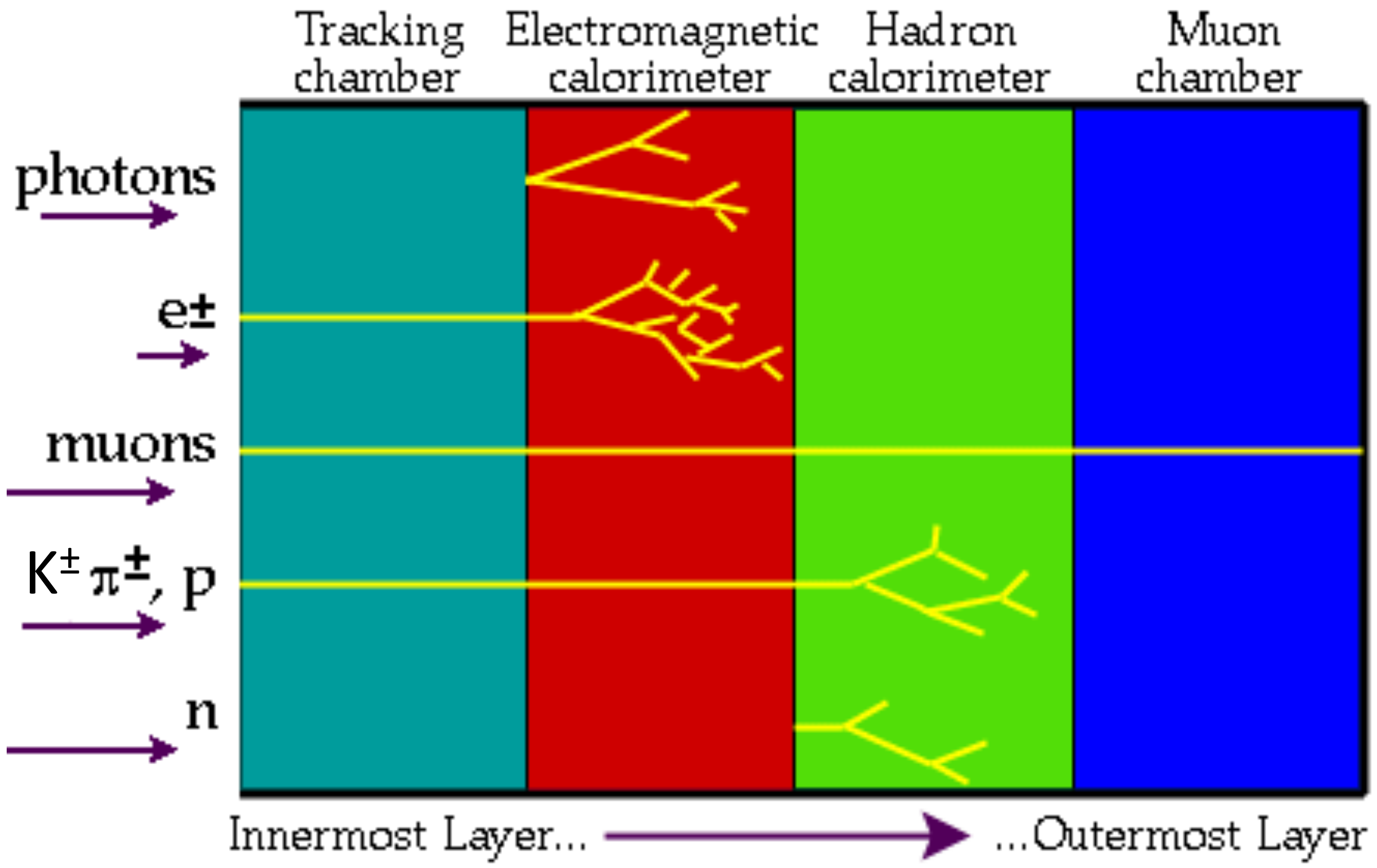
First measurement of the suppressed ADS decays at a hadron collider
CDF had weak K-pi separation (better than nothing).



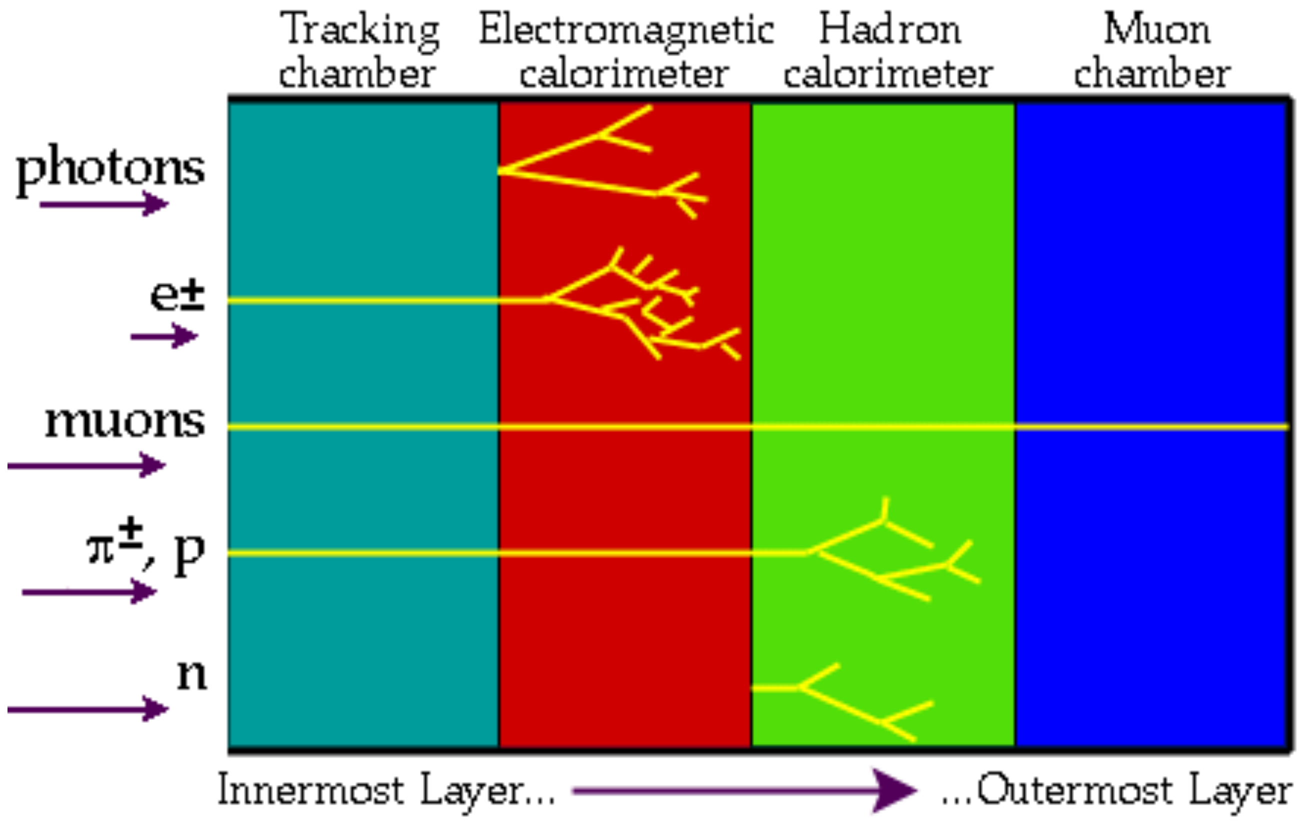
Equivalent plot from LHCb. Yields are only ~x3 higher, but the clean nature of the peaks makes considerable difference to the impact of the result.

PID information makes otherwise impossible analyses possible

Material interactions: The basics



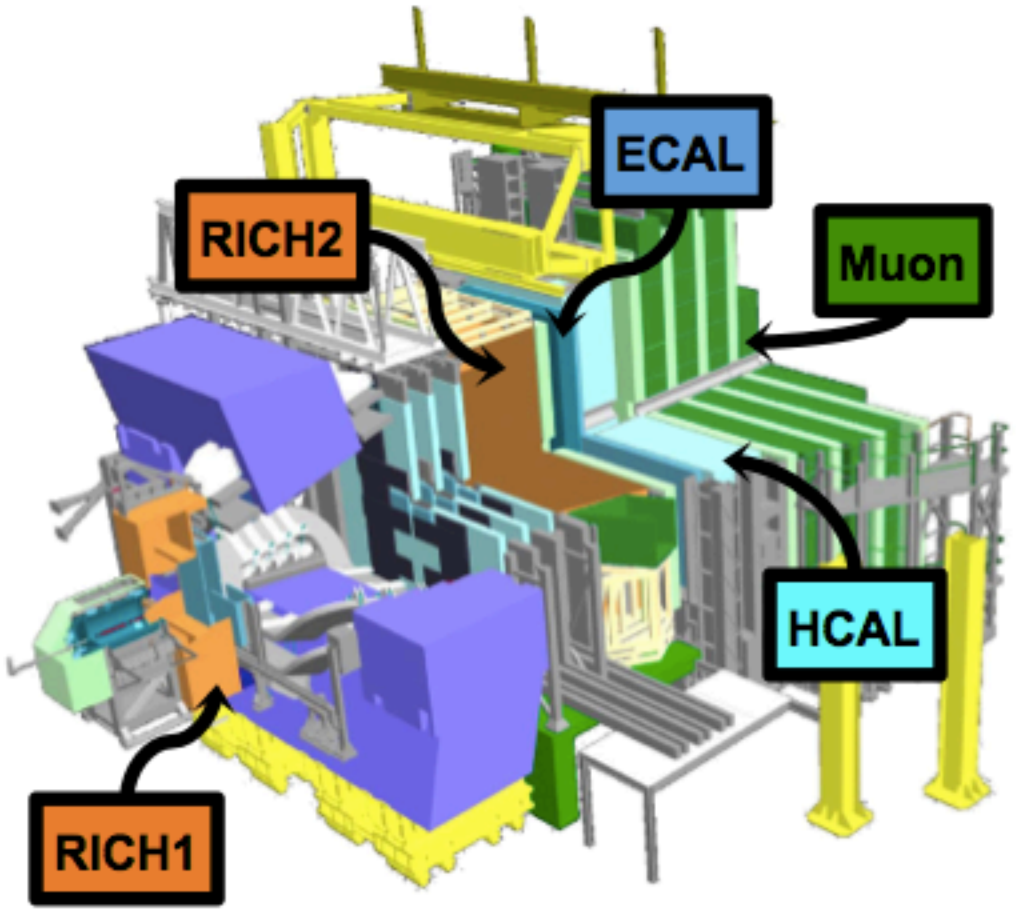
Material interactions: The basics



Explains what detector components help identification.

Also shows that for K/pi/p separation something else is required

Particle ID subdetectors



- Single-arm spectrometer
- Covering 2< η <5
- PID provided by three groups of detectors:

1. Ring Imaging Cherenkov (RICH) detectors
2. Calorimeters
3. Muon Chambers

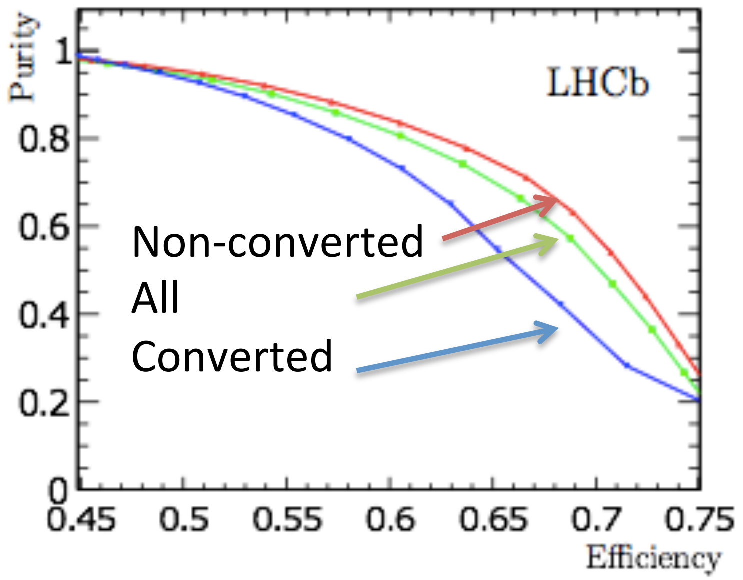
Photon ID

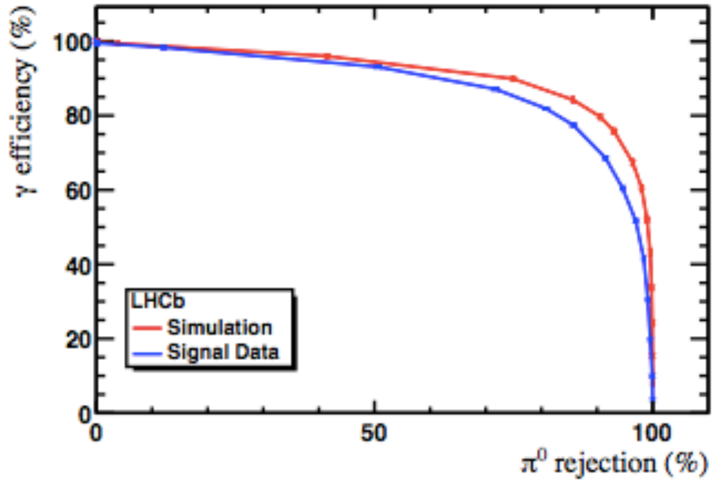
Calorimeters provide ID for photons, electrons and pi0.

Since electrons are also charged they are required to have an associated track.

Shape of the cluster allows for photon/pi0 separation.

Photons come in two types, non-converted & converted (i.e interacted with an upstream detector to make a e^+e^- pair)

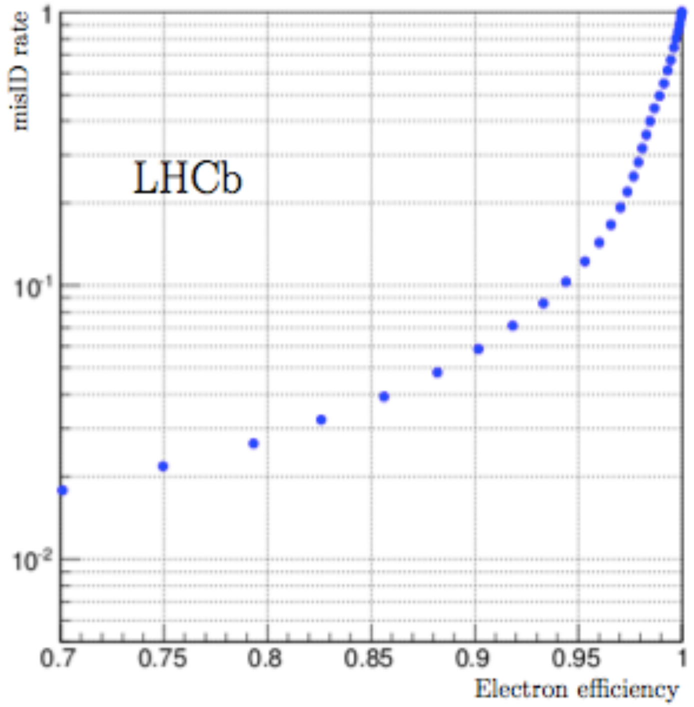




Electron ID

Likelihood is constructed taking into account all CALO sub-detectors

$$\Delta\log\mathcal{L}^{\mathrm{CALO}}(e-h) = \Delta\log\mathcal{L}^{\mathrm{ECAL}}(e-h) + \Delta\log\mathcal{L}^{\mathrm{HCAL}}(e-h) + \Delta\log\mathcal{L}^{\mathrm{PS}}(e-h)$$



ID efficiencies are determined via a tag & probe method using J/Ψ→ee

Reconstruct the J/Ψ signal

Tag electron has ID cuts placed on it (necessary for S/B)

Other electron is the probe. You know that it is an electron and but since no ID information is used it gives a unbiased estimate of the electron ID efficiency.

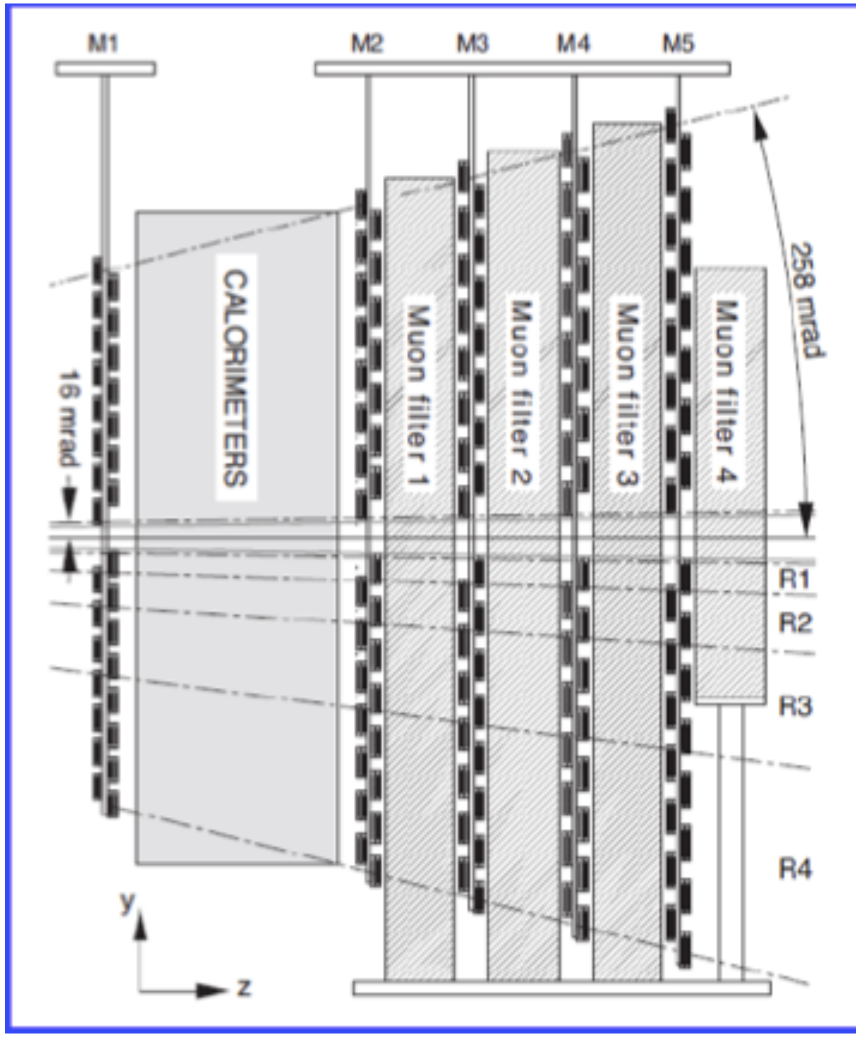
Muon ID

3GeV muon will make it to M2/M3

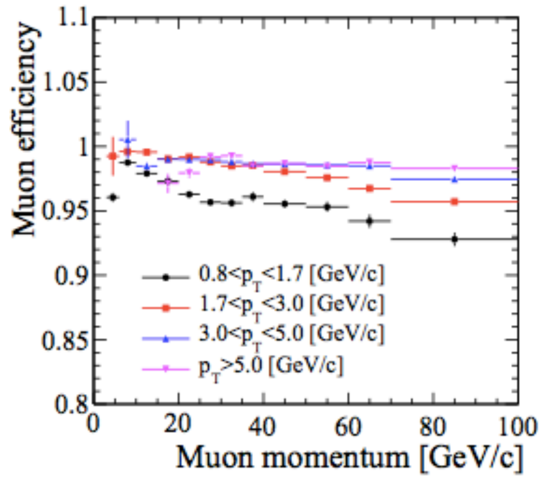
6 GeV muon will cross full system

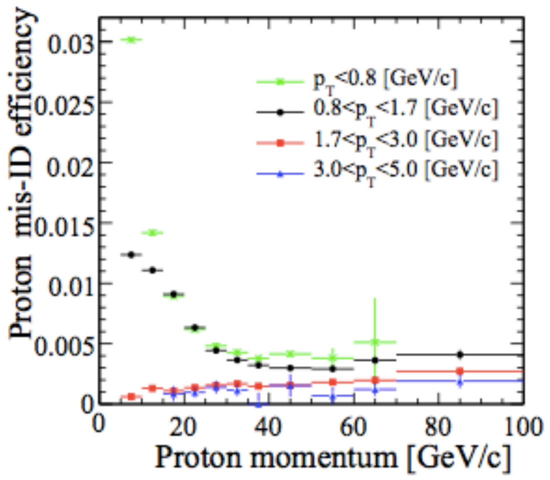
Muons are identified by extrapolating the trajectory of tracks in the tracking system to the muon detectors.

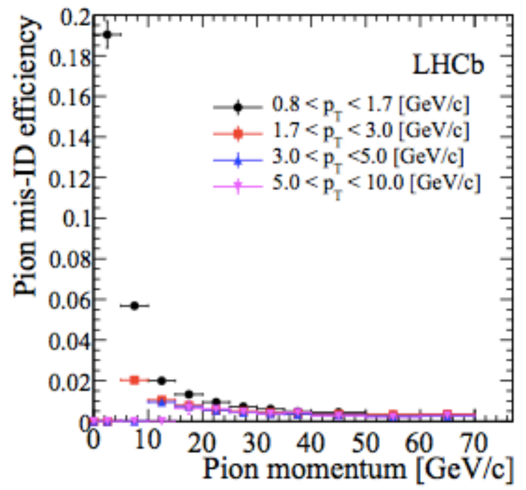
Search window are parametrised as a function of momentum to maximise efficiency while keeping pion mis-id low.

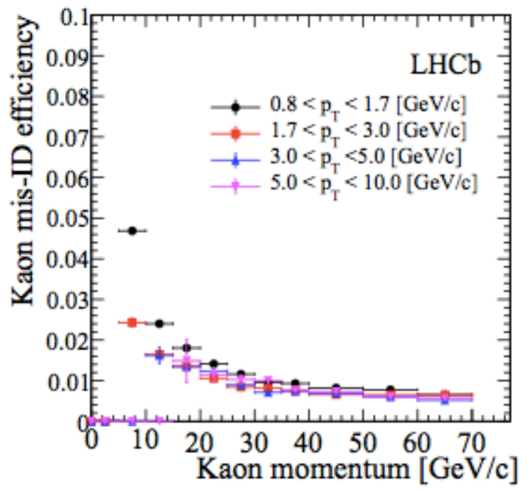


Muon ID









Reasons for mis-ID

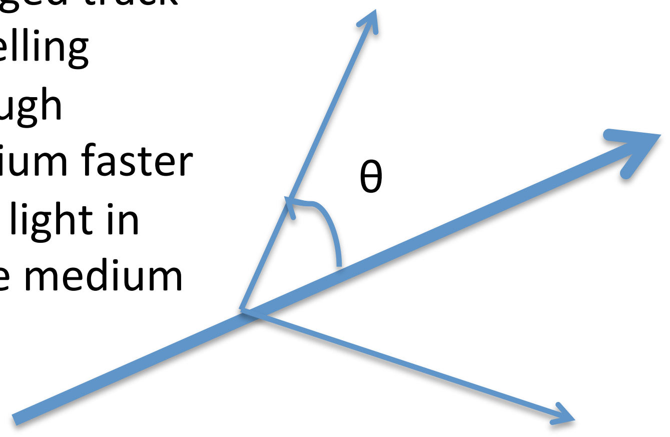
- Spurious hits in muon system
- Real muon in event with similar trajectory
- As P increases the windows for search decrease and more hits are required, hence the drop in mis-id rates.
- Decays in flight for kaons and pions hence larger mis-id rates than for protons

The RICH(es) - principle

Detectors considered so far cannot tell hadron species apart
Ability to do so allows us to fully exploit flavour physics

Use the principle of cherenkov radiation

Charged track travelling through medium faster than light in same medium



Photons emitted by medium

$\text{Cos}(\theta) = 1/(n\beta)$

n – the refractive index

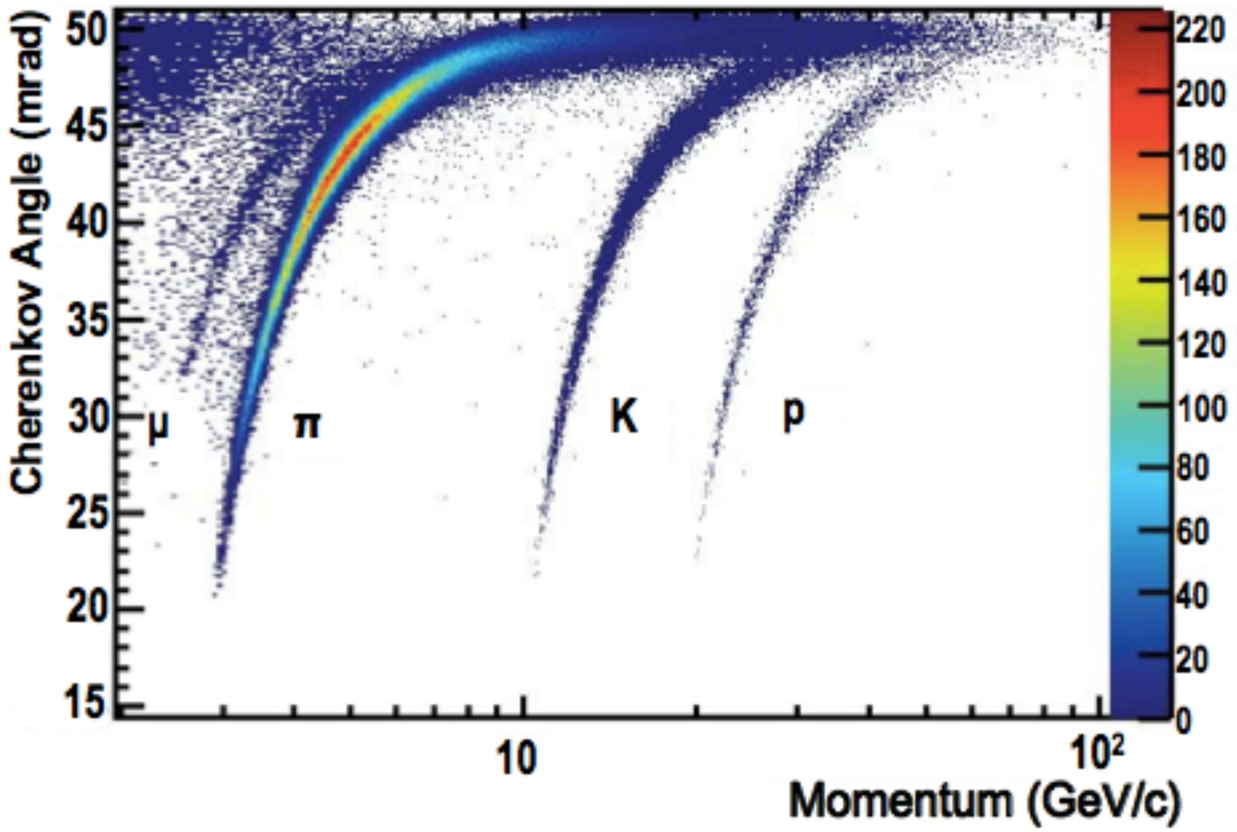
β – the velocity of the track

Simplistic principle:
Track momentum known through the tracking system

Angle θ measured from the emitted light → speed.

With speed and momentum known so is the mass, which identifies the particle

Cherenkov angle vs P



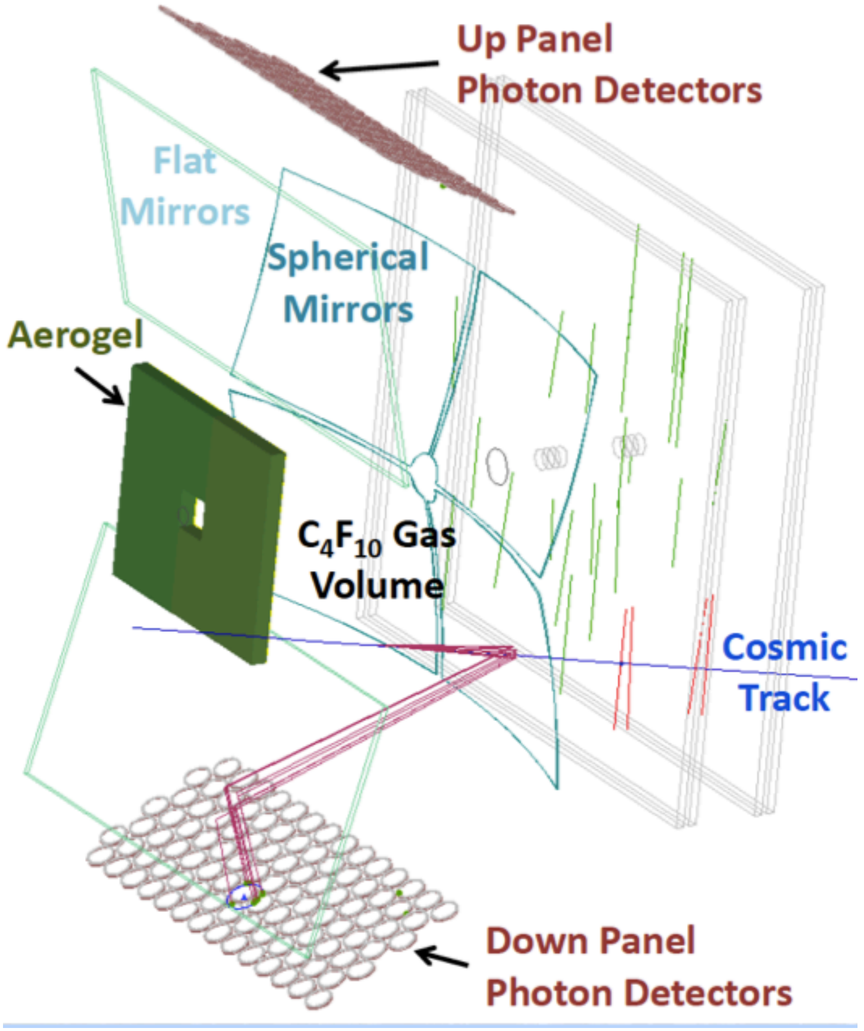
RICH setup

2 RICH detectors

RICH -1 radiator is C_4F_{10}
Provides separation for the lower momentum tracks (up to ~ 30GeV)

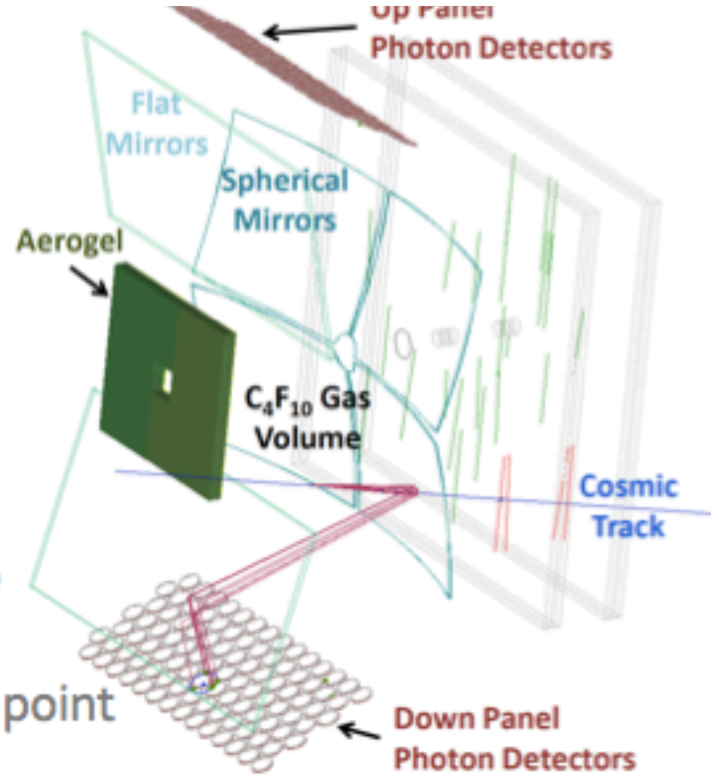
RICH -2 radiator is CF_4
Provides separation for higher momentum tracks up to ~100GeV.
Smaller angular acceptance (but this is where the high-P tracks are)

Arrangement of mirrors focuses the the rings onto photon detector array outside of the LHCb acceptance

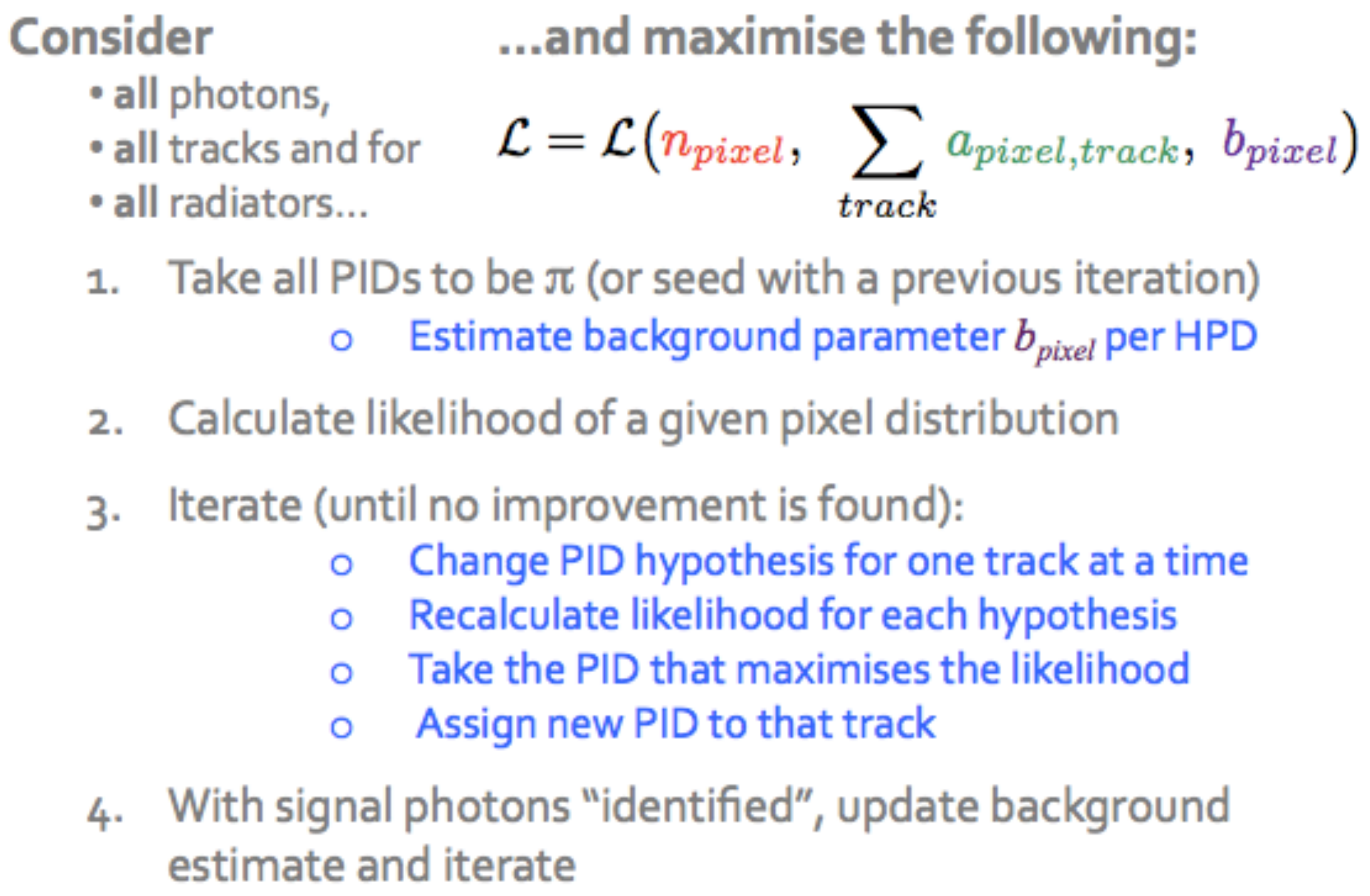


RICH reconstruction

- **Interpret HPD info.**
  - Decode raw buffer data
  - Create pixels
  - Apply hit cleaning
  - Translate pixel hit into a spatial position
- **Select Tracks**
  - Identify good tracks
  - Reject those that don't transverse the RICH sub-detector
  - Determine radiator entry/middle/exit point
- **Reconstruct Photon Candidates**
  - Assume photon emission point to be the middle of the track
  - Pair-up photons and tracks and compute photon parameters



RICH likelihood determination



Consider

- all photons,
- all tracks and for
- all radiators…

…and maximise the following:

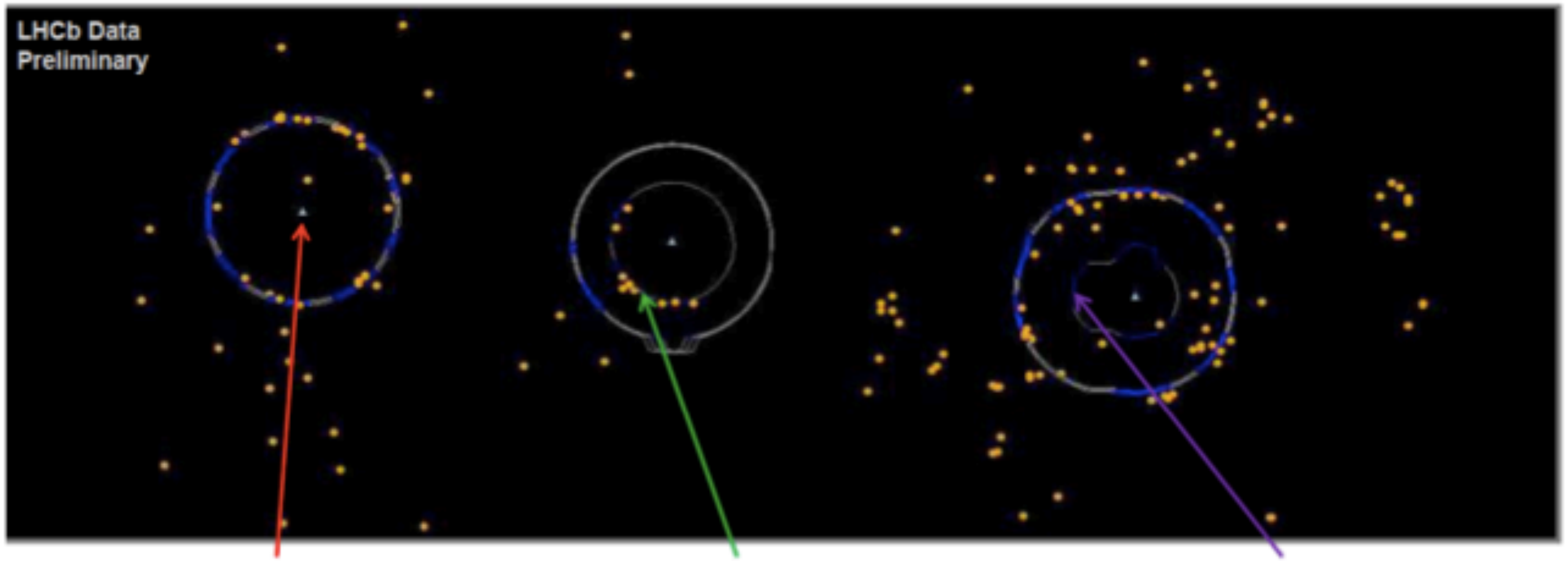
$$\mathcal{L} = \mathcal{L}\left(n_{pixel},\ \sum_{track} a_{pixel,track},\ b_{pixel}\right)$$

1. Take all PIDs to be π (or seed with a previous iteration)
   - Estimate background parameter b_{pixel} per HPD
2. Calculate likelihood of a given pixel distribution
3. Iterate (until no improvement is found):
   - Change PID hypothesis for one track at a time
   - Recalculate likelihood for each hypothesis
   - Take the PID that maximises the likelihood
   - Assign new PID to that track
4. With signal photons "identified", update background estimate and iterate

RICH rings

An event display from real data show "rings" projected on to RICH1's photon detector plane:

Detector acceptance



Saturated track: particle hypotheses indistinguishable

Photons clearly favour the Kaon ring hypothesis

Ring distortions due to detector geometry

What is the thing in my ntuple

- Particle likelihoods are determined separately by the CALO and MUONs as well
- A combined, overall, likelihood can be determined for each particle type, e.g. for hadrons:

$$\mathcal{L}(h) = \mathcal{L}^{\mathrm{RICH}}(h) \cdot \mathcal{L}^{\mathrm{CALO}}(\mathrm{non}\ e) \cdot \mathcal{L}^{\mathrm{MUON}}(\mathrm{non}\ \mu)$$

- These are referred to as the **COMBINED** $\Delta\log\mathcal{L}$'s
  - This is what you get by default in `TupleToolPID`

$$\begin{aligned}\Delta\ln\mathcal{L}_{K\pi} &= \ln\mathcal{L}(K) - \ln\mathcal{L}(\pi) \\ &= \ln\mathcal{L}(K-\pi)\end{aligned}$$

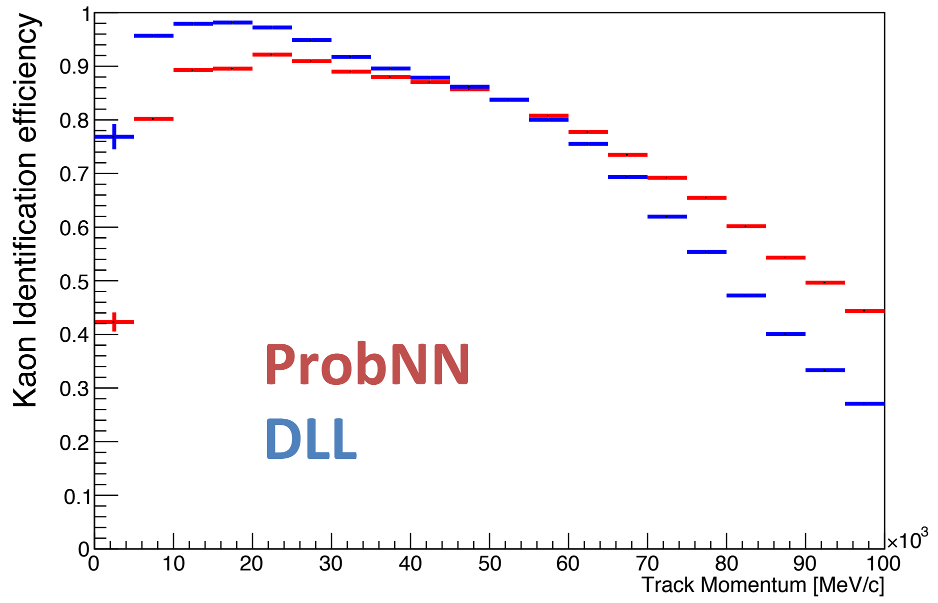
Difference always against the pion hypothesis.
DLL(P-K) = DLL(P-pi) – DLL(K-pi)

No I have something different

- ProbNN variables are newer (been around for a while)
- Neural network trained on MC using:
  - RICH, CALO, MUON PID
  - RICH Rad. Threshold Info
  - VELO, Tracking information
- Dedicated network for each hypothesis
- Bayesian Probability output [0,1]:
  - ProbNN{K, pi, p, mu, e}
  - Also available in DaVinci TupleToolPID
- Come in different versions, depending on the training
  - V1 – old. Don't start using this
  - V2 – what most people use
  - V3 – quite new. Should see difference in performance for ProbNNe – but its not better across the board

ProbNN vs DLL ?

- They have a different performance as a function of the track kinematics
- Depending on your needs you may find one or the other better.
- ProbNN has improved
- I have the impression that ProbNN is more heavily used
- I would recommend looking at both to make sure you pick the optimal choice
- Its not a problem to use a mixture (of ProbNNs or ProbNN + DLL)



- Overall cut on DLL and ProbNN chosen to have same efficiency (75%)

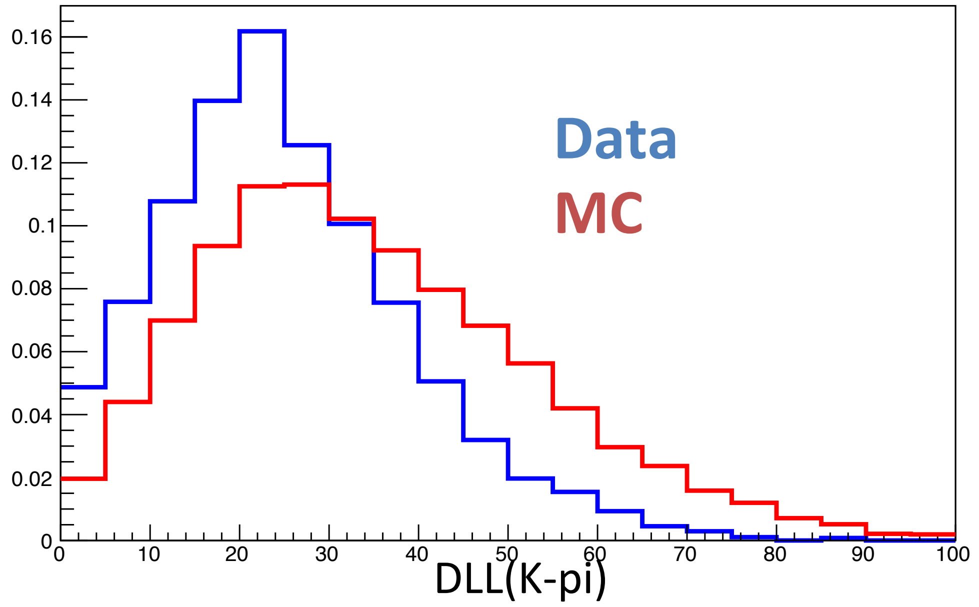
PID and my analysis

You use PID information to

- Select the particle types in your signal channel
- Reject the particle types that cause backgrounds

- Data / MC agreement poor
  - DLLK for Kaons
  - Data is 98% pure
  - Difference not bkg
  - Situation not better for the ProbNN var.

  - If you use the MC PID variables for optimisation you won't do the best job
  - If you use the MC PID variables for efficiencies you'll get the wrong value



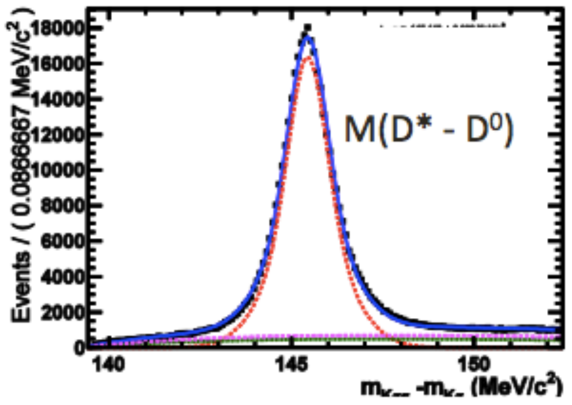
Necessary to find channels data that will help us understand the response

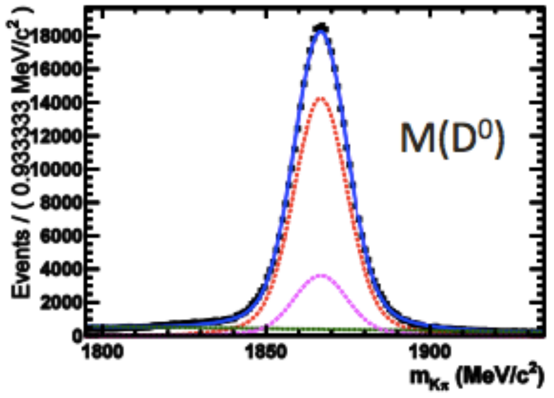
Data driven calibration

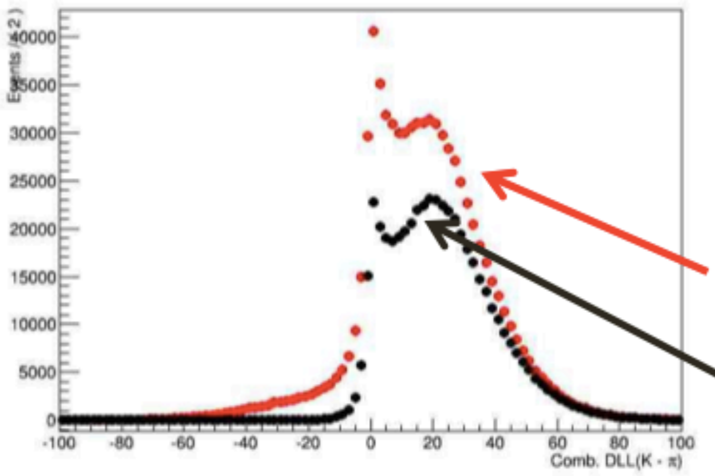
"Known" particle types:

$D^{*+} \rightarrow D^0(\underline{K^-\pi^+})\pi_S{}^+$

- Calibration track type determined by charge.
- No PID requirement in the calibration stripping line for this decay.
- Wrong-sign decay vetoed.
- Background remains so sPlot method used to assign tracks a signal weight.
- 2-D fit to the D* and D^0 mass.







DLLK of all "K" tracks

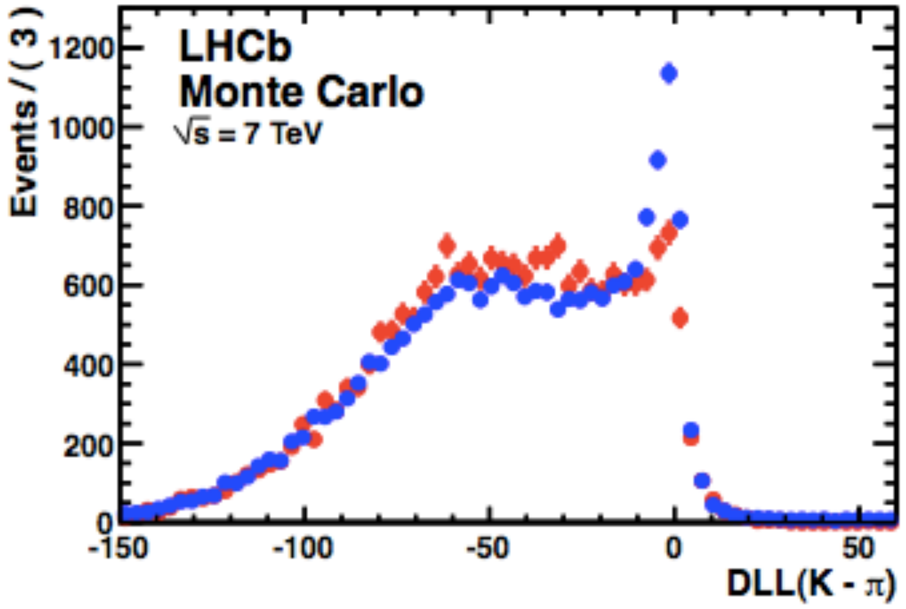
Background subtracted distribution.

PID tables?

Now that a Kaon & Pion calibration sample exists is there a look up table that tells me the efficiency for a given cut?

No – things aren't that simple.

All tracks don't have the same response

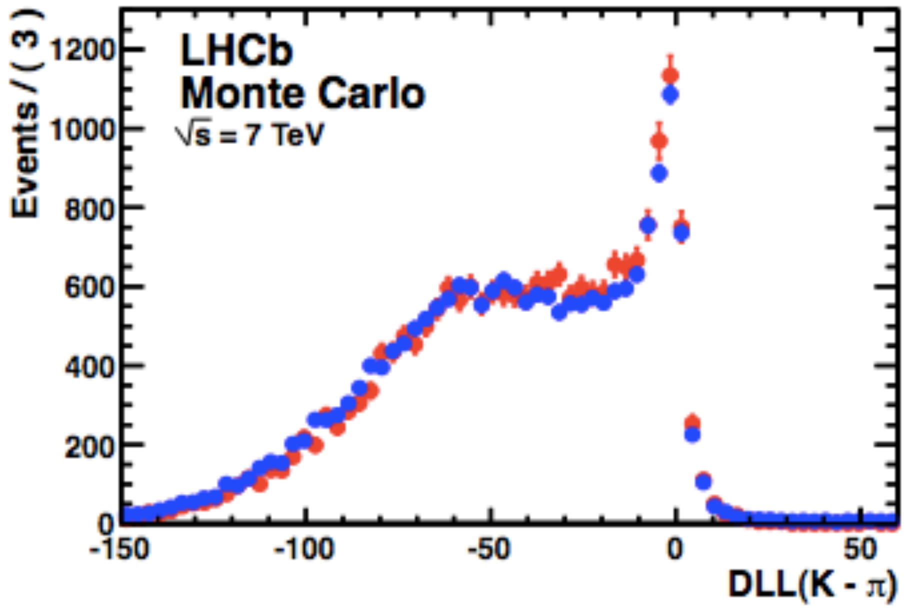


🟥 Calibration π

🟦 Signal bachelor π

Binning	χ^2/NDF
None	5.27

PID dependencies



● Calibration π

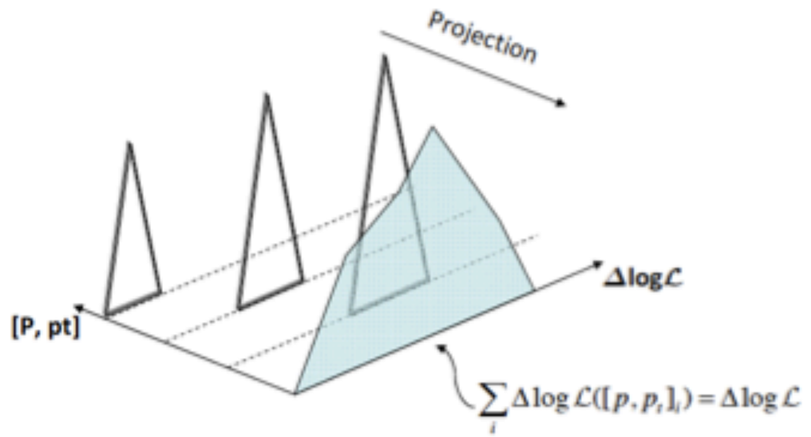
● Signal bachelor π

Binning	χ^2/NDF
None	5.27
A	2.29
B	1.93
C	1.63

A – reweight in momentum
B – reweight in momentum and pseudorapidity
C – reweight in momentum, pseudorapidity and nTracks

Calibration method

- A data-driven method to use calibration tracks to determine ε_{PID} for a given signal track


Projection
$\Delta\log\mathcal{L}$
[P, pt]
$\sum_i \Delta\log\mathcal{L}([p,p_t]_i) = \Delta\log\mathcal{L}$

- If one bins in PID dependent variables, then all tracks residing in a given bin (no matter their origin) will have consistent PID decisions

Clearly there is a trade off: Consistent PID decisions across a bin requires small bins
Statistics limits how small the bin can be.

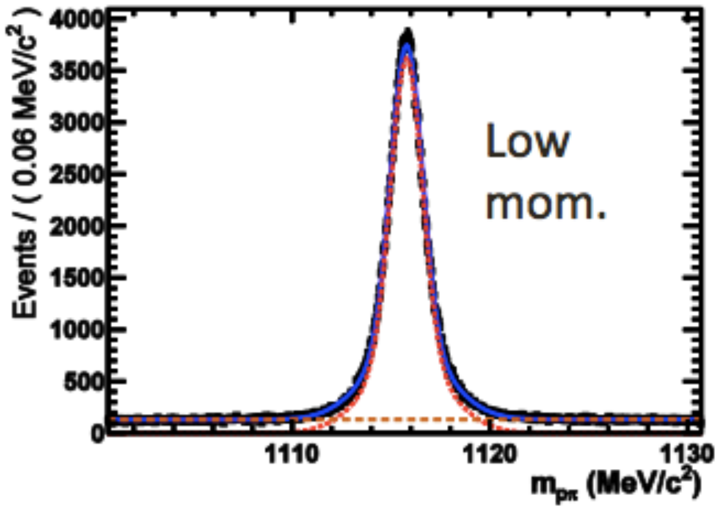
Assume that the response is factorised purely by 3 variables

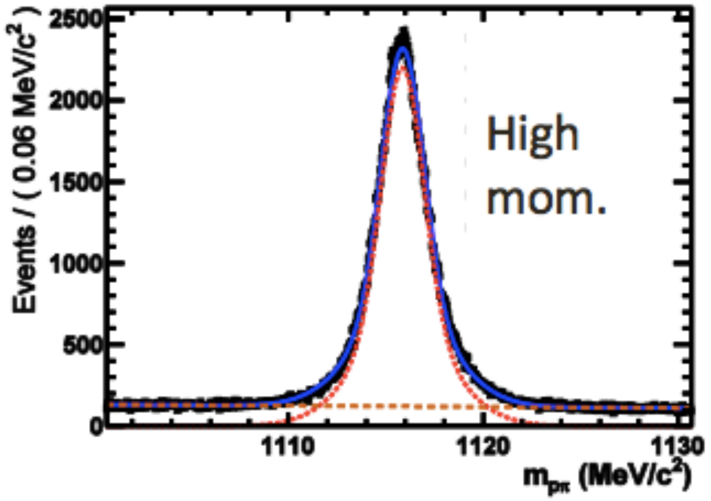
Other calibration samples --P

"Known" particle types:

$$\Lambda^0 \to \underline{p}^+\pi^-$$

- Calibration proton determined by reconstruction.
- No PID requirement in the calibration stripping line for this decay.
- Ks→ππ vetoed.
- 2 stripping lines, one chooses hi-momentum protons with relaxed pre-scale.
- Background remains so sPlot method used to assign tracks a signal weight.





Additional Protons

Inclusive $\Lambda_c^+ \to pK^-\pi^+$

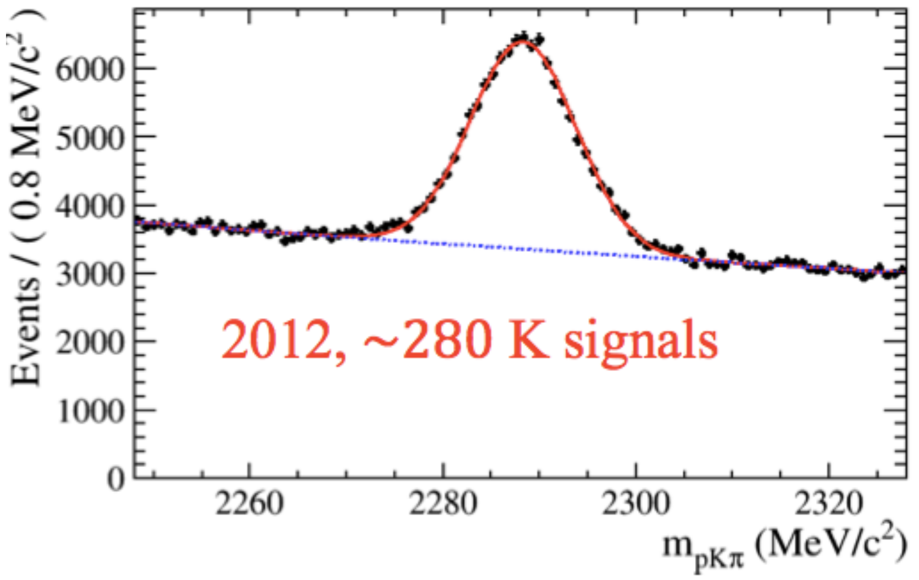
Covers low eta high-P region better than protons from V0

2 orders of magnitude lower statistics

Purity is not great

There for use if necessary. Systematic uncertainties will be higher

Best used to fill in the gaps where the standard sample is not sufficient.

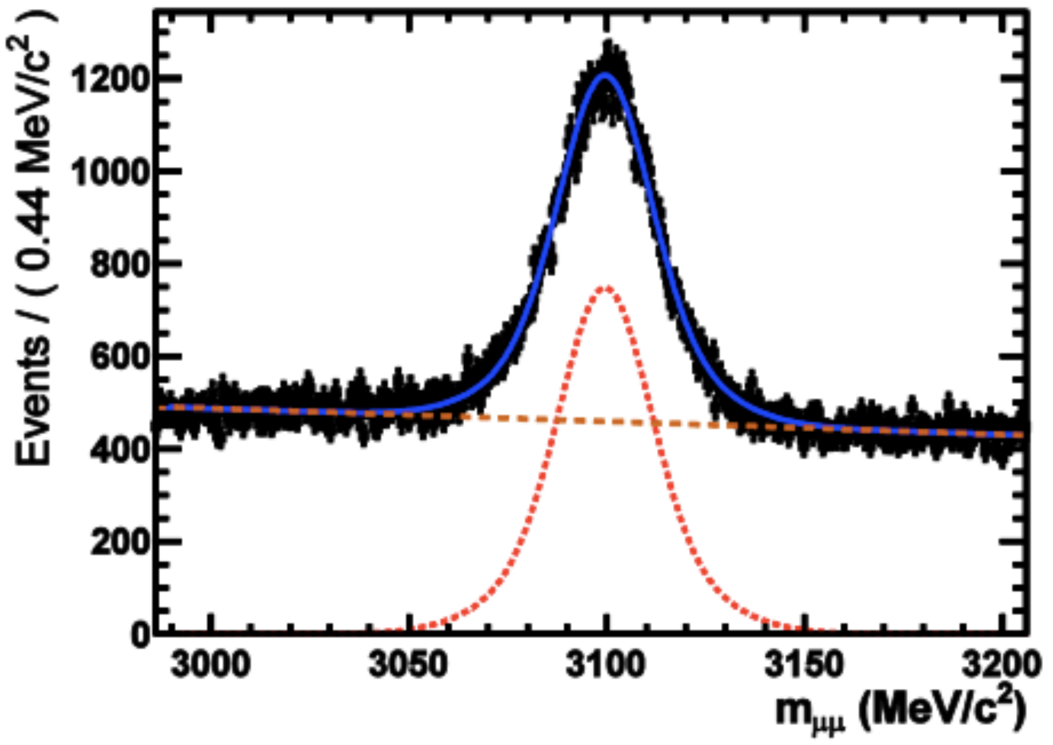


Muons

"Known" particle types:

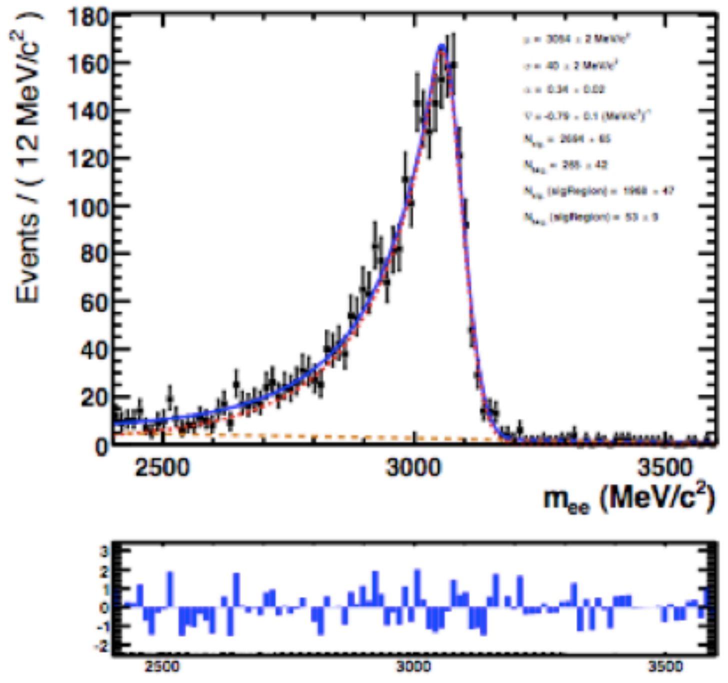
$B^+ \to J/\psi(\to \underline{\mu}^+\mu^-)K^+$

- Calibration proton determined by tag and probe method.
- Muon detector is in the trigger path and hence selected events required to be TIS wrt to these lines.
- Background remains so sPlot method used to assign tracks a signal weight.

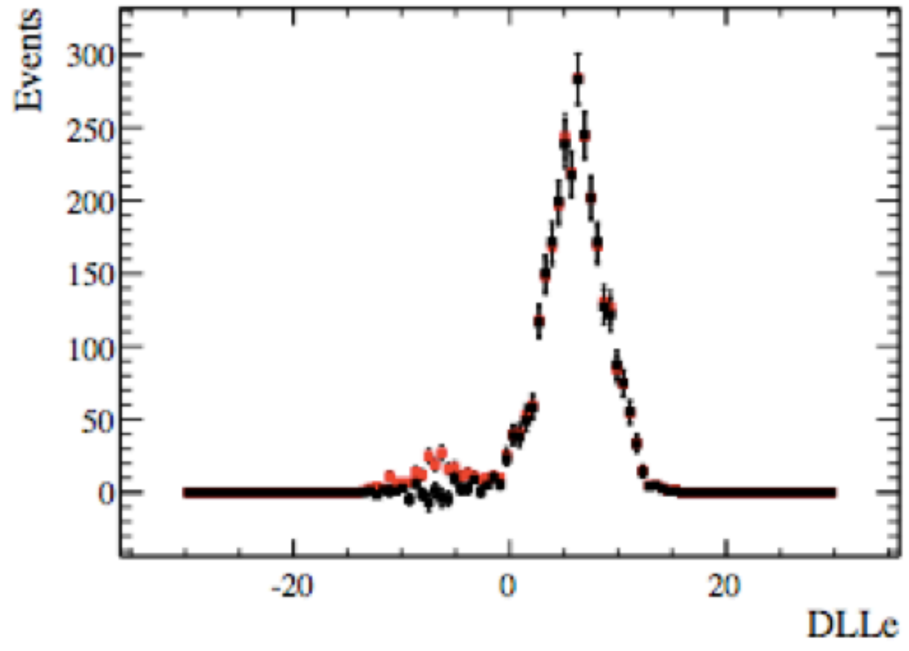


Electrons

Electrons are now available in PIDCalib



J/Ψ➔e$^+$e$^-$ tag and probe



e$^+$ MagUp 2011 Red: no sweights

Black: sweights

Calibration samples

Calibration samples available for all species of tracks

Can use to determine efficiency of identification, probability of mis-identification

Tracks are stored as RooDatasets

Divided into run number chunks

DataSet contains track, sweights, limited kinematic and event information, PID variables,

Use it in anyway you like

This directory for the standard scripts:

PIDPerfScripts/scripts/python/MultiTrack

This ones for making plots:
/PIDPerfScripts/scripts/python/Plots
/PIDPerfScripts/scripts/python/PubPlots

This one if you want to try and reweight your MC (for use in BDT)

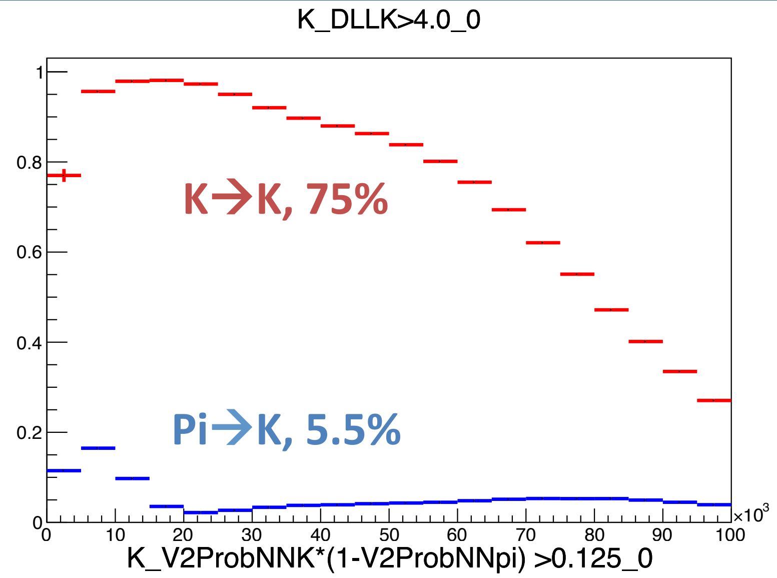
PIDPerfScripts/scripts/python/MCResampling

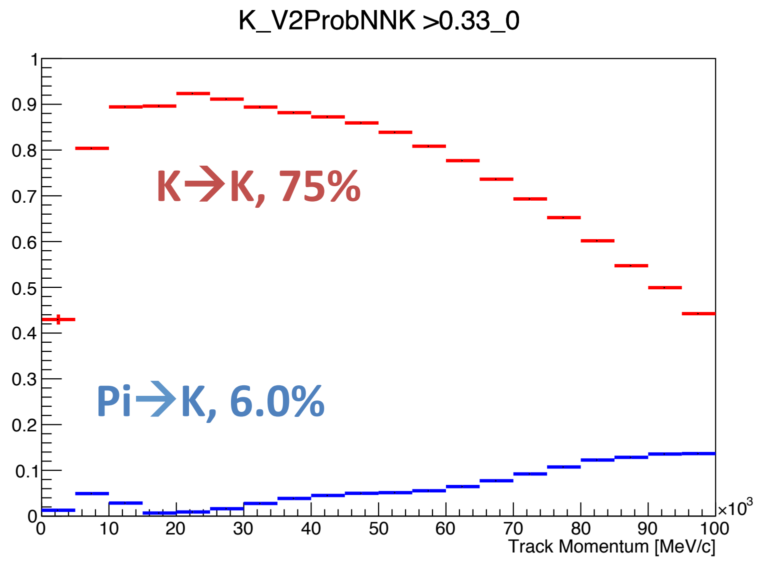
This one for examples on how to change the binning:
PIDCalib/PIDPerfScripts/scripts/python/Binning

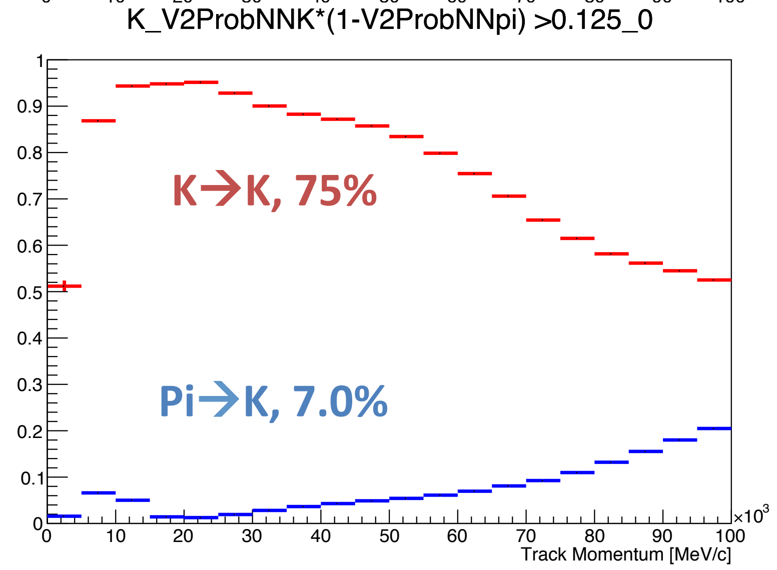
PIDCalib systematics

- Stat errors from calibration and reference sample. Calculated by PIDCalib scripts and are generally small
- Binning systematic – probes the assumption that the response is uniform over a reweighting bin – can be significant
- The one coming from the PIDCalib general procedure
  - Hard to quantify
  - Studies over last year suggest AT LEAST 0.1% for K/pi
    - This is absolute, not relative. So (90.2 +/- 0.1)) % , (0.3 +/- 0.1)%
    - Could be larger – if your analysis is senstive to this PIDCalib may not be the correct tool
  - Likely to be larger for the samples with lower purity
- Uncertainties because your reference sample doesn't quite match your signal sample

PID performance (1)

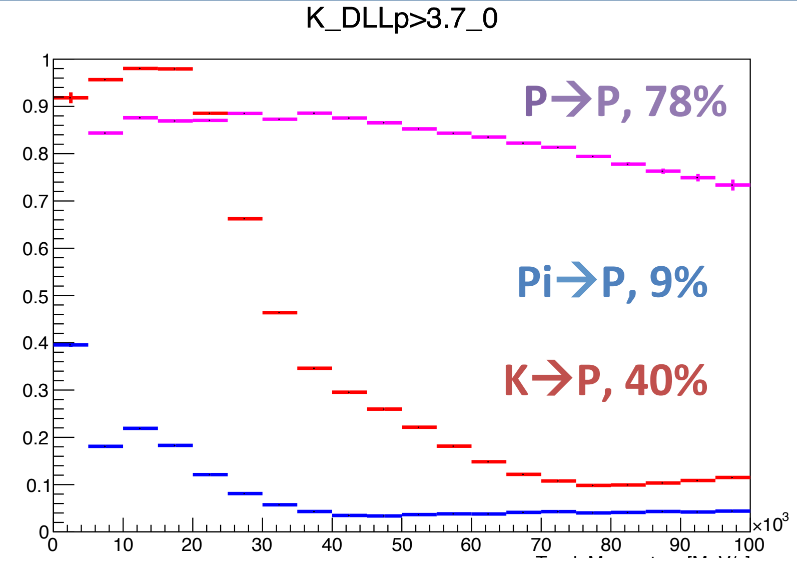


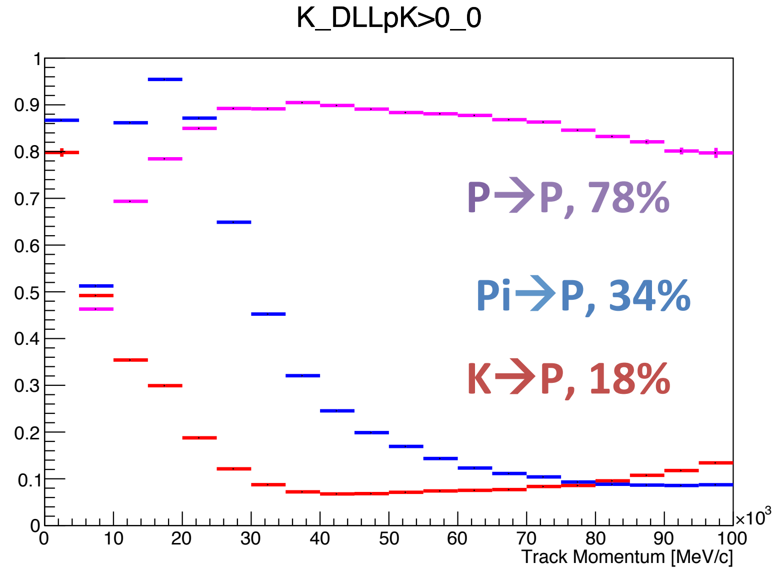


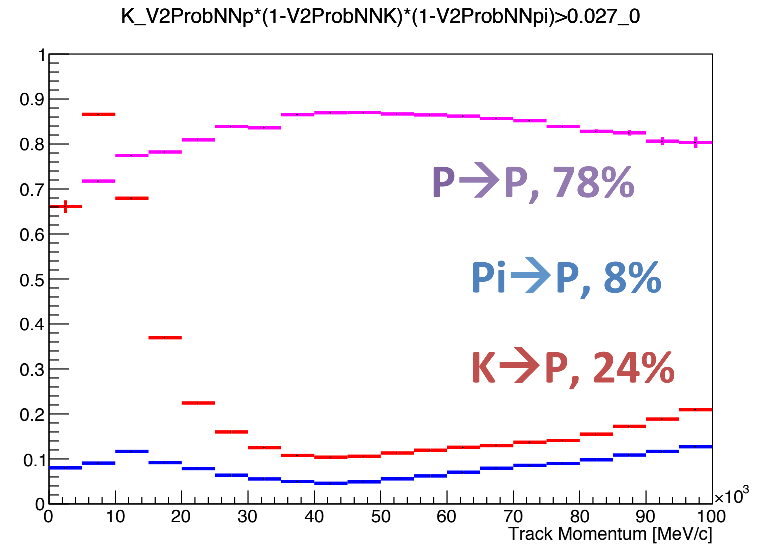


All choices of cut give the same overall Kaon ID
Overall Pion-misID varies
All the curves behave differently wrt to P, so clearly the final perfomance is very much signal mode dependent

PID Performance (2)





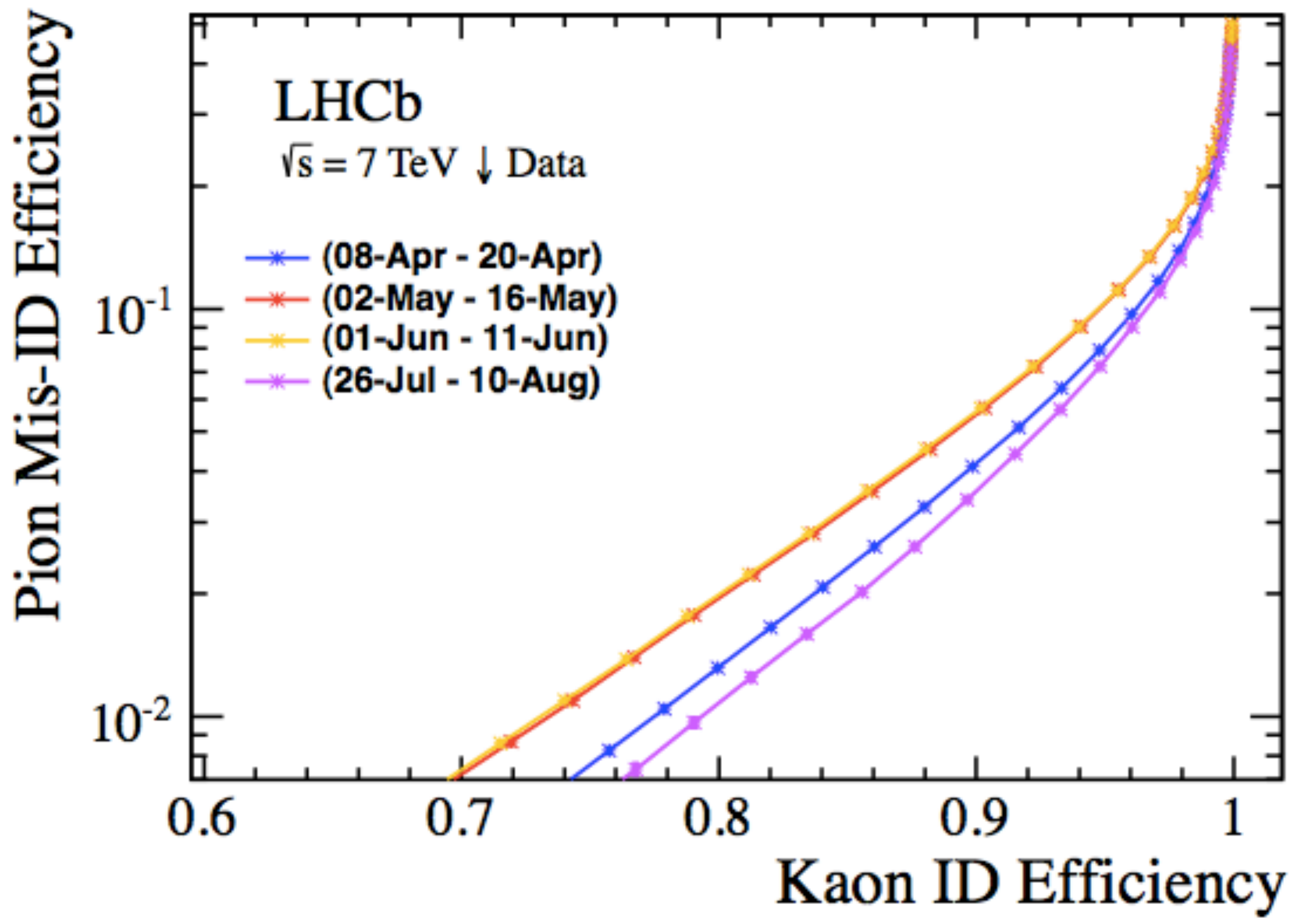


Large variation between the curves

DLL(P-Pi) rejects pions

DLL(P-K) rejects kaons

PID Performance (3)



Varies as a function of time

Important that you take that into account correctly in your own analysis.

For most cases PIDCalib does this by default

Conclusions

- Particle ID @ LHCb is one core strength
- It probably makes your analysis possible
- Its worth the pain of the data calibration

PIDCalib Tutorial

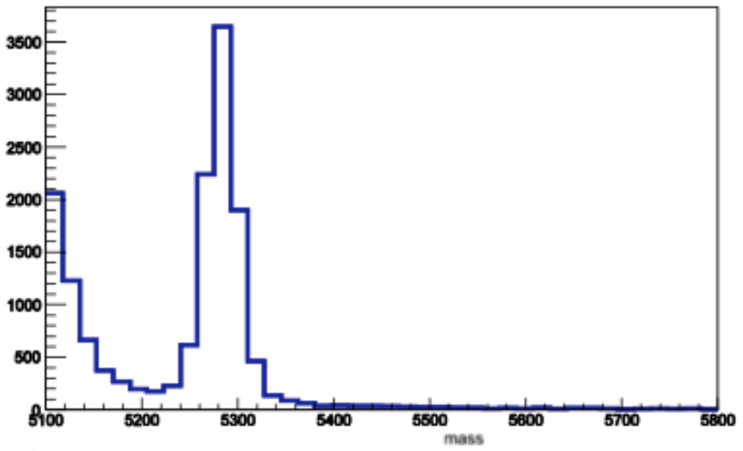
https://twiki.cern.ch/twiki/bin/view/LHCb/PIDCalibPackage

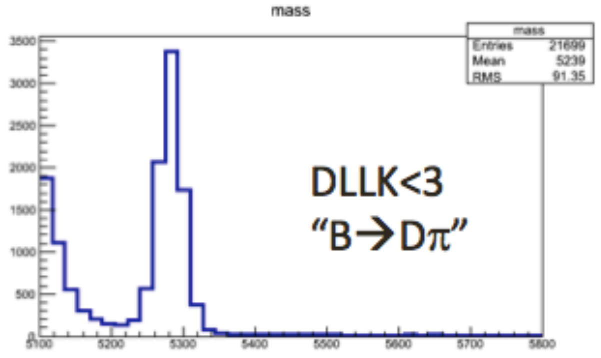
Plenty of instructions and information on the twiki

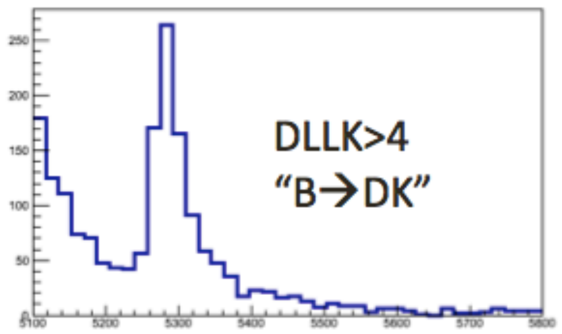
- I try and keep it up to date. Email the mailing list if it appears not to be
- Read the twiki page first if you have questions, the ones that are most frequently asked are there
- Follow the setting up instructions
- Write into your ANA note the version of PIDPerfScripts that you used

Tutorial problem

- I selected some B→Dh data in S20 with MagUp.
- I put some PID cuts on the bachelor.
- I say if DLL(K-pi)<3 then the event is "B→Dπ"
- If DLL(K-pi)>4, then the event is B→DK

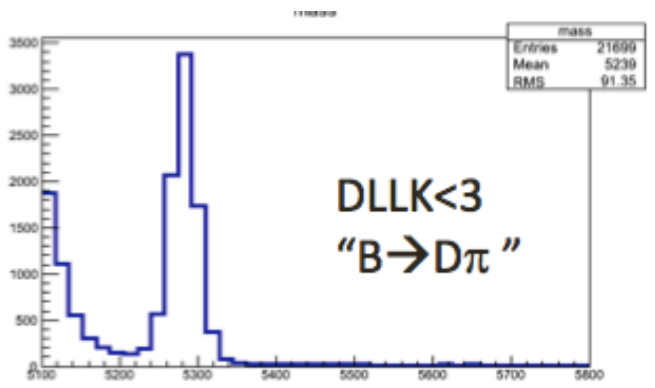


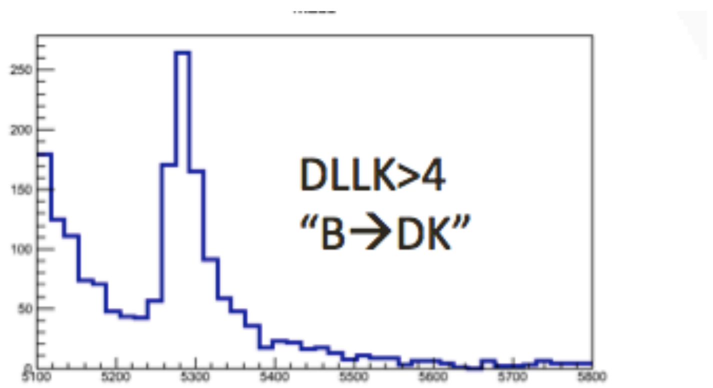




- Can clearly see the "B→DK" plot has some mis-ID (RH tail) How much?
- I want to work out the physics ratio of (DK)/(Dπ).
  - Assume all other selection cuts have same efficiency for each decay channel. But the PID cuts mean the signal efficiency isn't the same for each decay – must work it out.

Efficiencies required





- Assuming my bachelor track is a pion, what is the probability that that it passes DLLK < 3
- Assuming my bachelor track is a Kaon, what is the probability that it passes DLLK > 4
- Assuming my bachelor track is a pion, what is the probability that it will pass the cut DLLK > 4

These efficiencies depend on the kinematics of my bachelor track – hence no standard numbers for DLLK < 3 etc.
The answer will vary from one decay to another.

Base assumptions

We know the efficiency can vary as a function of track P & track η and the global Number of tracks in an event

We can divide the phasespace into 3-D binning in these variables.
We assume that the efficiency across this bin is constant and the same regardless of the track origin.

If I take the calibration sample tracks that fall in a particular bin I can look at how many pass a given cut.

$$\varepsilon_{cal(DLLK<3)\,bin(a)} = \frac{N^{pions\ with\ DLLK<3}}{N^{pions}} = \varepsilon_{sig(DLLK<3)\,bin(a)}$$

I can then assume that a signal track in my sample has the same efficiency as the calibration sample does in the same kinematic bin.

Getting Calibration efficiencies

Largely automated by running
PIDPerfScripts/scripts/python/MultiTrack/MakePerfHistsRunRange.py

- You select the stripping version, magnet polarity and particle type you are interested in. Optionally can restrict calibration to a certain run range.
- You pass the PID cuts you are interested in
- The (default) script will produce a 3-D histogram. Each bin corresponds to a region of kinematic phasespace. The value of the bin is the efficiency for passing the listed cut.

Pion efficiency for cut DLLK<3 and DLLK >4

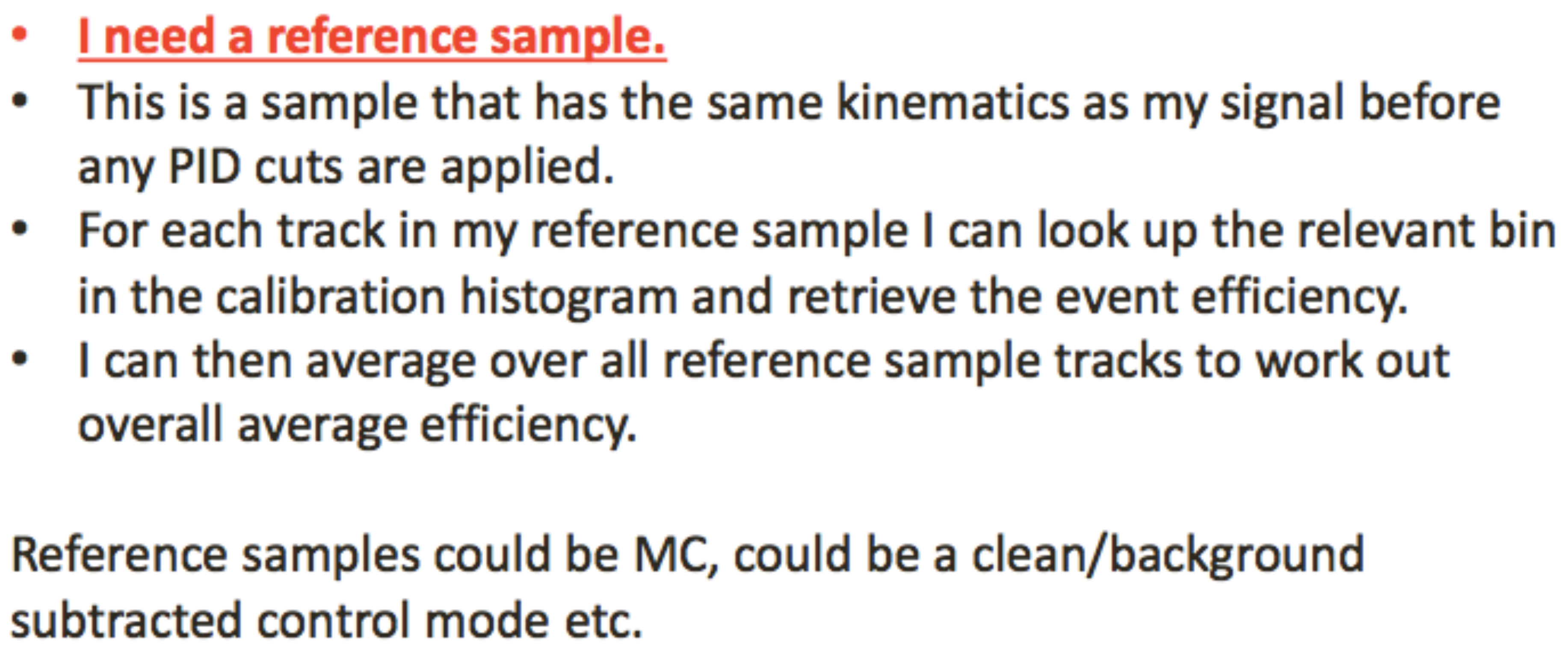
```
python MakePerfHistsRunRange.py “20” “MagUp” “Pi” “[DLLK < 3.0, DLLK > 4.0]”
```

Kaon efficiency for cut DLLK >4

```
python MakePerfHistsRunRange.py “20” “MagUp” “K” “[DLLK > 4.0]”
```

Output : PerfHists_K_Strip20_MagUp_P_ETA_nTrack.root,
PerfHists_Pi_Strip20_MagUp_P_ETA_nTrack.root

Reference sample

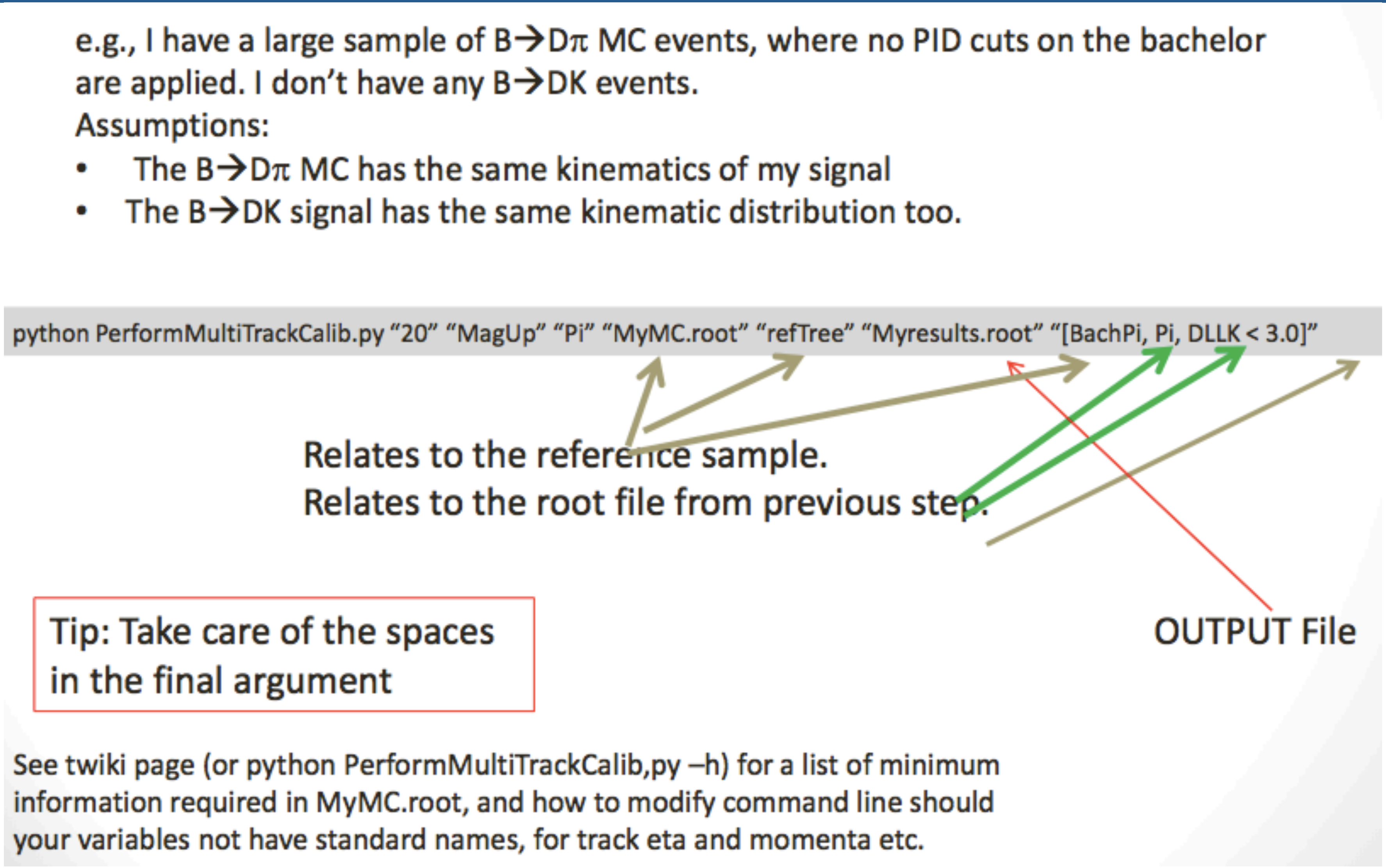


- **I need a reference sample.**
- This is a sample that has the same kinematics as my signal before any PID cuts are applied.
- For each track in my reference sample I can look up the relevant bin in the calibration histogram and retrieve the event efficiency.
- I can then average over all reference sample tracks to work out overall average efficiency.

Reference samples could be MC, could be a clean/background subtracted control mode etc.

But even my MC has PID cuts?
Not a problem, think about using double cuts to work out the efficiency from PIDCalib, or use the efficiency to reverse the effect of the cut. (of course you will pick up systematics due to the kinematics etc but life is not perfect)

Using the reference sample



e.g., I have a large sample of B→Dπ MC events, where no PID cuts on the bachelor are applied. I don't have any B→DK events.

Assumptions:

- The B→Dπ MC has the same kinematics of my signal
- The B→DK signal has the same kinematic distribution too.

```
python PerformMultiTrackCalib.py "20" "MagUp" "Pi" "MyMC.root" "refTree" "Myresults.root" "[BachPi, Pi, DLLK < 3.0]"
```

Relates to the reference sample.

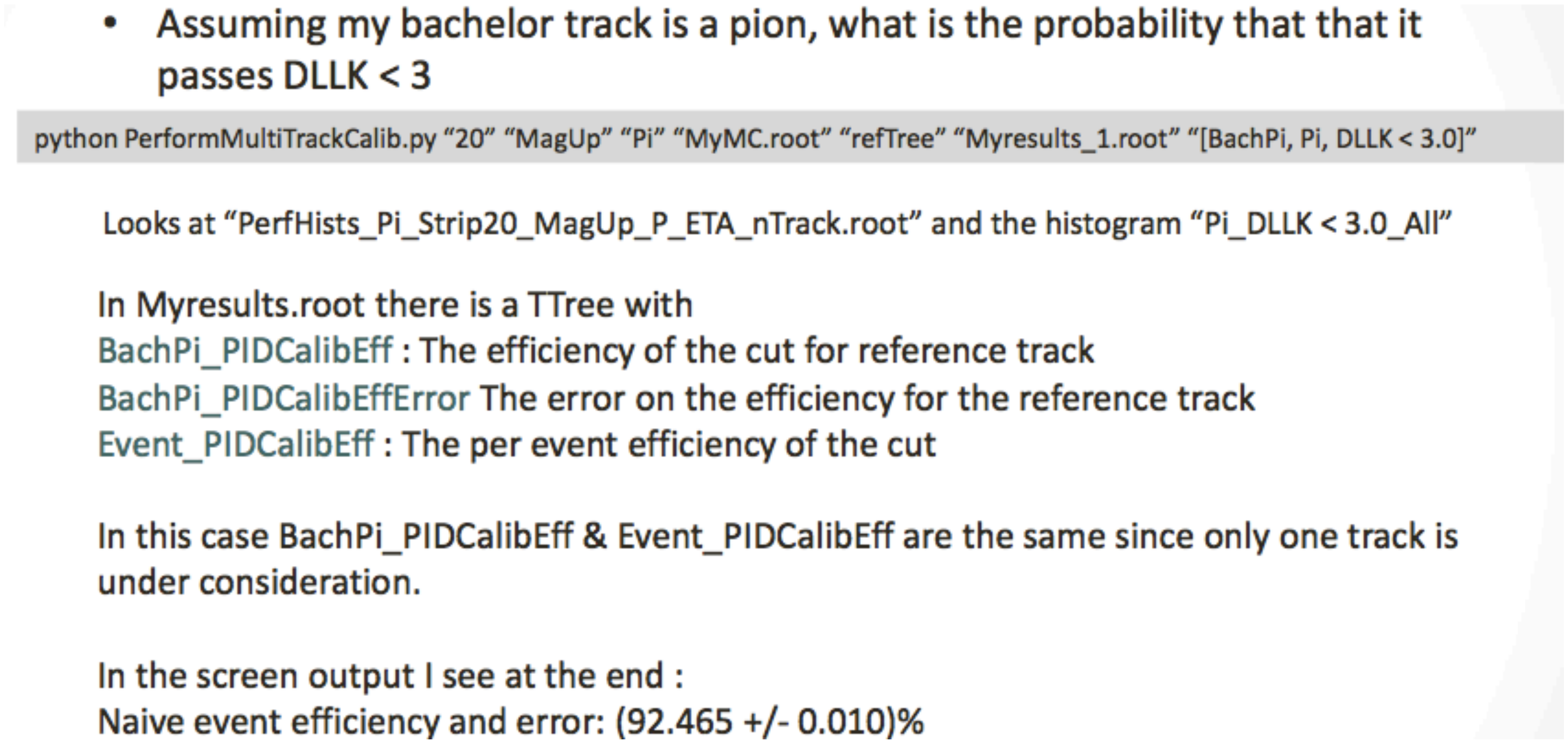
Relates to the root file from previous step.

OUTPUT File

Tip: Take care of the spaces in the final argument

See twiki page (or python PerformMultiTrackCalib,py –h) for a list of minimum information required in MyMC.root, and how to modify command line should your variables not have standard names, for track eta and momenta etc.

First req. eff



- Assuming my bachelor track is a pion, what is the probability that that it passes DLLK < 3

```
python PerformMultiTrackCalib.py “20” “MagUp” “Pi” “MyMC.root” “refTree” “Myresults_1.root” “[BachPi, Pi, DLLK < 3.0]”
```

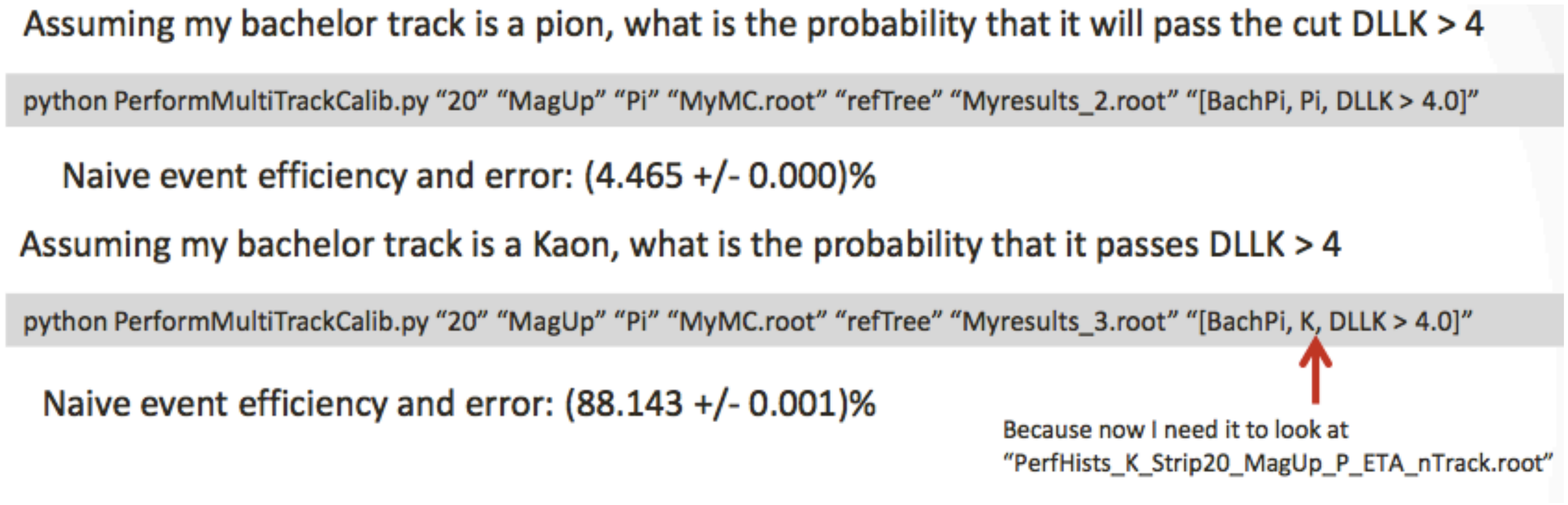
Looks at “PerfHists_Pi_Strip20_MagUp_P_ETA_nTrack.root” and the histogram “Pi_DLLK < 3.0_All”

In Myresults.root there is a TTree with
BachPi_PIDCalibEff : The efficiency of the cut for reference track
BachPi_PIDCalibEffError The error on the efficiency for the reference track
Event_PIDCalibEff : The per event efficiency of the cut

In this case BachPi_PIDCalibEff & Event_PIDCalibEff are the same since only one track is under consideration.

In the screen output I see at the end :
Naive event efficiency and error: (92.465 +/- 0.010)%

Rest of required information



Assuming my bachelor track is a pion, what is the probability that it will pass the cut DLLK > 4

```
python PerformMultiTrackCalib.py "20" "MagUp" "Pi" "MyMC.root" "refTree" "Myresults_2.root" "[BachPi, Pi, DLLK > 4.0]"
```

```
Naive event efficiency and error: (4.465 +/- 0.000)%
```

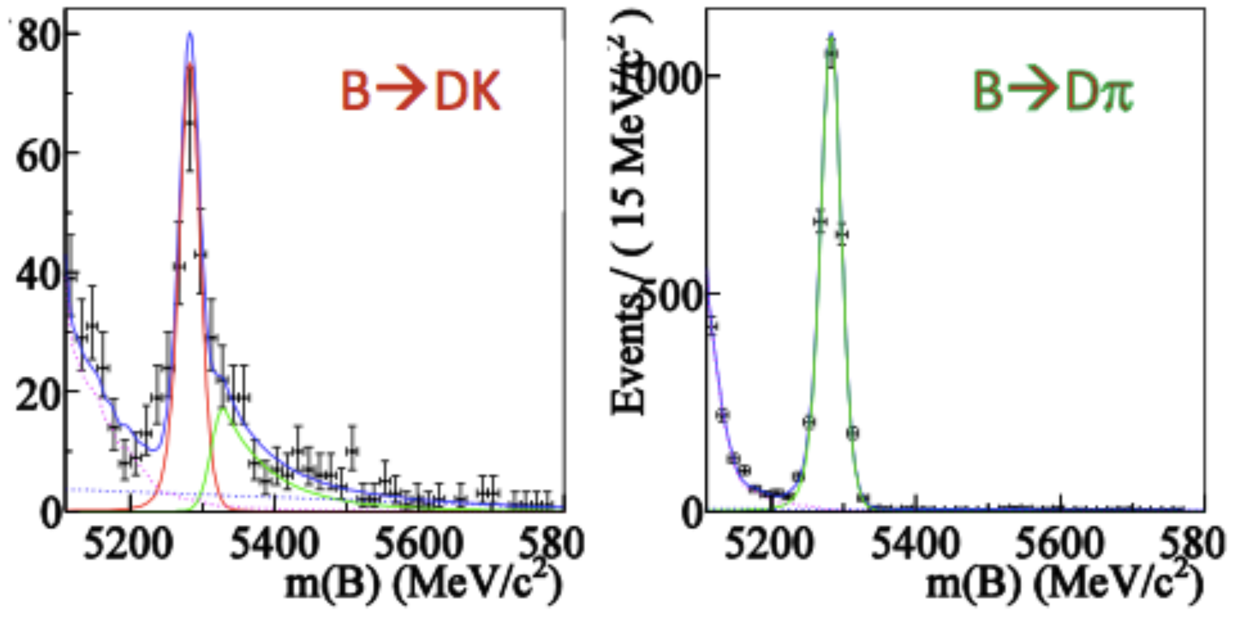
Assuming my bachelor track is a Kaon, what is the probability that it passes DLLK > 4

```
python PerformMultiTrackCalib.py "20" "MagUp" "Pi" "MyMC.root" "refTree" "Myresults_3.root" "[BachPi, K, DLLK > 4.0]"
```

```
Naive event efficiency and error: (88.143 +/- 0.001)%
```

Because now I need it to look at "PerfHists_K_Strip20_MagUp_P_ETA_nTrack.root"

Finish analysis



I can do a simultaneous fit to extract N(Dpi) and N(DK)

I can fix the mis-id yield in the LH plot to be N_{mis}=N(Dpi)*0.045/0.925

Physics ratio = $\frac{N(DK)}{0.881} \times \frac{0.925}{N(D\pi)}$

What is left

Systematic uncertainties

- Final standard script adds the uncertainty of the reference sample
- Then you need to add the method error, binning error and that coming from differences in the reference and signal sample

PIDCalib also makes a variety of performance plots (like those you just saw)

PIDCalib also can draw some of the calibration sample kinematics

These can be useful for estimating a good binning choice, estimating the effect of your cuts before running through the whole machinery etc etc..

Finally send questions to the mailing list:

lhcb-phys-pid-calibration@cern.ch