Table of Contents

PIDCalib for early measurements

News

This page gives information on how to use the forthcoming stripping 22 samples for the early measurements. The samples that are accessed by the code at the moment are for Magnet Down, data that was available at 8am on Monday 13/07 in bookkeeping. This cut off was chosen so that a sample could be provided to the Jpsi -xsection in time for the conference. These samples will eventually be replaced with the full EM datataking period. An announcement will be made when this happens.

Samples available

- Kaons and Pions from D*
- Protons from Lambdas
- Muons from J/psi

Protons from Lambda_c and electrons are not included. The statistics in this early data would be too low to be of use.

Archived samples

The samples first uploaded on 14th July 2015 can be found by altering the cmt/requirements file to point to this area: set CALIBDATASTORE eoslhcb.cern.ch//eos/lhcb/grid/prod/lhcb/calib/lhcb/calib/pid/CalibData/1stset-14-07-15/CalibData set MUONCALIBDATASTORE eoslhcb.cern.ch//eos/lhcb/grid/prod/lhcb/calib/lhcb/calib/pid/CalibData/1stset-14-07-15/CalibData

The samples used by the j/psi x section analysis that contained a few extra fills can be found by chaning the cmt/requirements file to set CALIBDATASTORE eoslhcb.cern.ch//eos/lhcb/grid/prod/lhcb/calib/lhcb/calib/pid/CalibData/JpsiXsecEMreq/CalibData set MUONCALIBDATASTORE eoslhcb.cern.ch//eos/lhcb/grid/prod/lhcb/calib/lhcb/calib/pid/CalibData/JpsiXsecEMreq/CalibData

variables

These are now computed offline with a slightly different algorithm to S21. The range on these variables now inlcudes -2 which is the default value for where the calculation fails. If you do not wish these tracks to be included in the efficiency calculation for your analysis please place the cuts accordingly

Setting up the code for use

Please use the normal setup of code as written on the main PID twiki page

Online and Offline

The significant difference between Run 1 and Run 2 is the heavy use of PID in the higher trigger algorithms, and the presence of the Turbostream that allows for analysis of data that is processed only online. The PID performance is expected to be different online and offline. Hence it is necessary to store both options.

- Analyses done using TurboStream should only access online quantities - that applies to kinematics as well as PID variables

- Analyses using full reconstruction where either no PID was used in the trigger or the efficiency of any such cut is irrelevant should use the offline calculated variables
- Analysis using a full reconstruction that do care about the efficiency of PID cuts placed by the trigger need to use the online variables for the trigger cuts and the offline variables for any further selection placed offline.

Naming convention

To remain consistent with Reco 14 calibration samples the offline variables will not have a new name. They will simply be P, PT, ETA, DLLK, V2ProbNNp etc. The online variables will be have a prefix of "Tesla_". So far the only Tesla variables stored are for the kinematics and the DLL variables. Full list of variable names is below.

Variables stored in the calibration datasets

These are the varibles which can: be cut on before the dataset is made, have their efficiency measured by `MakePerfHistsRunRange.py` or be a variable which is binned in. Not all variables feature in all datasets so you may observe some warnings. Please contact the PID mailing list if you think something is missing that should be there.

Var name	Description
`Charge`	+ve/-ve charged track (see this section)
`trackcharge`	a more user friendly version of Charge
`nTracks`	Total number of tracks in Best container
`runNumber`	runNumber of event
`DLLK`	Combined $\Delta \log \mathcal{L}_{K-\pi}$ as determined by Brunel
`DLLp`	Combined $\Delta \log \mathcal{L}_{p-\pi}$as determined by Brunel
`DLLe`	Combined $\Delta \log \mathcal{L}_{e-\pi}$as determined by Brunel
`DLLmu`	Combined $\Delta \log \mathcal{L}_{\mu-\pi}$as determined by Brunel
`Tesla_DLLK`	Combined $\Delta \log \mathcal{L}_{K-\pi}$ as available online
`Tesla_DLLp`	Combined $\Delta \log \mathcal{L}_{p-\pi}$ as available online
`Tesla_DLLe`	Combined $\Delta \log \mathcal{L}_{e-\pi}$as available online
`Tesla_DLLmu`	Combined $\Delta \log \mathcal{L}_{\mu-\pi}$as available online
`V2(3)ProbNNK`	Bayesian posteriori probability with either MCtuning v2 or v3: K
`V2(3)ProbNNpi`	Bayesian posteriori probabilitywith either MCtuning v2 or v3: π
`V2(3)ProbNNp`	Bayesian posteriori probability with either MCtuning v2 or v3: p
`V2(3)ProbNNmu`	Bayesian posteriori probability with either MCtuning v2 or v3: μ
`V2(3)ProbNNe`	Bayesian posteriori probability with either MCtuning v2 or v3: e
`V2(3)ProbNNGhost`	Bayesian posteriori probability with either MCtuning v2 or v3: Ghost
`P`	Track momentum as determined by Brunel(p) [MeV/c]
`PT`	Track transverse momentum as determined by Brunel (p_{T}) [MeV/c]
`ETA`	Track pseudo-rapidity as determined by Brunel (η)
`PHI`	Track phi angle as determined by Brunel (ϕ) [rad]
`Tesla_P`	Track momentum as available online (p) [MeV/c]
`Tesla_PT`	Track transverse momentum as available online (p_{T}) [MeV/c]
`Tesla_ETA`	Track pseudo-rapidity as available online (η)
`Tesla_PHI`	Track phi angle as available online (ϕ) [rad]
`IsMuon`	Track passes IsMuon requirement
`InMuonAcc`	Track passes InMuonAcc requirement
`IsMuonLoose`	Track passes IsMuonLoose requirement

`nShared`	Number of tracks with shared hits in the Muon stations
`RICHThreshold_pi`	Track had enough momentum to pass the RICH π thresholed
`RICHThreshold_p`	Track had enough momentum to pass the RICH p thresholed
`RICHThreshold_e`	Track had enough momentum to pass the RICH e thresholed
`RICHThreshold_K`	Track had enough momentum to pass the RICH K thresholed
`RICHAerogelUsed`	Track passed RICHAerogelUsed requirements
`RICH1GasUsed`	Track passed RICH1GasUsed requirements
`RICH2GasUsed`	Track passed RICH2GasUsed requirements
`HasRich`	Track passes HasRich requirement
`HasCalo`	Track passes HasCalo requirement
`CaloRegion`	Region where track hit the CALO (0=Unkown, 1=BeamPipeHole, 2=Inner, 3=Middle, 4=Outer)
`HasBremAdded`	Track momentum has had bremsstrahlung photons added to it
`nSPDHits`	Number of hits in the SPD detector
`TagTOS`	Tag muon track passes certain trigger requirements
`Unbias_HLT1`	Kaon or Pion track passes certain trigger requirements at L0 && HLT1
`Unbias_HLT12`	K/Pi/P track passes certain trigger requirements at L0 && HLT1&& HLT2

Making use of the standard user scripts.py

The functionality of the standard script remains as before. Below a few example command lines are given with a brief example of what they achieve. They assume that you are in the "MultiTrack" directory which can be navigated to by following the setup instructions above. or by doing

```
cd $PIDPERFSCRIPTSROOT/scripts/python/MultiTrack/
```

:

```
python MakePerfHistsRunRange.py 22 MagUp K "[DLLK >0.0"]
```

The minimum arguments are

The use of "22" as the first argument signifies that you wish to look at the calibration samples associated to stripping 22. This is the early measurement dataset. The output of this command will be a 3-D histogram in P, ETA, nTracks that gives the efficiency of a Kaon passing the cut DLLK > 0.0. The binning used will be the default binning which is likely to be unsuitable for charm early measurements. Please provide your own. See later section.

```
python MakePerfHistsRunRange.py -Z "" -c "trackcharge==1" 22 MagUp P "[DLLp >0.0"]
```

The use of -Z "" reduces the binning to 2-D by integrating over nTracks. The option -c "trackcharge==1" determines the efficiency for positive tracks only. This time protons are requests as determined by the use of "P"

```
python MakePerfHistsRunRange.py  -b ex_customBinning.py -s emu -c "IsMuon==1" 22 MagUp Mu "[DLLmu
```

The use of -b ex_custombinning.py -s optimisedbinning tell the script to use the binning scheme called "emu" and that the scheme can be found in the file ex_custombinning. A requirment is made that "isMuon" is 1 and the particle type is Muon. The binning scheme must be in P, ETA, nTracks. If it has been written in different variables then the names of these variables must be passed to the command line. e.g -Z "nSPDHits".

```
python MakePerfHistsRunRange.py 22 MagUp Pi "[DLLK < 2.0 && Tesla_DLLK < 4.0]"
```

This asks for the efficiency of the case where the online PIDK variable is less than 4 AND the offline (full reco) PIDK variable is less than 2. The default binning is used (offline variables).

```
python MakePerfHistsRunRange.py -b ex_customBinning.py -s DummyTesla -X "Tesla_P" -Y "Tesla_ETA"
```

This asks for the efficiency of the case where the online PIDK variable is less than 2. Assume this is a turbostream analysis. In this case it makes no sense to bin the variable according to the offline values of track P, eta etc and so the binning is also done in the variables available at the online stage. the scheme is called "DummyTesla" and can be found in ex_custombinning.py. It is only an example, it can't possibly be a good choice for actual use. However note that the following as to be added to the command line "-X "Tesla_P" -Y "Tesla_ETA" -Z "nSPDHits"" so that the binning takes place in these variables.

Making use of the standard user script:.py

Nothing changes from before. See here

Reminder on uncertainties

See here

Unphysical efficiencies?

The sPlot method does allow for unphysical efficiencies below 0 or above 1. In extreme cases it will result in a huge (positive or negative) value. This can happen when very few calibration tracks are present in a given bin. In this case the analyst should alter the binning scheme or exclude those regions of phasespace from their analysis.

-- SnehaMalde - 2015-06-18

This topic: LHCb > LHCbPIDCalibForEarly
Topic revision: r10 - 2016-03-09 - SnehaMalde