AFB studies at 500 GeV

LCFI+ Flavour Tag Optimization

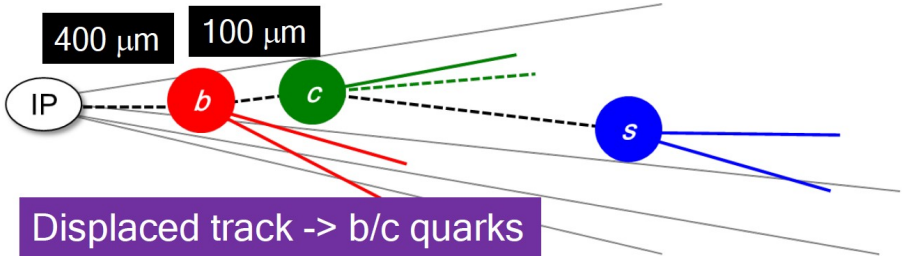
ILD Top/HF group meeting
2/12/22

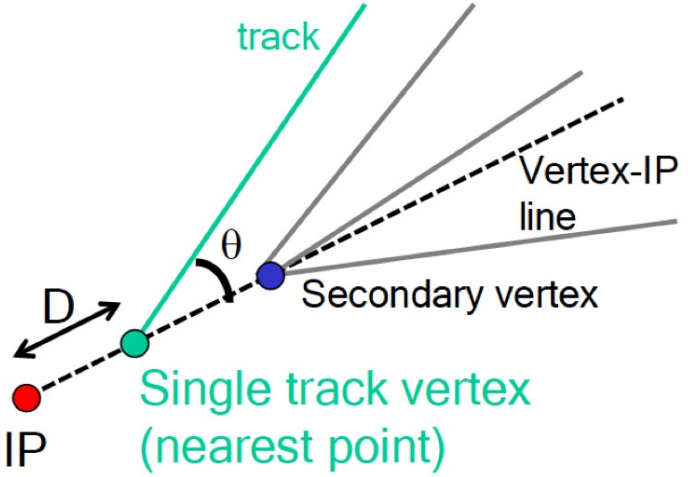
Jesús P. Márquez Hernández

LCFI+ (vertexing)

LCFI+ (Taikan Suehara, Tomohiko Tanabe; arXiv:1506.08371) :

- Vertex finder:
  - Reconstruct collinear or close-to-collinear vertexes by merging particle tracks from the event information.
  - Distance ($\tau_q \cdot c$) from the IP is key for b and c quark ID: Displaced vertexes.
  - We also encounter single track vertexes: pseudo-vertexes.
  - There are more details to select the tracks used for quark id.
    - e.g. V^0 rejection for neutral particles.

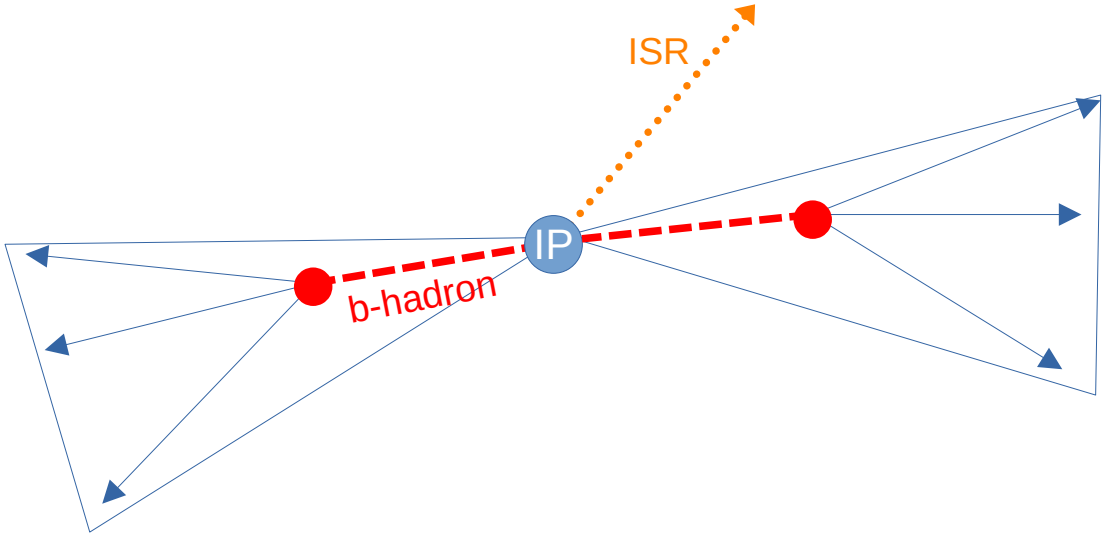




LCFI+ (jet clustering)

LCFI+ (Taikan Suehara, Tomohiko Tanabe; arXiv:1506.08371) :

- Jet Clustering:
  - Use the vertexing information in someways.
  - Different algorithms could be used (k_T, Durham, **VLC**, etc.).
  - In our case, we expect two back-to-back jets with ISR:



TMVA in LCFI+

- Most of the variables used in the TMVA are derived from d0, z0, the number of vertexes, the number of pseudo-vertexes and other kinematical variables like r-phi plane and transverse momenta.
  - e.g. b-quark probability in d0 values for all tracks
- This tagging in particular
  - The training is performed 4 times (A, B, C, D), with different selection of vertexing and single track pseudo-vertexing.

Category	A	B	C	D
Number of vertices	0	1	1	2
Number of single-track pseudovertices	0-2	0	1	0

Events for each category

Z-Pole (LCFI+ paper[1])

Events (%)			
Cat.	b jets	c jets	uds jets
A	22.9	59.5	98.1
B	39.7	39.8	1.80
C	13.5	0.54	0.02
D	23.8	0.19	0.04

1. LCFIPlus: A Framework for Jet Analysis in Linear Collider Studies

Without ISR removal

250 GeV samples

Events (%)			
Cat.	b jets	c jets	uds jets
A	20.8	50.8	98.0
B	28.4	46.6	1.59
C	21.7	1.98	0.13
D	29.1	0.59	0.25

500 GeV samples

Events (%)			
Cat.	b jets	c jets	uds jets
A	18.3	45.6	96.3
B	28.0	49.0	2.82
C	21.3	3.84	0.29
D	32.4	1.52	0.59

Category	A	B	C	D
Number of vertices	0	1	1	2
Number of single-track pseudovertices	0-2	0	1	0

Events for each category

With ISR removal

Z-Pole (LCFI+ paper[1])

Events (%)			
Cat.	b jets	c jets	uds jets
A	22.9	59.5	98.1
B	39.7	39.8	1.80
C	13.5	0.54	0.02
D	23.8	0.19	0.04

250 GeV samples

Events (%)			
Cat.	b jets	c jets	uds jets
A	13.9	46.2	98.2
B	30.5	51.0	1.59
C	23.9	2.29	0.11
D	31.7	0.55	0.14

500 GeV samples

Events (%)			
Cat.	b jets	c jets	uds jets
A	11.2	35.8	96.7
B	28.6	58.3	2.64
C	22.9	4.65	0.26
D	37.3	1.27	0.42

1. LCFIPlus: A Framework for Jet Analysis in Linear Collider Studies

Category	A	B	C	D
Number of vertices	0	1	1	2
Number of single-track pseudovertices	0-2	0	1	0

Particle Swarm Optimization (PSO)

- Iterative method to optimise a machine learning classification.
  - In our case, to optimise the multi-class BDT (Boosted Decision Tree) used for flavour tagging in LCFI+ (see back-up for more).

- The final goal is to obtain new weights for b-tagging and c-tagging.
  - These would have the best performance, while avoiding overtraining.

- We would do this for 250 and 500 GeV $q\bar{q}$ simulated events.

Boosted Decision Trees (BDT) - TMVA

- We are already working with these Gradient Boosted Decision trees using ROOT's Toolkit for MultiVariate data Analysis (TMVA). We use the following parameters:
  - BoostType=Grad.
  - NTrees.
  - Shrinkage.
  - UseBaggedBoost:BaggedSampleFraction.
    - Bagging: A new sampling is performed before each step (removes biases).
  - NCuts (binning used when sampling).
  - MaxDepth (Nº of leaves).

The Particle Swarm Algorithm optimizes the use of *these parameters*

We used all but the orange ones, which are method definitions

PSO - Overview

- Particle Swarm Optimization is a Gradient-free, bio-inspired, stochastic, population-based algorithm to optimize any kind of process towards a certain goal:
  - No maths involved in the optimization (no gradients or loss functions!).
  - It just keeps trying configurations and saves the best-performing one.
    - It mimics how animals look for resources, by trial and error.

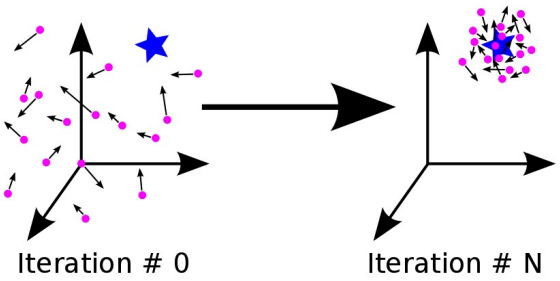


Image taken from a website

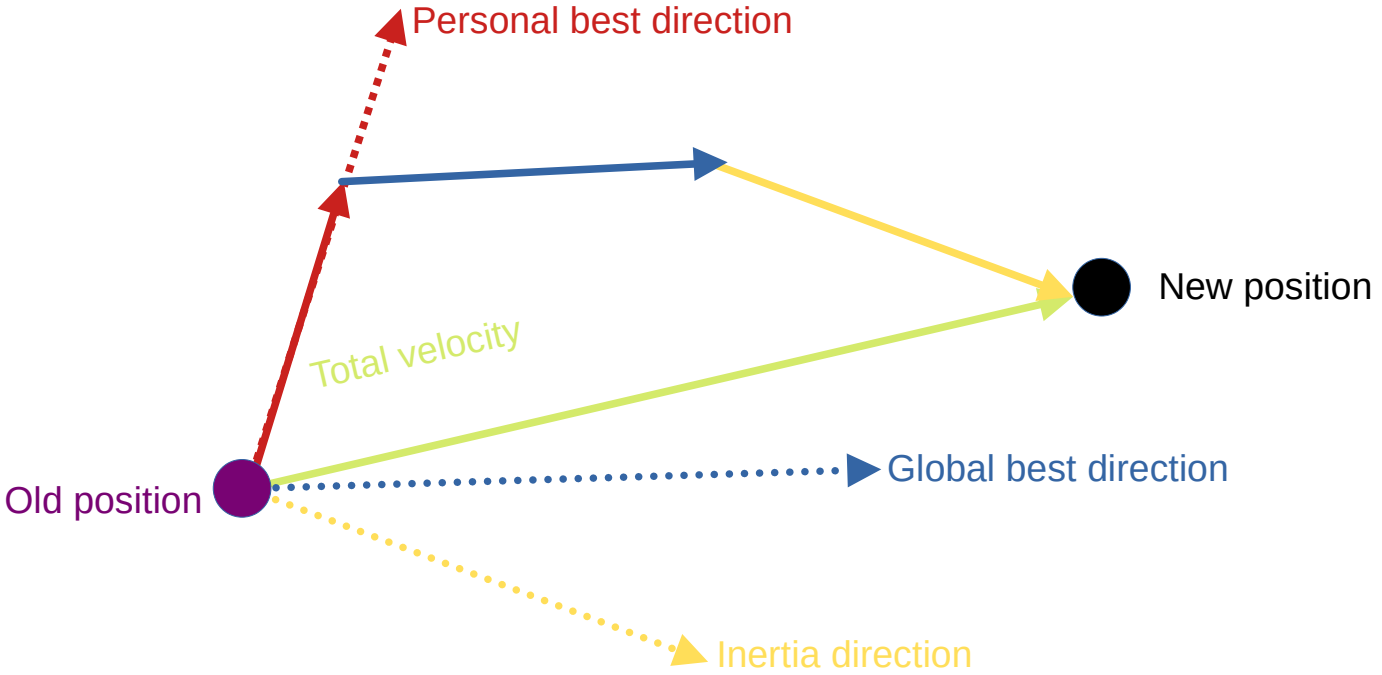
- How it works:
  - We have N “particles” (in our case: configurations of the BDT). Then:
    1) The BDT runs with the configuration of the particle.
    2) When finished, each particle gets a performance score.
       -We define a Function Of Merit (FOM) for this scoring
    3) We track each particle’s best configuration and the best global one.
    4) The particles moves to a new configuration (next slide).

For each iteration

PSO - Overview

Position: $\overrightarrow{X_i^{t+1}} = \overrightarrow{X_i^t} + \overrightarrow{V_i^{t+1}}$

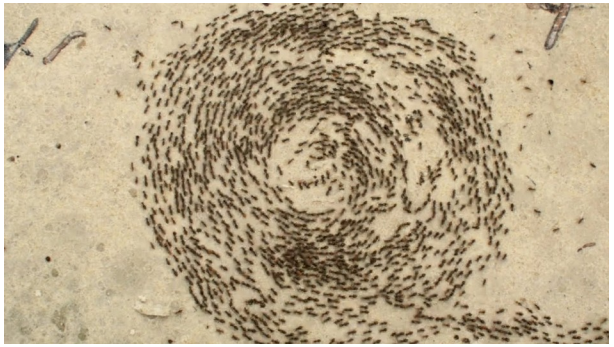
Velocity: $\overrightarrow{V_i^{t+1}} = w\overrightarrow{V_i^t} + c_1 r_1 \left(\overrightarrow{P_i^t} - \overrightarrow{X_i^t}\right) + c_2 r_2 \left(\overrightarrow{G^t} - \overrightarrow{X_i^t}\right)$



PSO – Adaptation to FT

- We need:
  - A 3-class classifier (b quarks, c quarks, uds quarks).
  - We also want to avoid overfitting:
    - Kolmogorov-Smirnov test
    - Anderson-Darling test

    Control biased test scores. (more info in back-up)
    Each of them have flaws, so using both is a safer way to go!
  - We need a FOM adapted to 3 different classes.
  - A final check is **always needed**:



Trial and error can go wrong sometimes!

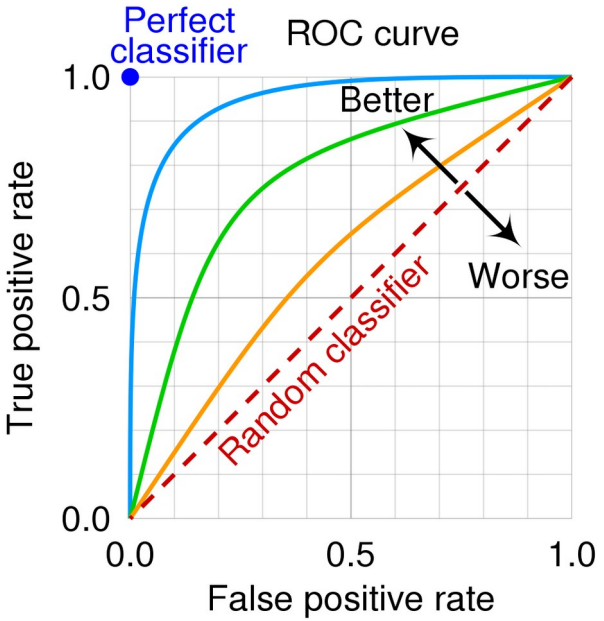
PSO - FOM

- The FOM being used is the averaged value of the Integral of the Receiver Operating Characteristic curve for each of the 3 data classes.
  - Considering the target class as signal and the others as background.

- The FOM is simply:

  $FOM=(AUC[b_{quark}] + AUC[c_{quark}] + AUC[uds_{quarks}]) / 3$

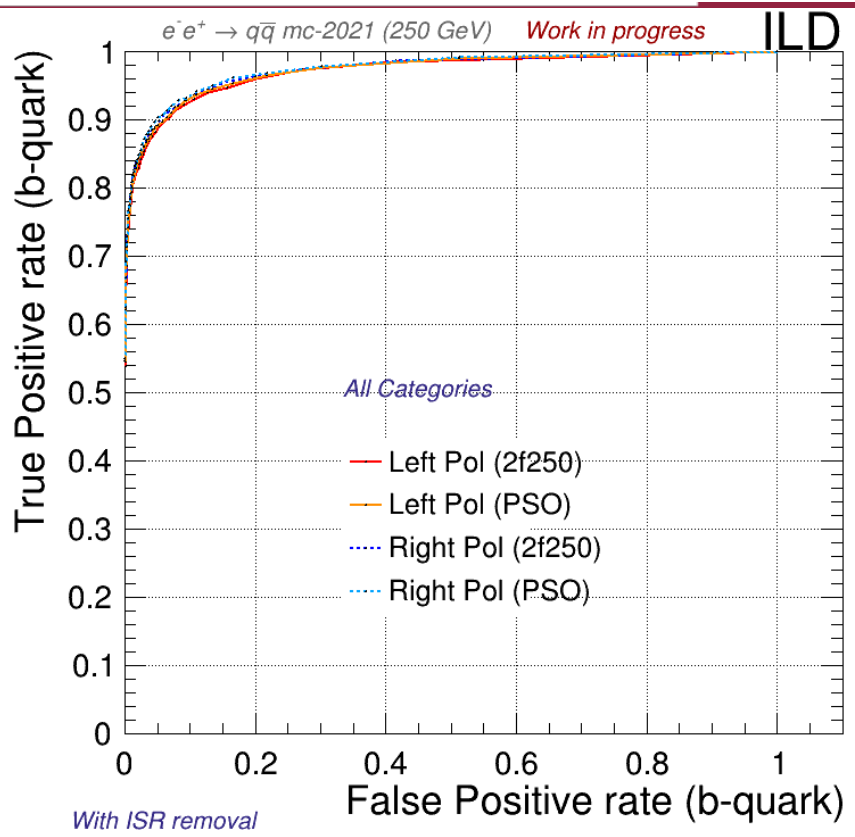
  , where AUC="Area Under Curve" (ROC Integral).

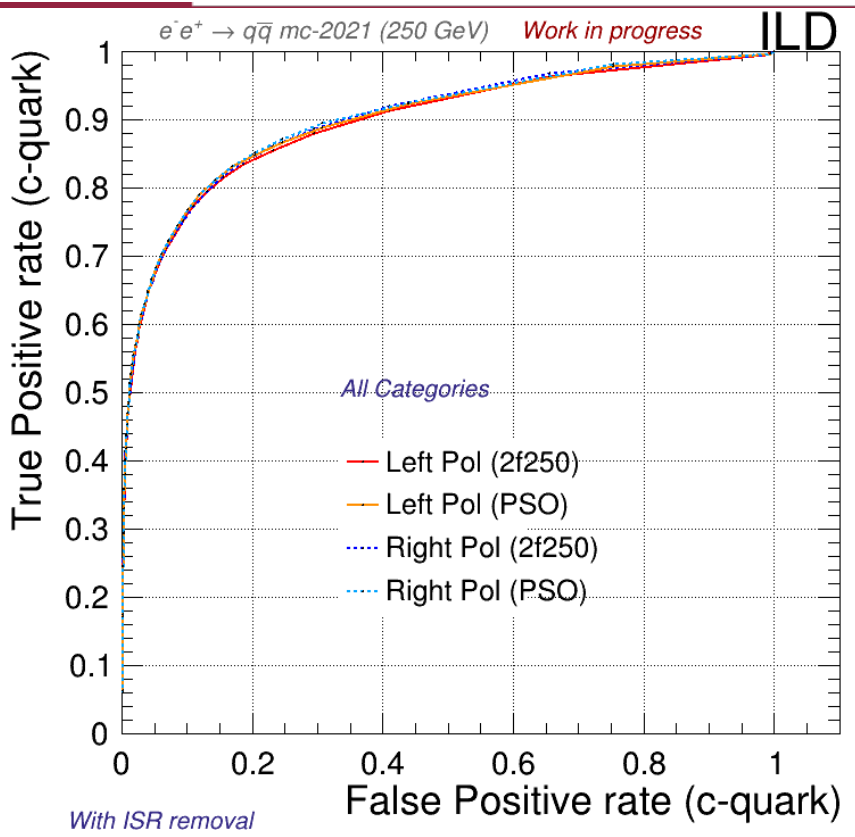


PSO – Performance plots

- On the next slides:
  - Plots for b-tag and c-tag:
    - ROC, considering the desired flavour as signal and the others as background.
      - Also AUC (ROC Integral) values to compare
    - Purity vs Efficiency.
  - All plots for 250 GeV & 500 GeV, and both polarization.
    - For 500 GeV we will also check the 4 categories in LCFI+.
      - We noticed a flaw in category b.

PSO Performance (250 GeV)

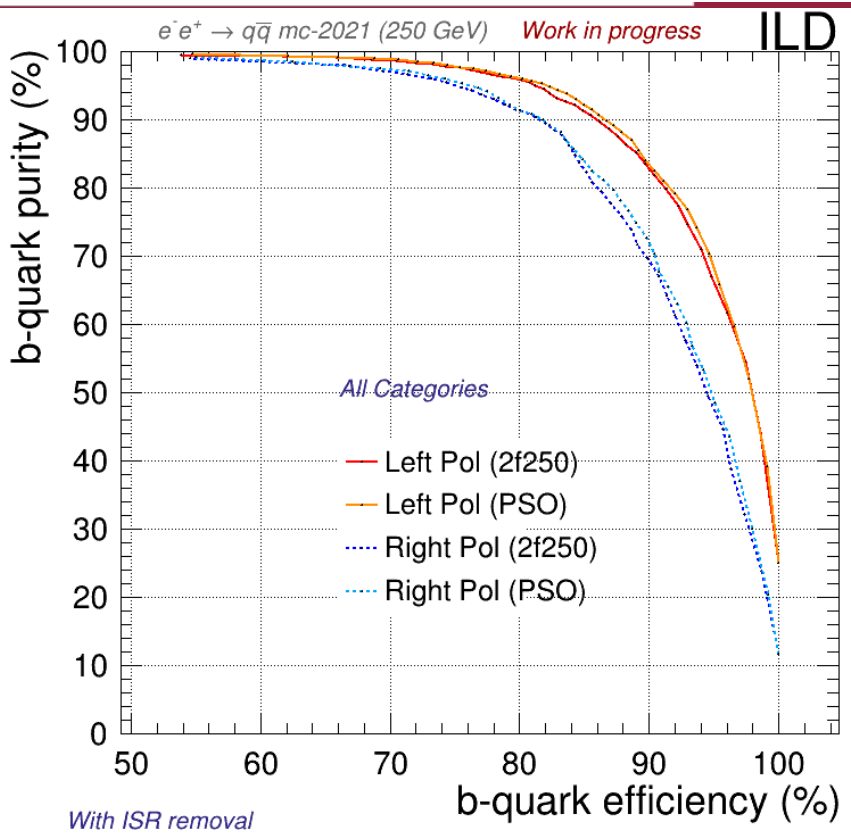


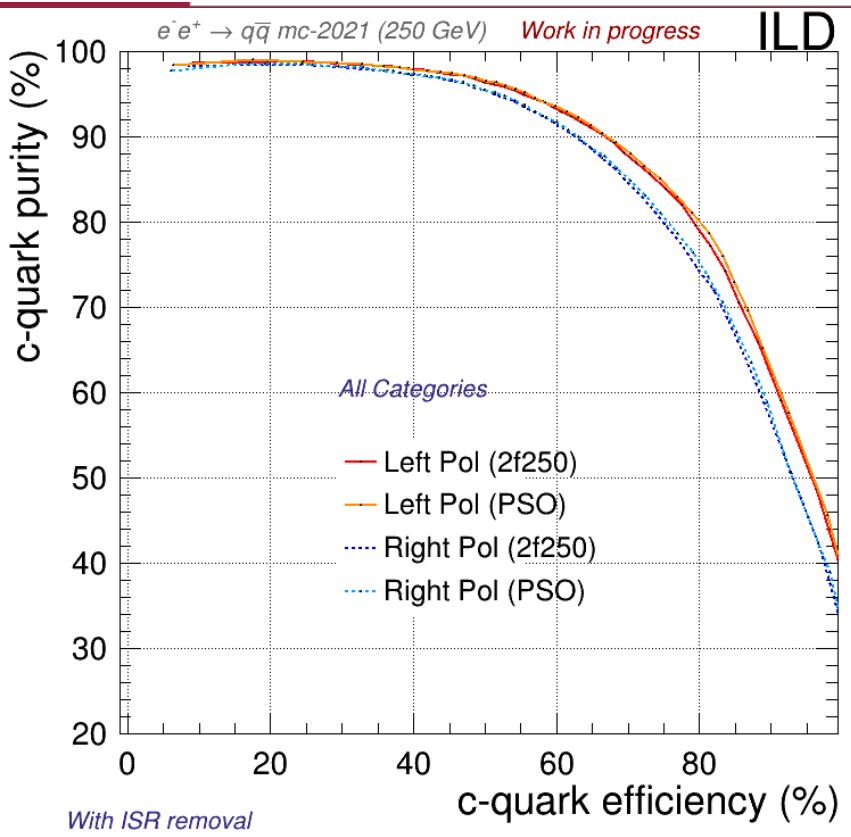


[Integrals 2f250] e^-_L: 0.971 | e^-_R: 0.973
[Integrals PSO] e^-_L: 0.973 | e^-_R: 0.976

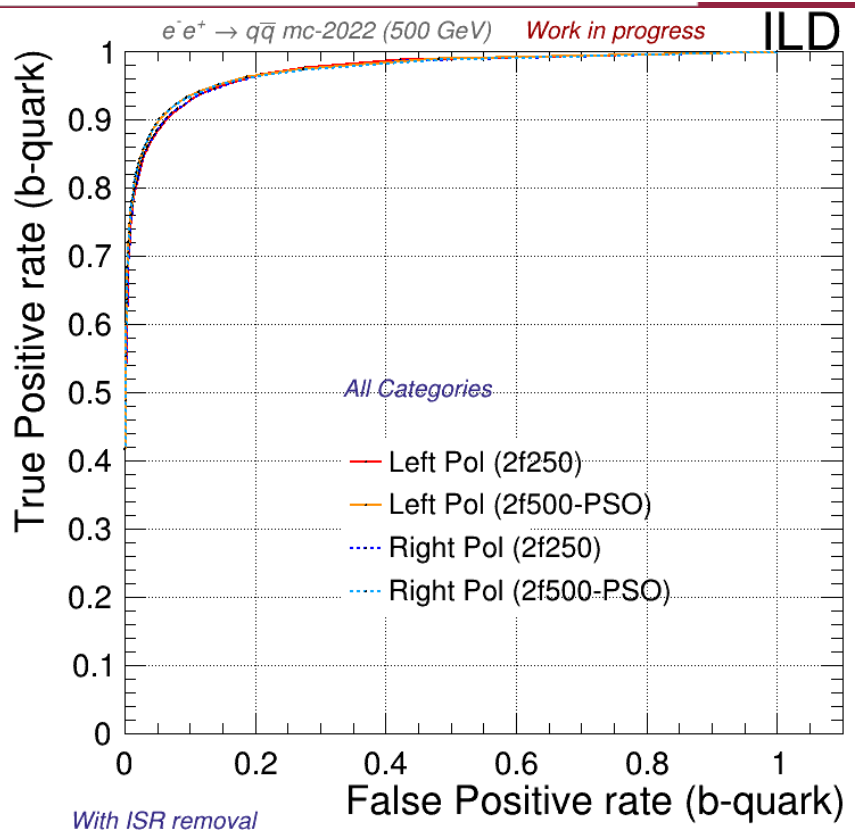
[Integrals 2f250] e^-_L: 0.898 | e^-_R: 0.902
[Integrals PSO] e^-_L: 0.901 | e^-_R: 0.904

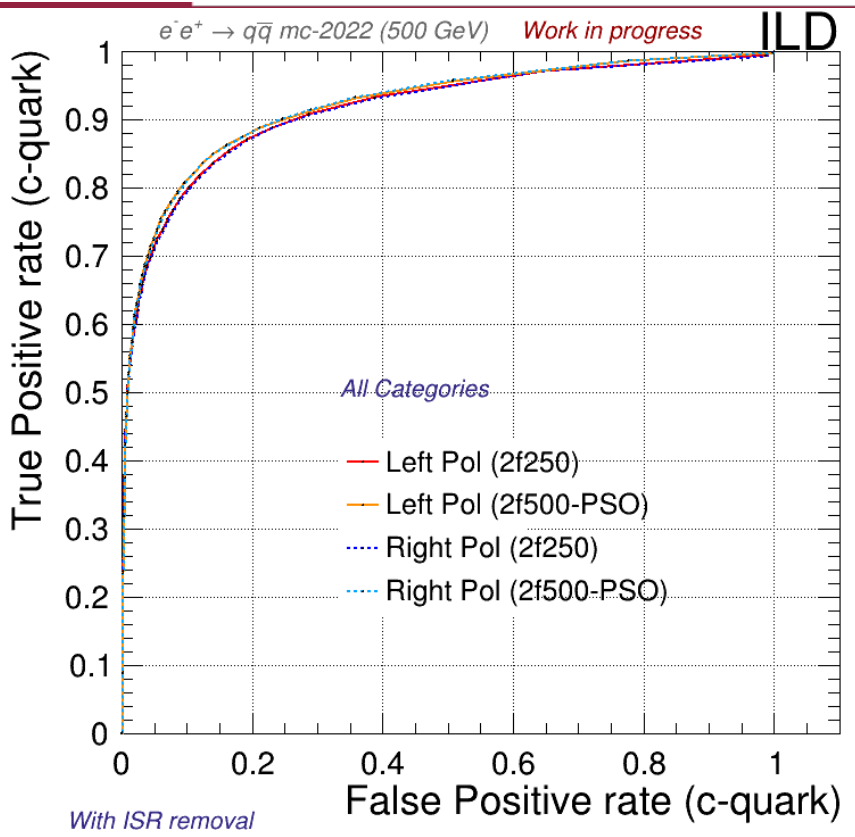
PSO Performance (250 GeV)





PSO Performance (500 GeV)

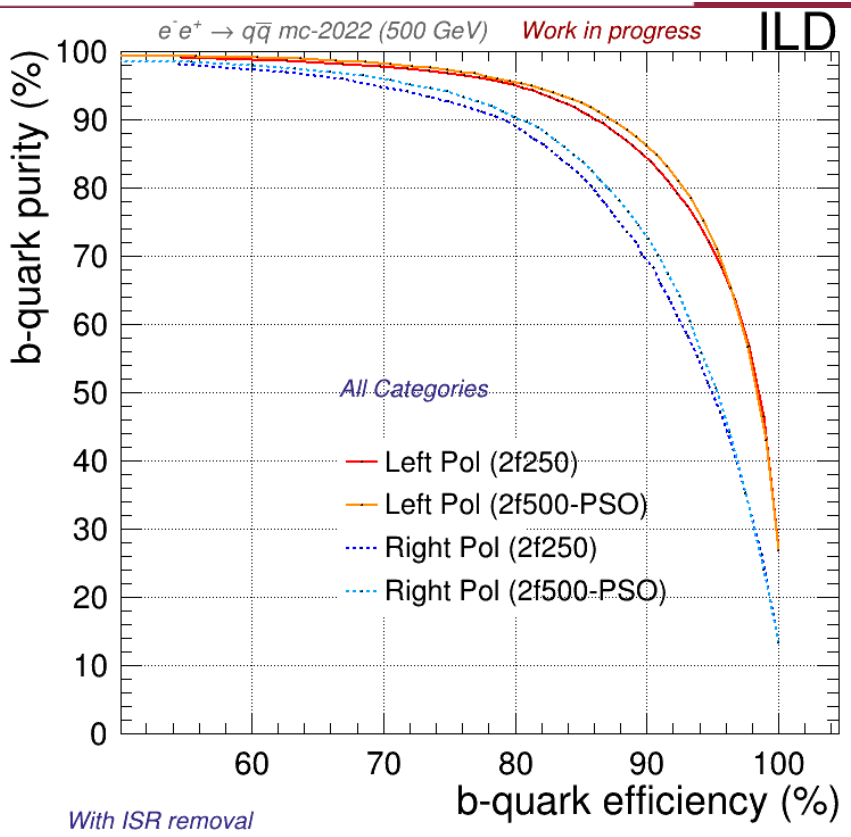


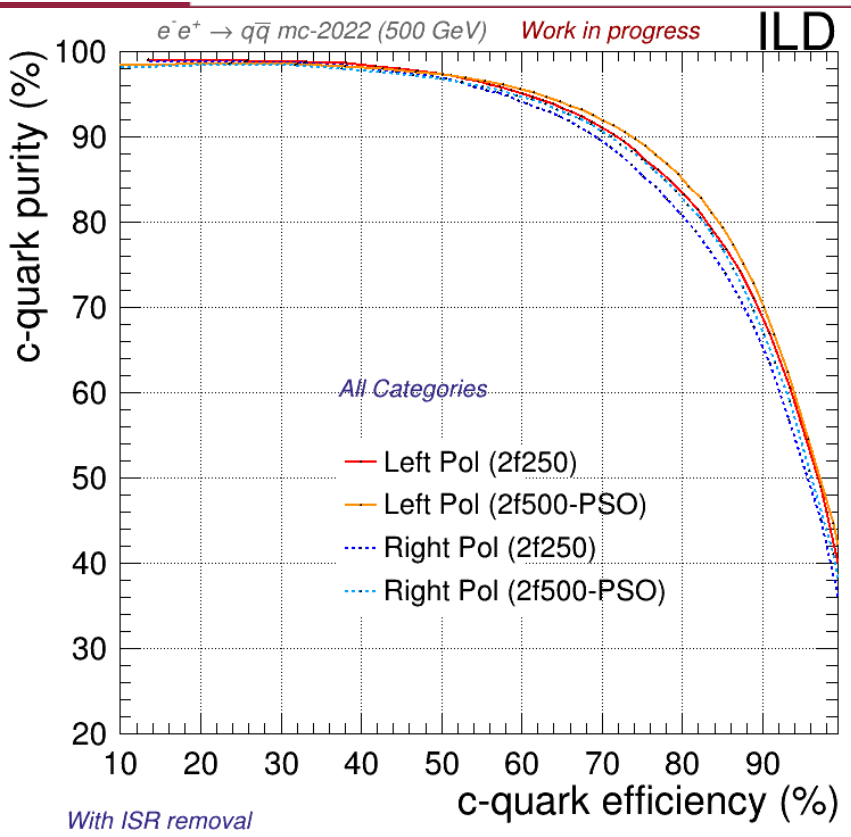


[Integrals 2f250] e^-_L: 0.972 | e^-_R: 0.971
[Integrals PSO] e^-_L: 0.974 | e^-_R: 0.973

[Integrals 2f250] e^-_L: 0.917 | e^-_R: 0.916
[Integrals PSO] e^-_L: 0.923 | e^-_R: 0.923

PSO Performance (500 GeV)





This results have been surpassed right now due to a bad optimization in category B. (more info in back-up)

PSO Performance - Observations

- We can notice:
  - A difference in efficiency for different polarization.
  - Category B is not well optimized.
- We can check this by:
  - Weight files specific to each polarization.
  - Re-optimize category B.

Polarization is *irrelevant*
Category B has been successfully re-optimized!

Currently doing both things at the same time (Advance in back-up)

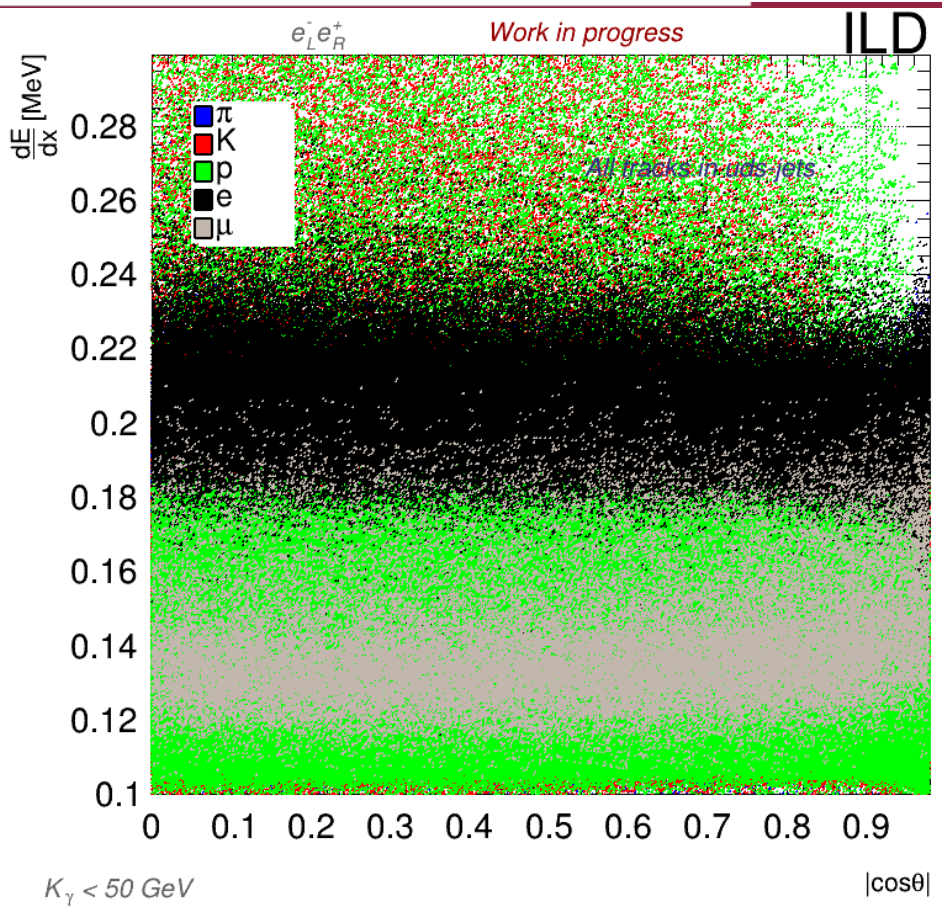
- Being done:
  - New plots with bigger and different samples.
    - Smoother and unbiased plots.
  - A check for getting the best b-tag & c-tag cuts in signal to get the best performance.
    - F1-score + purity restriction.
- We are going beyond this optimization by introducing new variables in LCFI+.
  - Working on observables with dEdx (next slides).

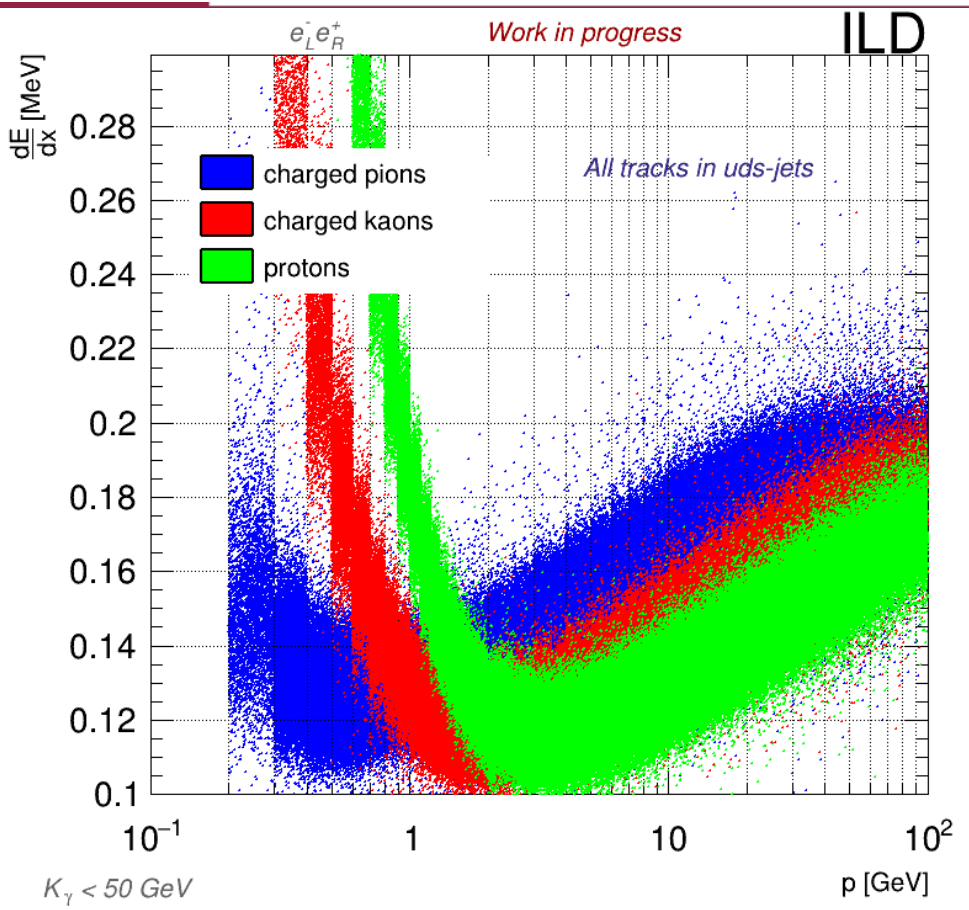
Using dEdx for flavour tagging

- We will study the distance of different tracks wrt Kaon dE/dx.
  - 1 pfo in a single track ⟶ 1 particle energy trace (Bethe-Bloch formula).
  - Our .lcio files already have an estimation of the distance between a given track and the estimated for a given particle (Kaon, pion, etc.), we selected the Kaon.
    - This distance is not always a good estimation, we have to preselect first a region in momenta in which this measurement is consistent.

- Once the distance is proved to be somehow useful to distinguish different particles:
  - Build new observables useful for flavour tagging!

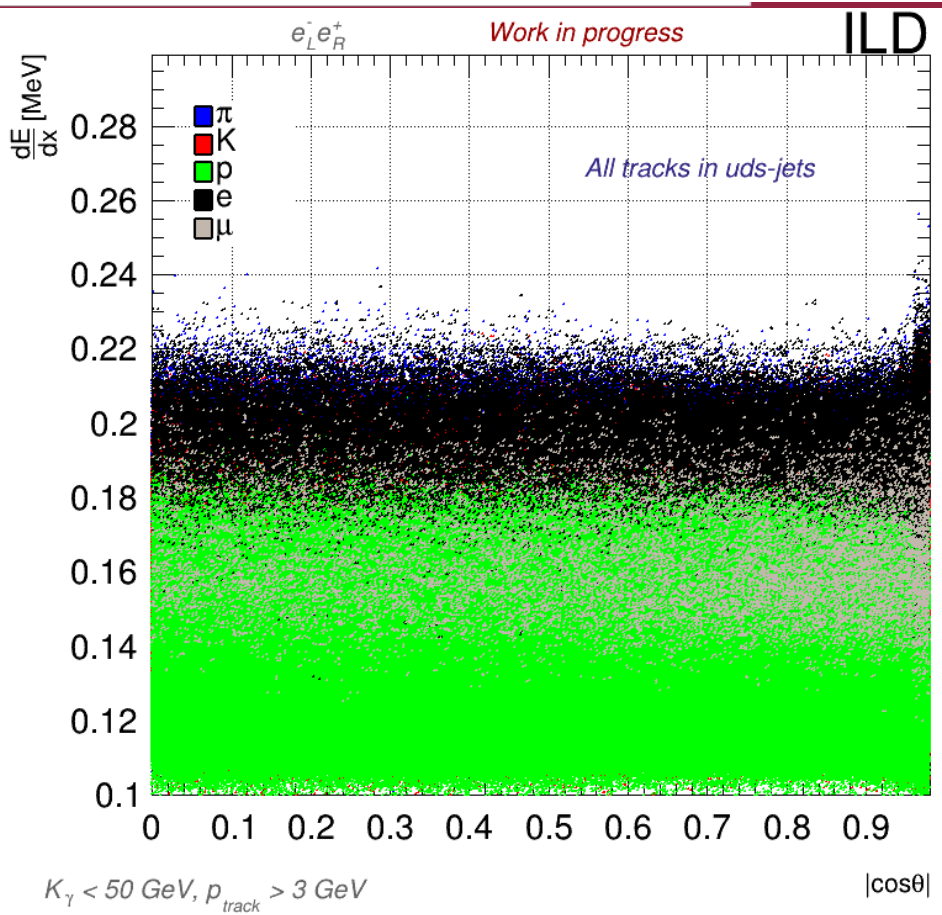
Using dEdx for flavour tagging

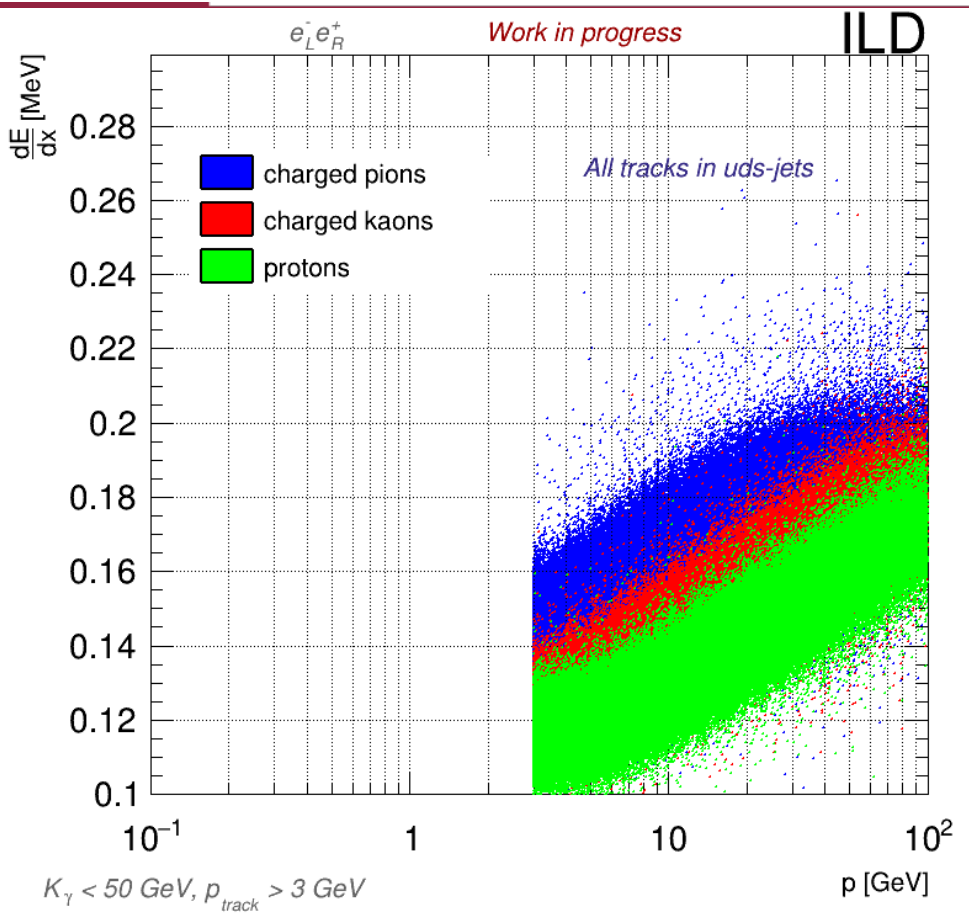




There's a high population at low momentum and below 3 GeV the distributions overlap!

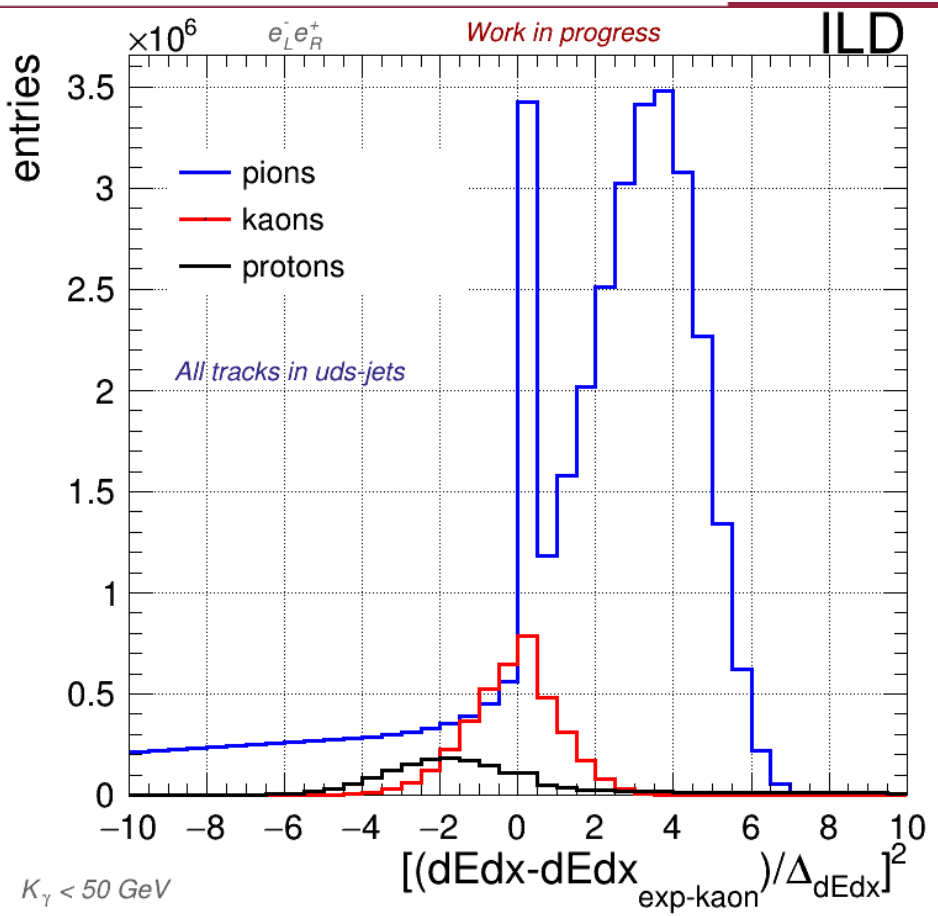
Using dEdx for flavour tagging

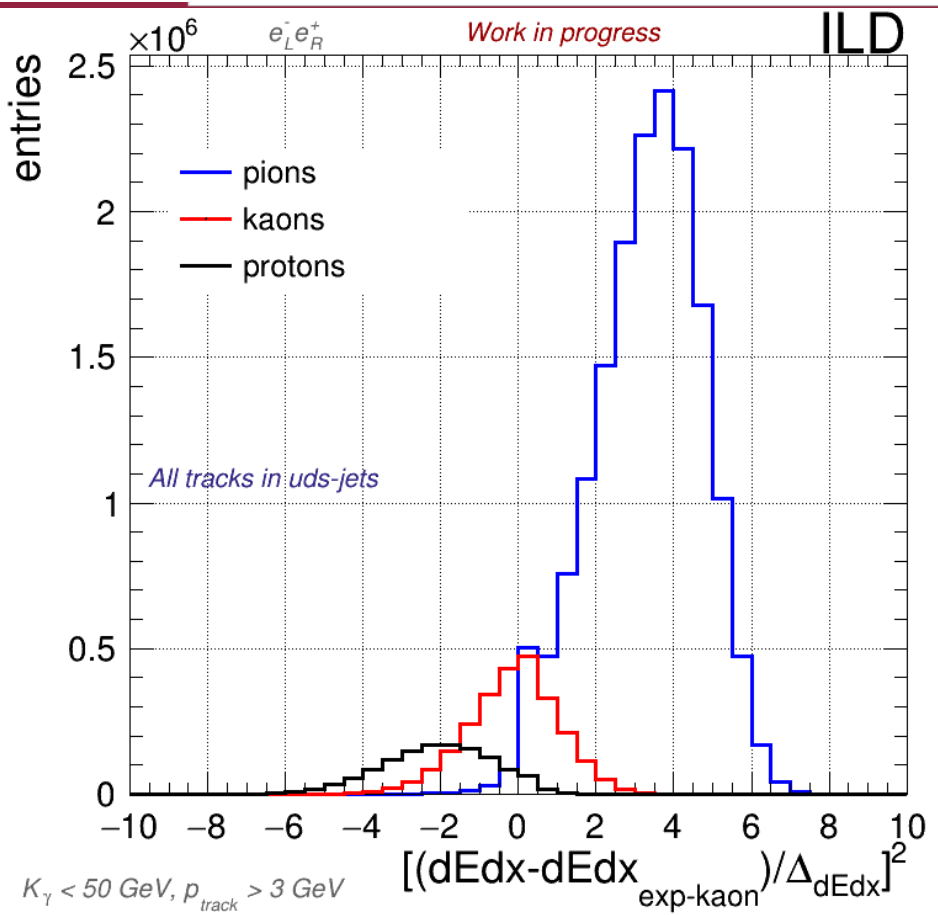




Effects of cutting the signals at 3 GeV. This behavior is similar to b and c jets.

Using dEdx for flavour tagging

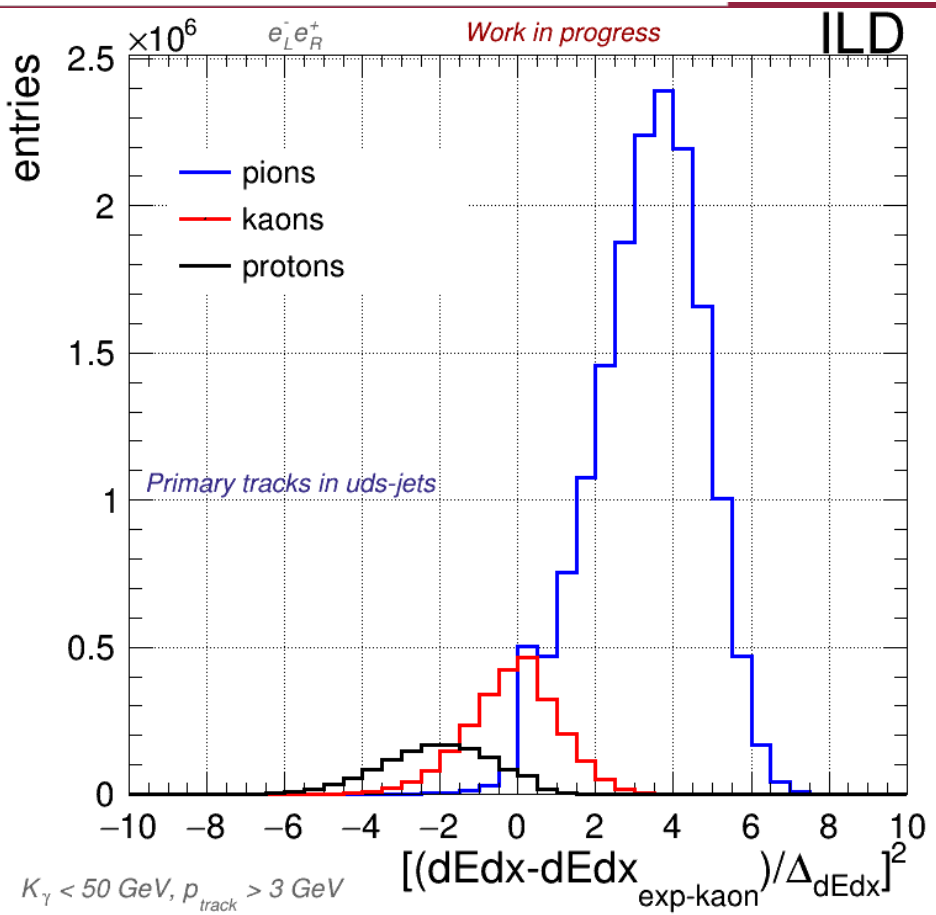


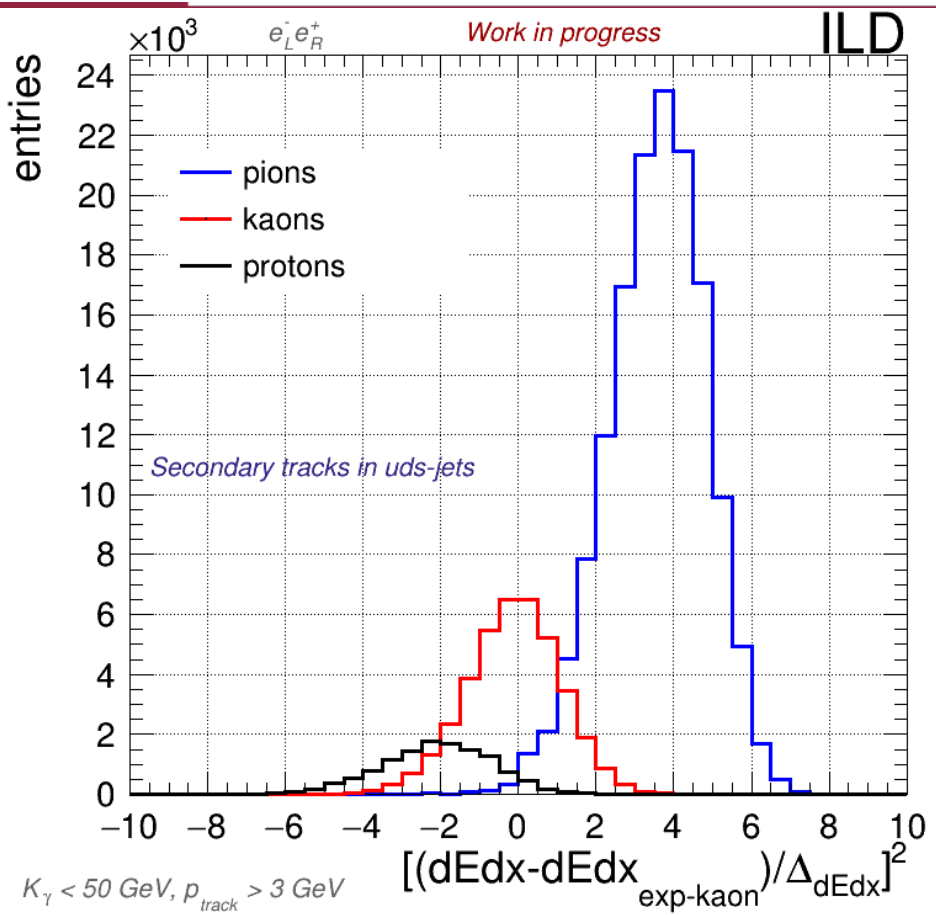


The peak at 0 is already solved cutting out cosθ > 0.95, plots are not still available...

Effects of cutting the signals at 3 GeV

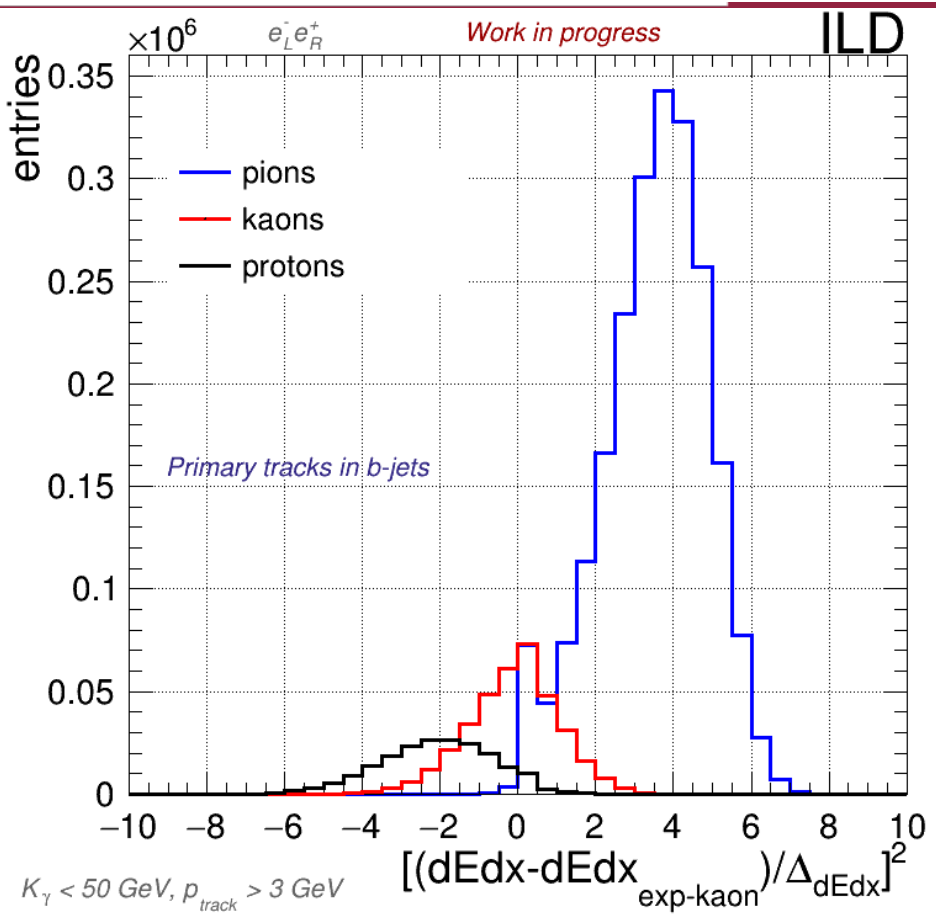
Using dEdx for flavour tagging

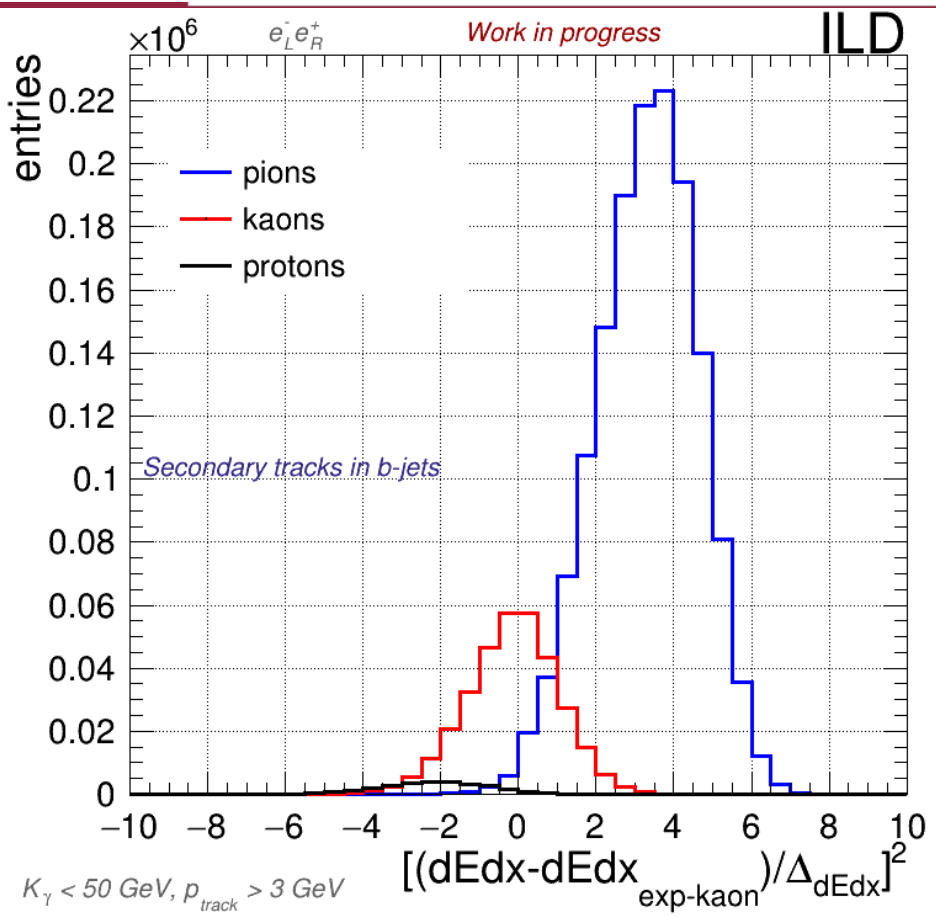




We can study separately primary & secondary tracks

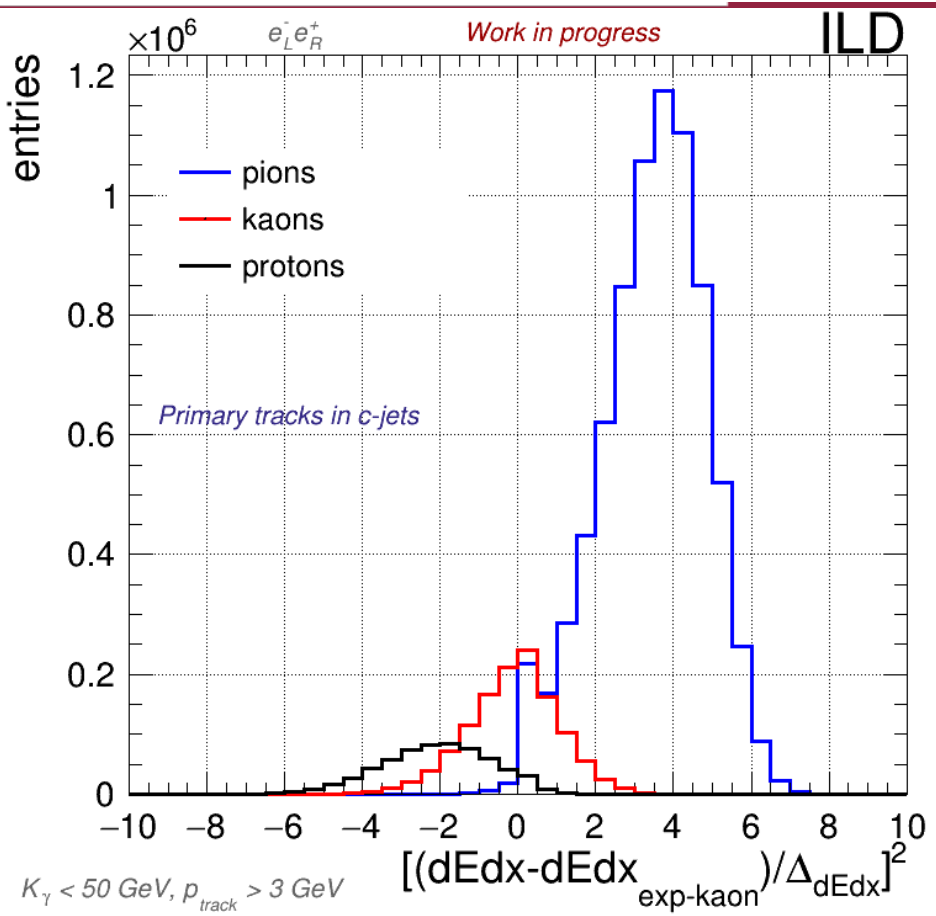
Using dEdx for flavour tagging

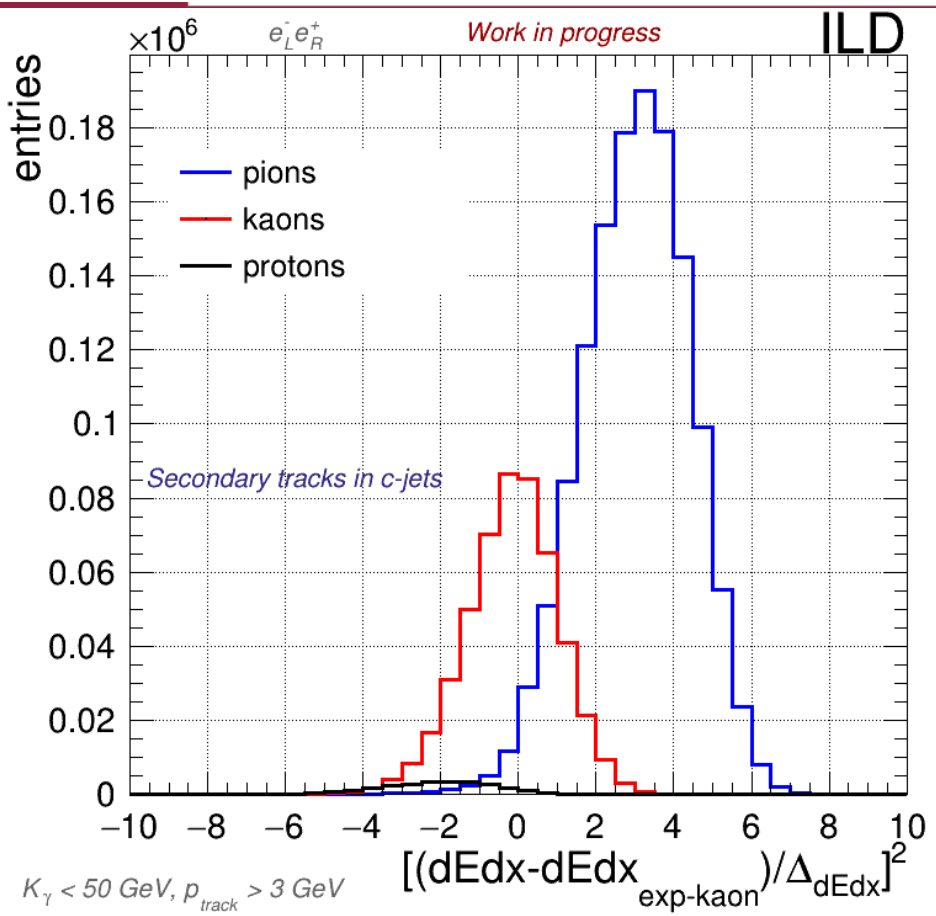




The distributions are different for different quark flavours

Using dEdx for flavour tagging





The distributions are different for different quark flavours

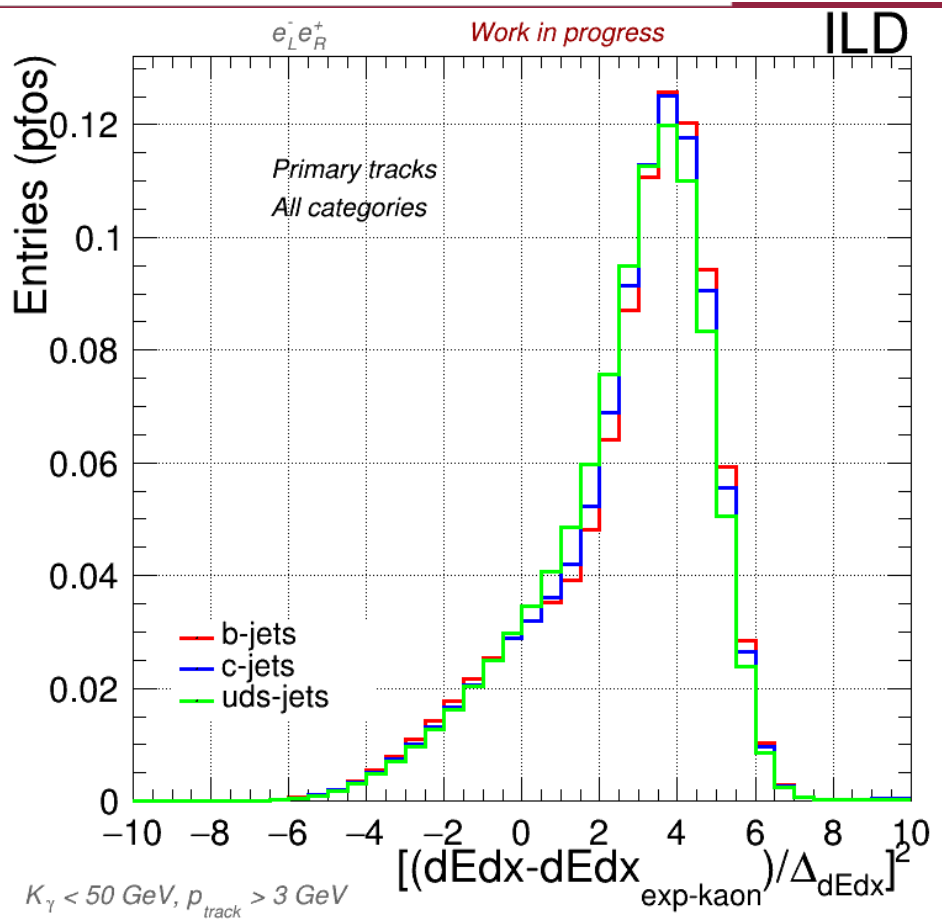
dEdx – Moving to real observables

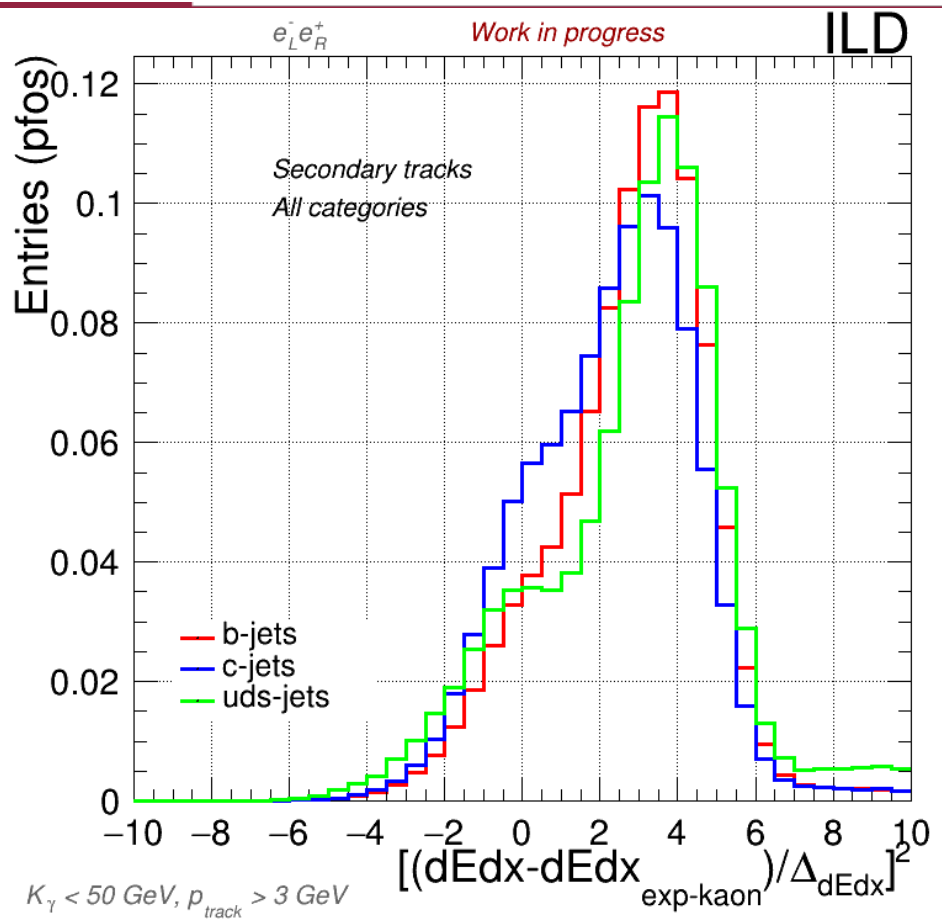
- Now that we have some hints about differences in pfos' dEdx distributions for different quark flavours we need move from quantitative distributions including all pfos to distributions that allow an inference about the content of 1 single jet according to its quark flavour.

- On the next slides we will see:
  - Histograms of untagged pfos' dEdx distance to kaon's dEdx experimental expected value.
    - Also, histograms for one *a priori* classification of pfos according to such distance: negative, null or positive distance.
      - I call these particles "Estimated protons, kaons or pions".
  - Observables using ratios between these estimated particles:
    - Estimated K/p.
    - Estimated π/p.
    - Estimated π/K.

    } **Jet by Jet!**

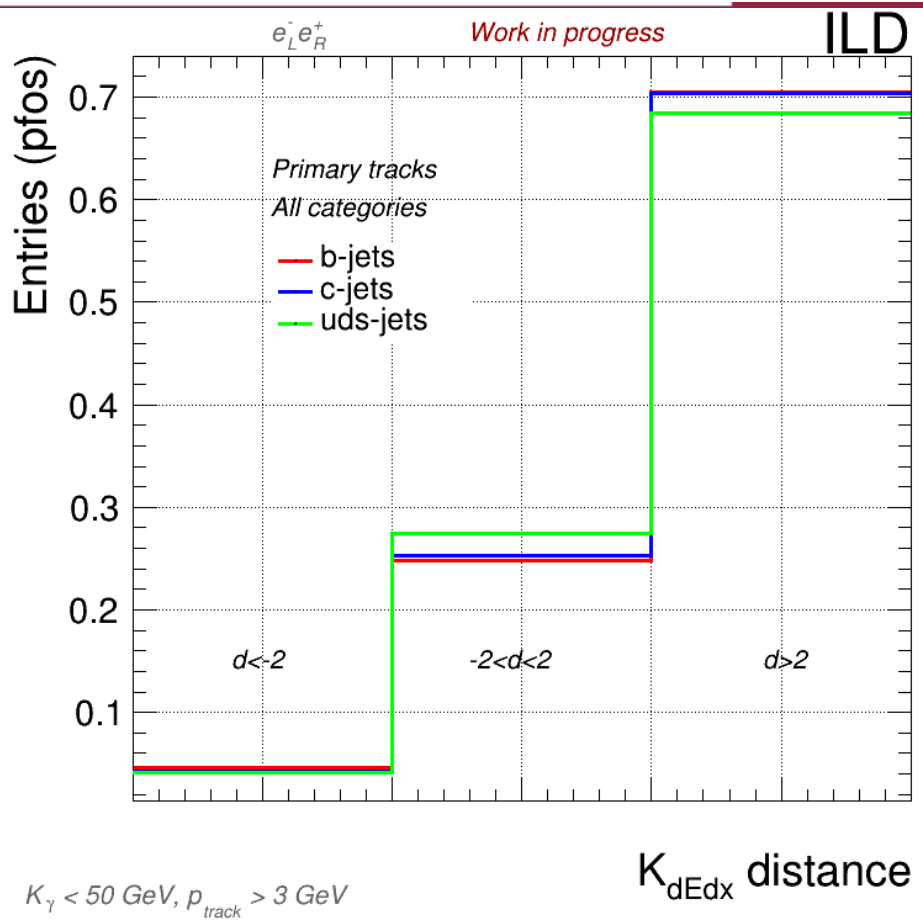
Using dEdx for flavour tagging

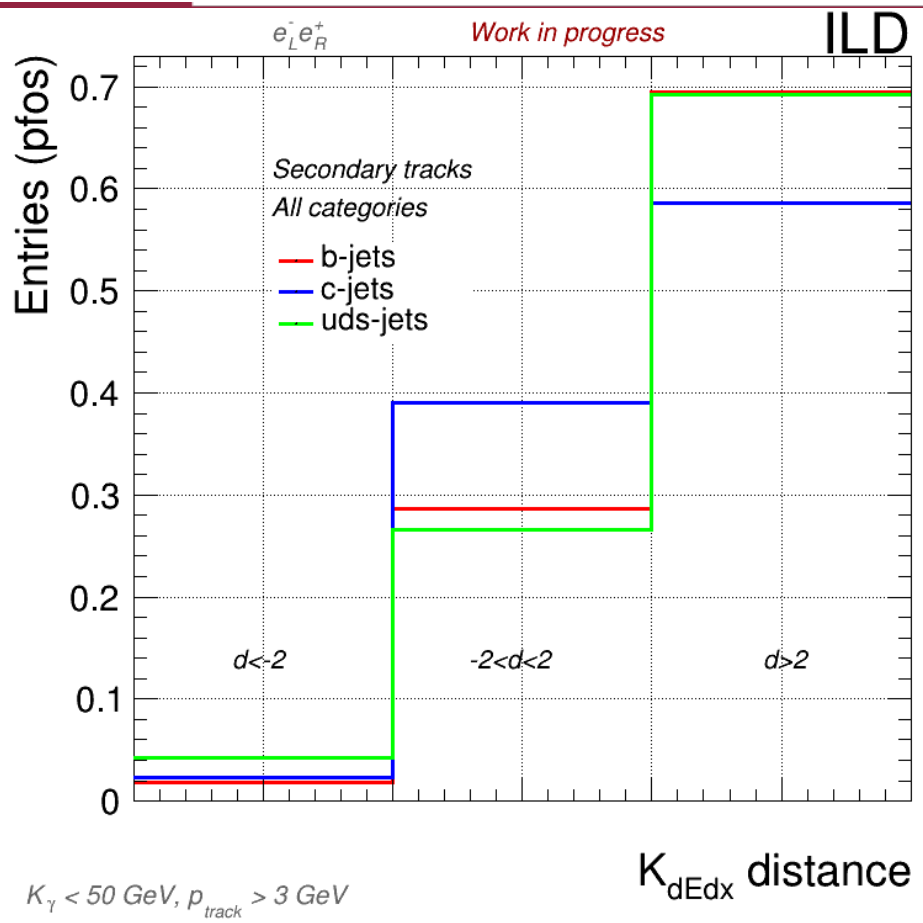




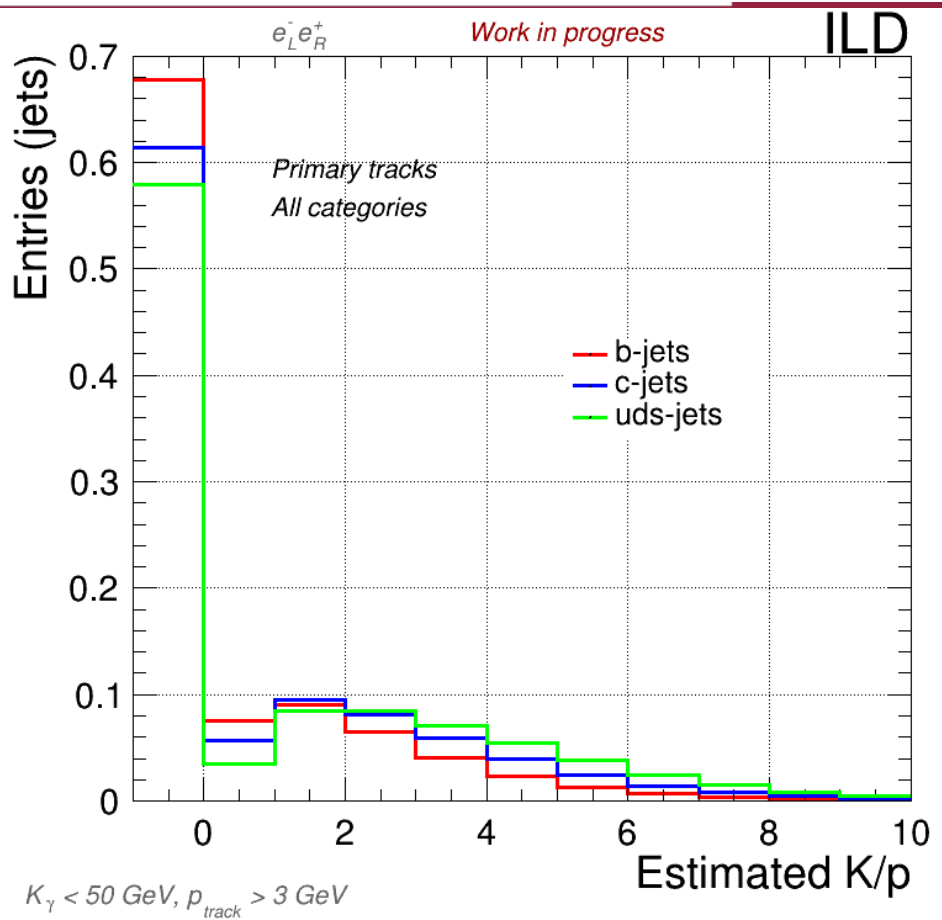
Real count of pfos' distance to the expected dEdx for kaons (no MC info used)

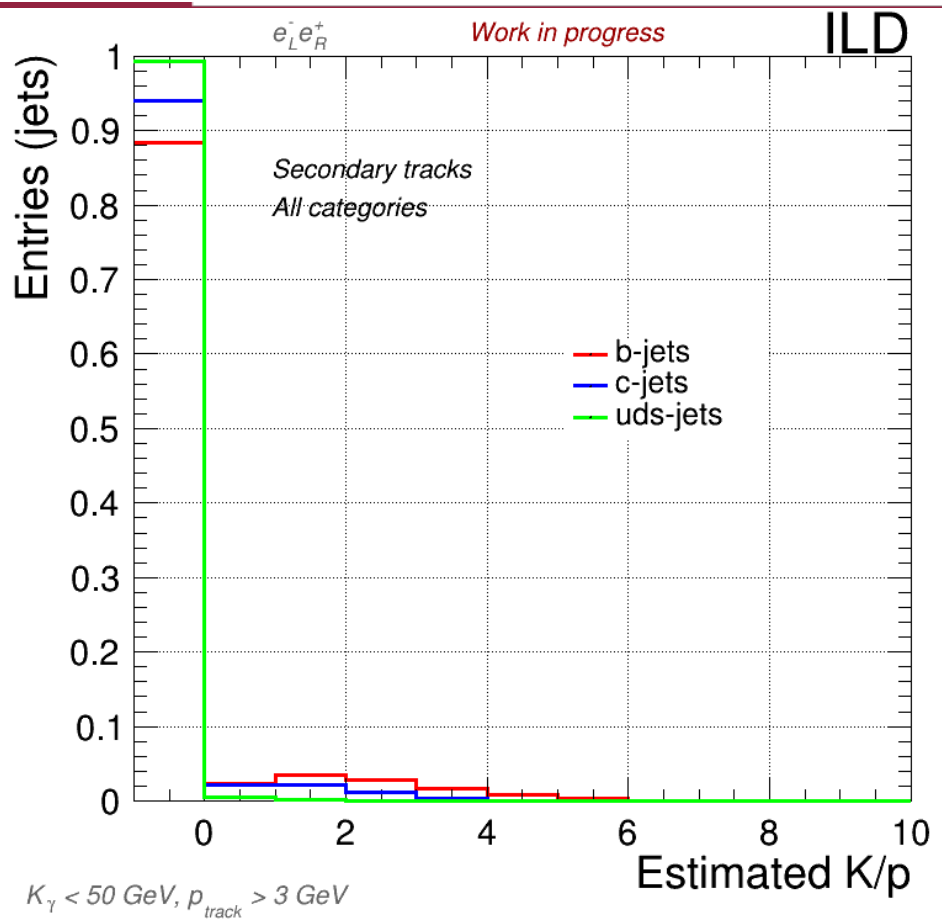
Using dEdx for flavour tagging





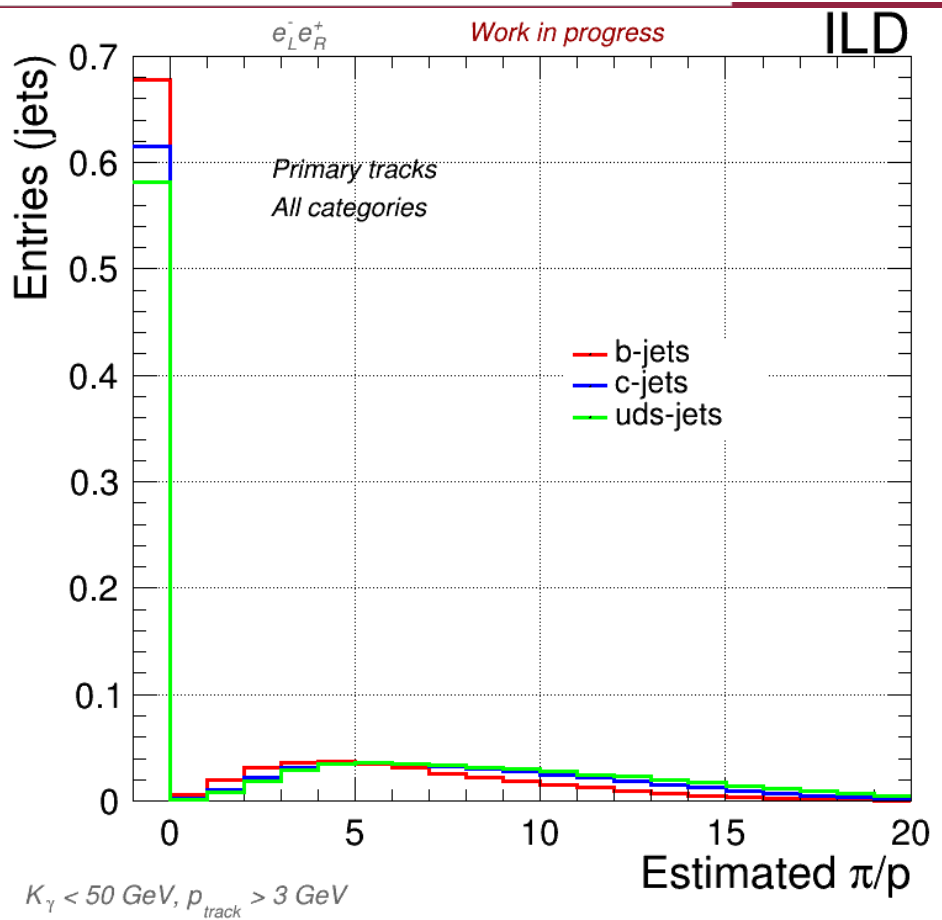
Using dEdx for flavour tagging

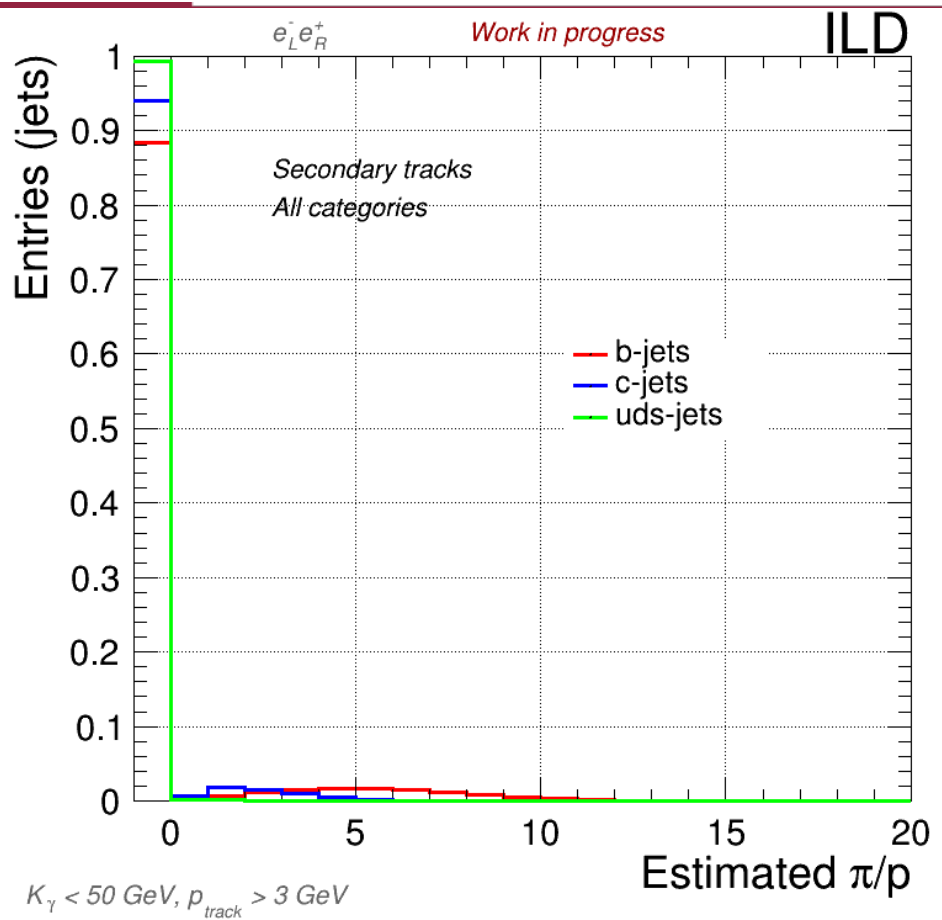




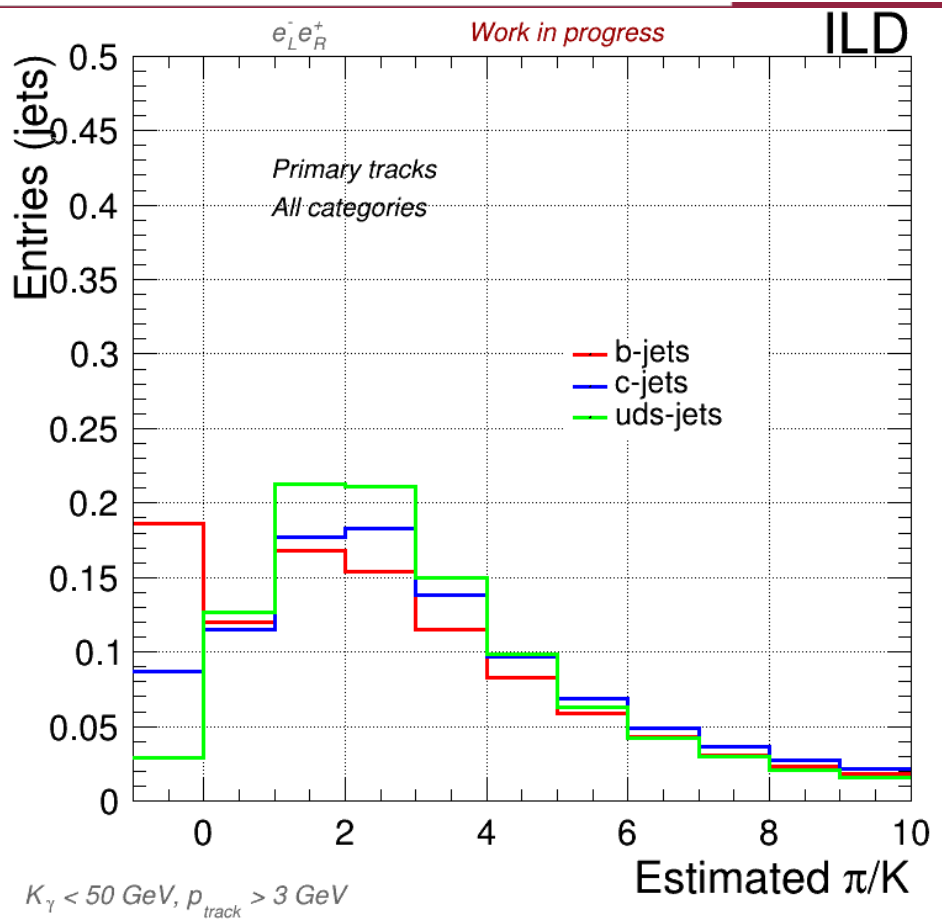
If the entry it's not well define, it's set to -1

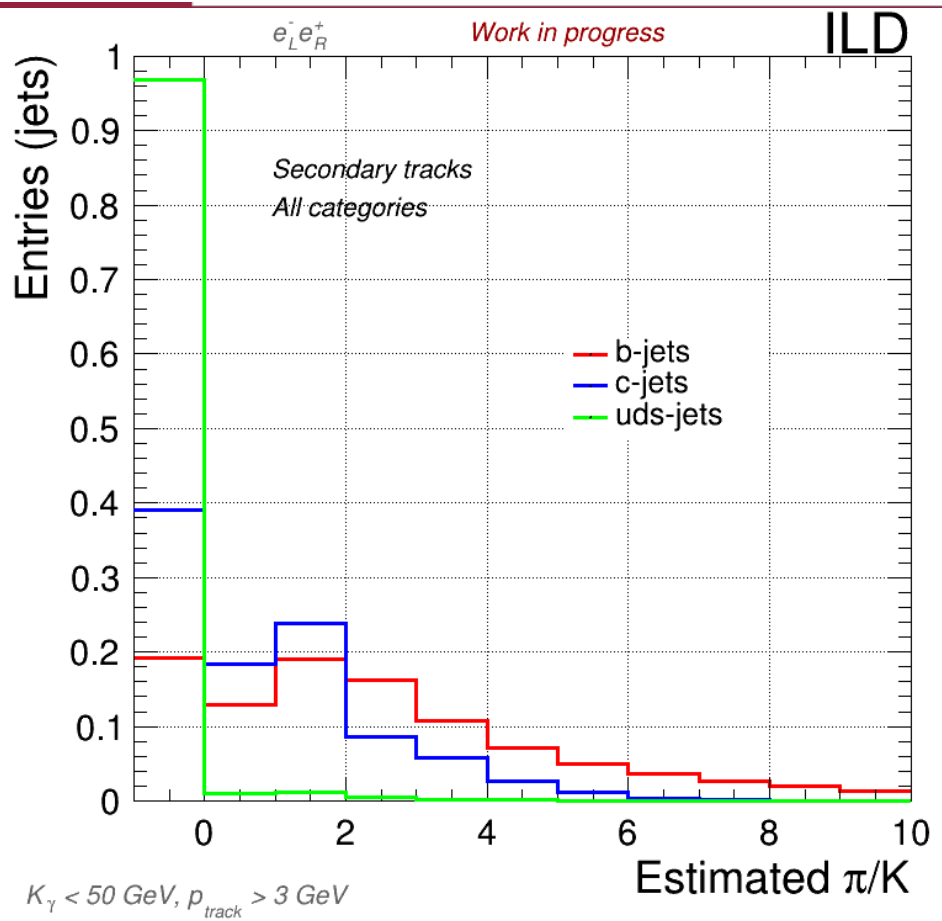
Using dEdx for flavour tagging



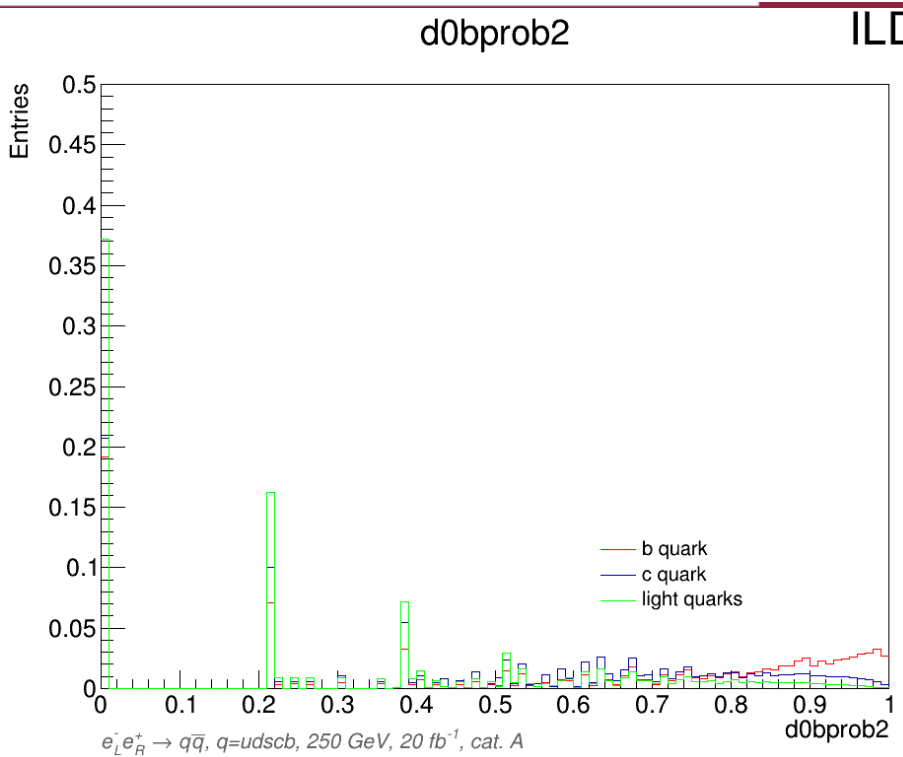


Using dEdx for flavour tagging

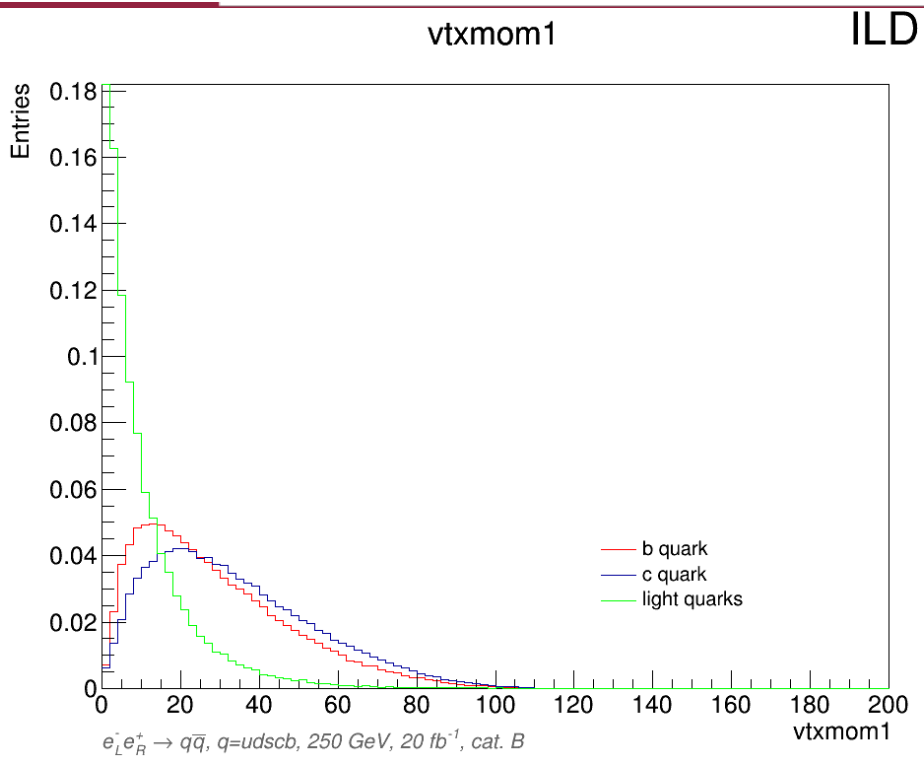




Examples of LCFI+ observables (250GeV)



Top 1 variable in cat. A



Top 1 variable in cat. B

The new observables looks promising when comparing them with the top ones

But, still, it will be useful to compare them category by category (being done atm)

dEdx – Next moves

- Computing plots *right now*:
  - Removing the glitch at d=0.
  - Trying cut p>10 GeV (Only high energy pfos).
  - Trying different definitions for the *estimated particles*.
  - Computing category by category!

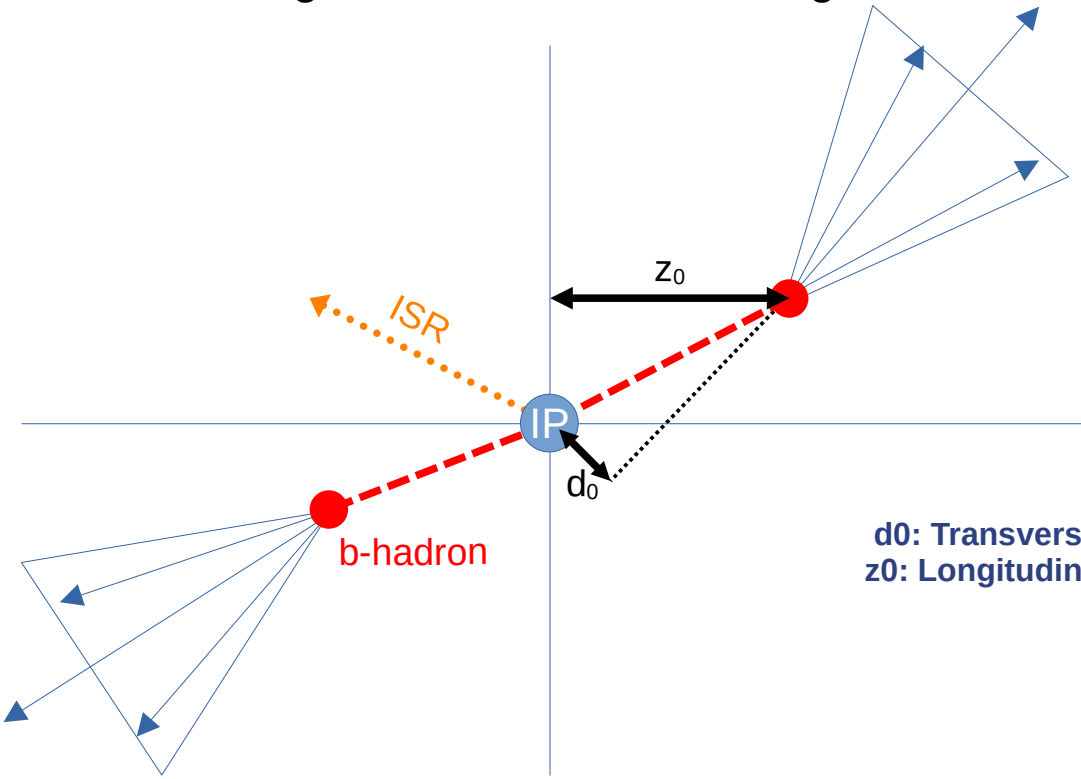
- After deciding the observables to use, time to implementing it to LCFI+:
  - 6 next observables!

Thanks for your attention

Back-up

LCFI+ (impact parameters)

We focus our attention in the most significant track, as a delegate for a secondary vertex:



d0: Transverse impact parameter
z0: Longitudinal impact parameter

TMVA in LCFI+ (variables)

Name	Description	Normalization factor	Used by category
trk1d0sig	d0 significance of track with highest d0 significance	1	A, B, C, D
trk2d0sig	d0 significance of track with second highest d0 significance	1	A, B, C, D
trk1z0sig	z0 significance of track with highest d0 significance	1	A, B, C, D
trk2z0sig	z0 significance of track with second highest d0 significance	1	A, B, C, D
trk1pt	transverse momentum of track with highest d0 significance	$1/E_{\text{jet}}$	A, B, C, D
trk2pt	transverse momentum of track with second highest d0 significance	$1/E_{\text{jet}}$	A, B, C, D
jprobr	joint probability in the r-phi plane using all tracks	1	A, B, C, D
jprobr5sigma	joint probability in the r-phi plane using all tracks having impact parameter significance exceeding 5 sigma	1	A, B, C, D
jprobz	joint probability in the z projection using all tracks	1	A, B, C, D
jprobz5sigma	joint probability in the z projection using all tracks having impact parameter significance exceeding 5 sigma	1	A, B, C, D
d0bprob	product of b-quark probabilities of d0 values for all tracks, using b/c/q d0 distributions	1	A, B, C, D
d0cprob	product of c-quark probabilities of d0 values for all tracks, using b/c/q d0 distributions	1	A, B, C, D
d0qprob	product of q-quark probabilities of d0 values for all tracks, using b/c/q d0 distributions	1	A, B, C, D
z0bprob	product of b-quark probabilities of z0 values for all tracks, using b/c/q z0 distributions	1	A, B, C, D
z0cprob	product of c-quark probabilities of z0 values for all tracks, using b/c/q z0 distributions	1	A, B, C, D
z0qprob	product of q-quark probabilities of z0 values for all tracks, using b/c/q z0 distributions	1	A, B, C, D
nmuon	number of identified muons	1	A, B, C, D
nelectron	number of identified electrons	1	A, B, C, D
trkmass	mass of all tracks exceeding 5 sigma significance in d0/z0 values	1	A, B, C, D

TMVA in LCFI+ (variables)

Name	Description	Normalization factor	Used by category
1vtxprob	vertex probability with all tracks associated in vertices combined	1	B, C, D
vtxlen1	decay length of the first vertex in the jet (zero if no vertex is found)	$1/E_{\text{jet}}$	B, C, D
vtxlen2	decay length of the second vertex in the jet (zero if number of vertex is less than two)	$1/E_{\text{jet}}$	D
vtxlen12	distance between the first and second vertex (zero if number of vertex is less than two)	$1/E_{\text{jet}}$	D
vtxsig1	decay length significance of the first vertex in the jet (zero if no vertex is found)	$1/E_{\text{jet}}$	B, C, D
vtxsig2	decay length significance of the second vertex in the jet (zero if number of vertex is less than two)	$1/E_{\text{jet}}$	D
vtxsig12	vtxlen12 divided by its error as computed from the sum of the covariance matrix of the first and second vertices, projected along the line connecting the two vertices	$1/E_{\text{jet}}$	D
vtxdirang1	the angle between the momentum (computed as a vector sum of track momenta) and the displacement of the first vertex	E_{jet}	B, C, D
vtxdirang2	the angle between the momentum (computed as a vector sum of track momenta) and the displacement of the second vertex	E_{jet}	D
vtxmult1	number of tracks included in the first vertex (zero if no vertex is found)	1	B, C, D
vtxmult2	number of tracks included in the second vertex (zero if number of vertex is less than two)	1	D
vtxmult	number of tracks which are used to form secondary vertices (summed for all vertices)	1	D
vtxmom1	magnitude of the vector sum of the momenta of all tracks combined into the first vertex	$1/E_{\text{jet}}$	B, C, D
vtxmom2	magnitude of the vector sum of the momenta of all tracks combined into the second vertex	$1/E_{\text{jet}}$	D
vtxmass1	mass of the first vertex computed from the sum of track four-momenta	1	B, C, D
vtxmass2	mass of the second vertex computed from the sum of track four-momenta	1	D
vtxmass	vertex mass as computed from the sum of four momenta of all tracks forming secondary vertices	1	B, C, D
vtxmasspc	mass of the vertex with minimum pt correction allowed by the error matrices of the primary and secondary vertices	1	B, C, D
vtxprob	vertex probability; for multiple vertices, the probability P is computed as 1-P = (1-P1)(1-P2)...(1-PN)	1	B, C, D

Boosted Decision Trees (BDT) - Overview

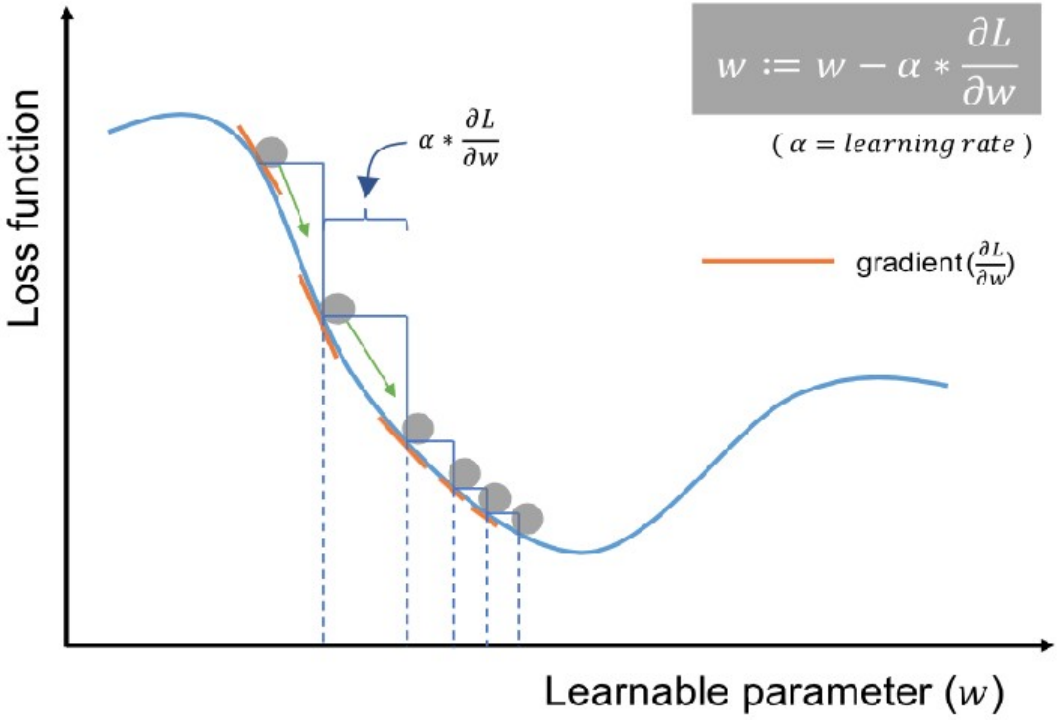
- A decision tree is a weak learner (an estimator): It classifies your data according to certain/s question/s (nº of leaves).
- Boosting is using many of these trees one after each other until you get a final classification.
  - Gradient Boosting is the most common one (and the one I'm using):

$$F_m(x) = F_{m-1}(x) + \eta \cdot h_m(x),\ 0 < \eta \leq 1$$

    - Where F is the total prediction, h the prediction of a tree, and η is the learning rate (or shrinkage).
      - Each of the trees actually perform its classification using the gradient of the loss function of the previous one, so it keeps "refining" the result.

Boosted Decision Trees (BDT) - Overview

- Simple visual representation[1]:



1. Website (Bradley Boehmke & Brandon Greenwell): https://bradleyboehmke.github.io/HOML/gbm.html

PSO – Implementation - Origin

- I didn’t start from scratch, but rather adapted Andrej Saibel’s code. This code:
  - Is a signal-background classifier (2 classes).
    - Uses ROOT’s classes that are optimal to Signal-vs-Background classifiers.
  - Includes nTuples variables as extra parameters to play with.
    - Optimizes the use of physical variables as well.
  - Includes a test to avoid overfitting (Kolmogorov-Smirnoff test).
  - Includes different types of FOM to choose from.
  - Was originally prepared to run in CMS computing services.
    - There are different codes interacting in Python, C (C++) and bash; to prepare the particles, executing them, read the results and update their new configurations.

PSO – Kolmogorov-Smirnov Test

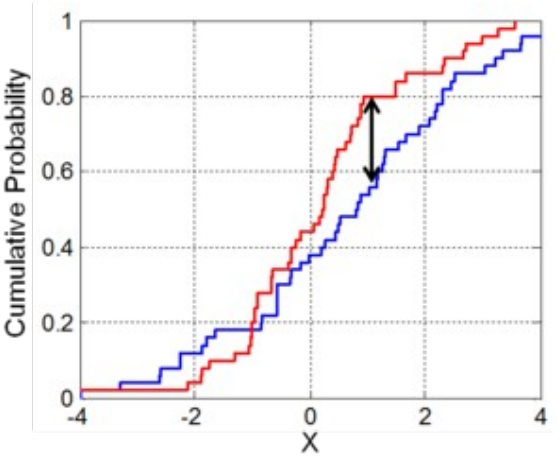
- Compare how likely there is that two different empirical distributions (histograms) came from the same underlying distribution function.
  - It uses the max. distance between the cumulative probability(CPD) of both histograms:

$$D_{n,m} = \sup_x |F_{1,n}(x) - F_{2,m}(x)|$$

  - Then, we past a test for such distance to a certain degree of significance level α (usually 0.05):

$$D_{n,m} > \sqrt{-\ln\left(\frac{\alpha}{2}\right) \cdot \frac{1+\frac{m}{n}}{2m}}.$$

  - The output is a p-value which determine how likely it is that both histograms came from the same distribution according to our significance level (e.g. 0.05 stands for 95% of agreement).



Notice how a big jump in the CPD even in a very narrow region will lead to a very high distance (low KS score): Hyper-sensibility if the distributions are not smooth enough

PSO – Anderson-Darling Test

- The AD test statistic is defined as:

$$A^2 = -n - S$$

- Where:

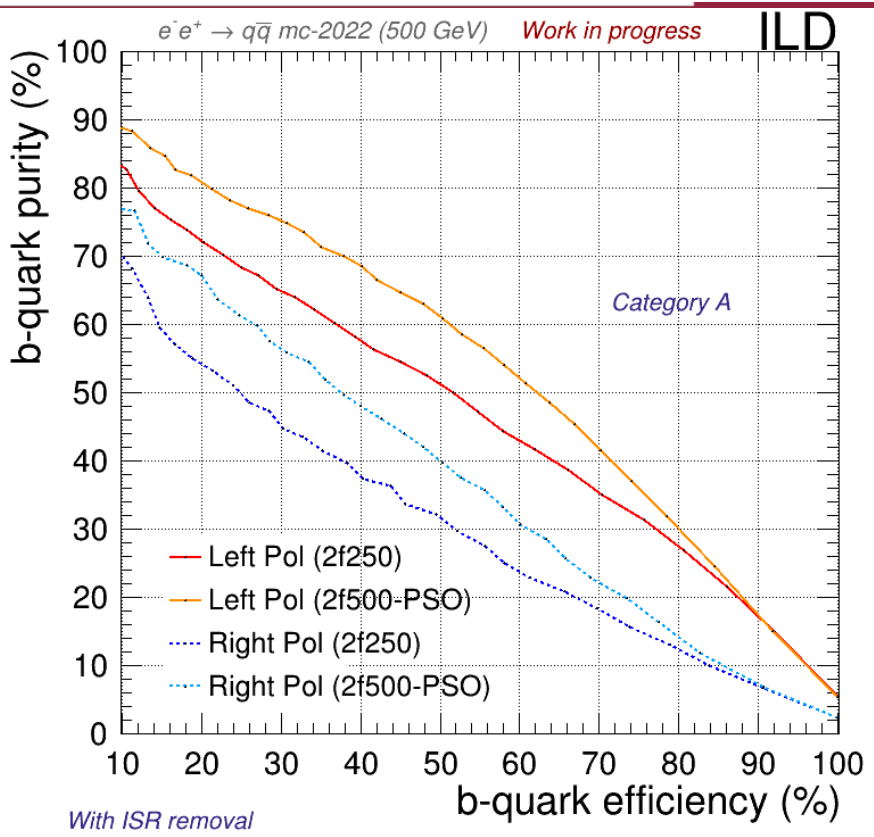
$$S = \sum_{i=1}^{n} \frac{2i-1}{n} \left[\ln(F(Y_i)) + \ln(1 - F(Y_{n+1-i})) \right]$$

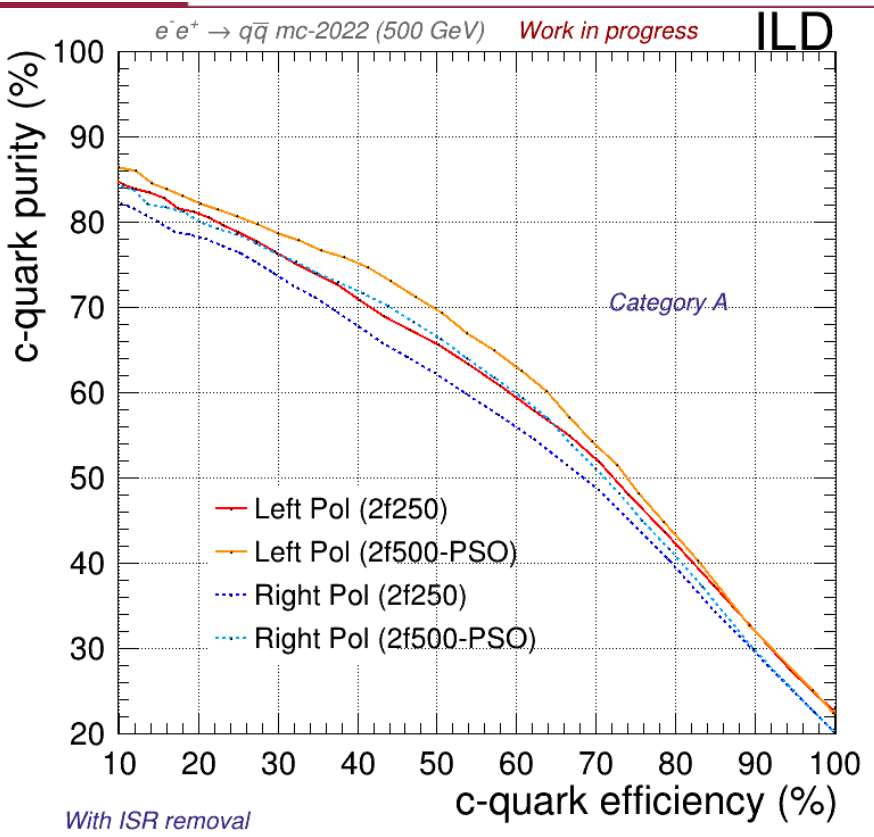
- Being F the cumulative probability distribution for a certain distribution (or the other sample in our 2-samples scenario).
  - Works better with uniform distributions and higher binning.
- Again, the output is an estimator based on a cut in $A > A_{critical}$

Notice how this kind of testing avoid the hyper-sensibility that we had in narrow jumps in the CPD but what if one of these jumps in CPD is actually relevant?

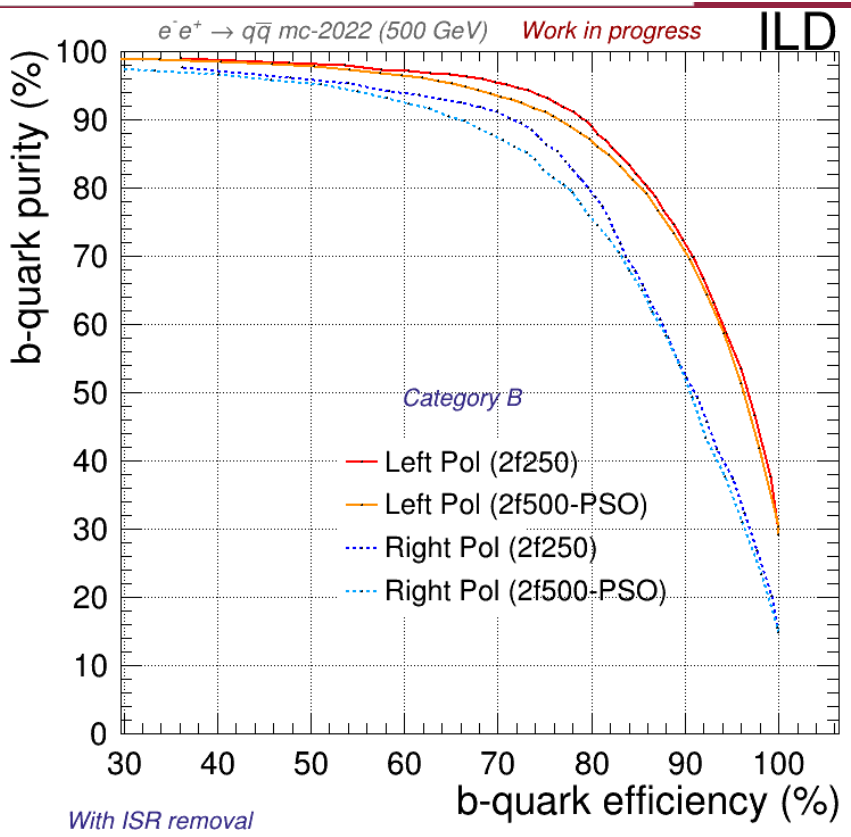
I chose very conservative (and secure) way to proceed: Applying both tests!

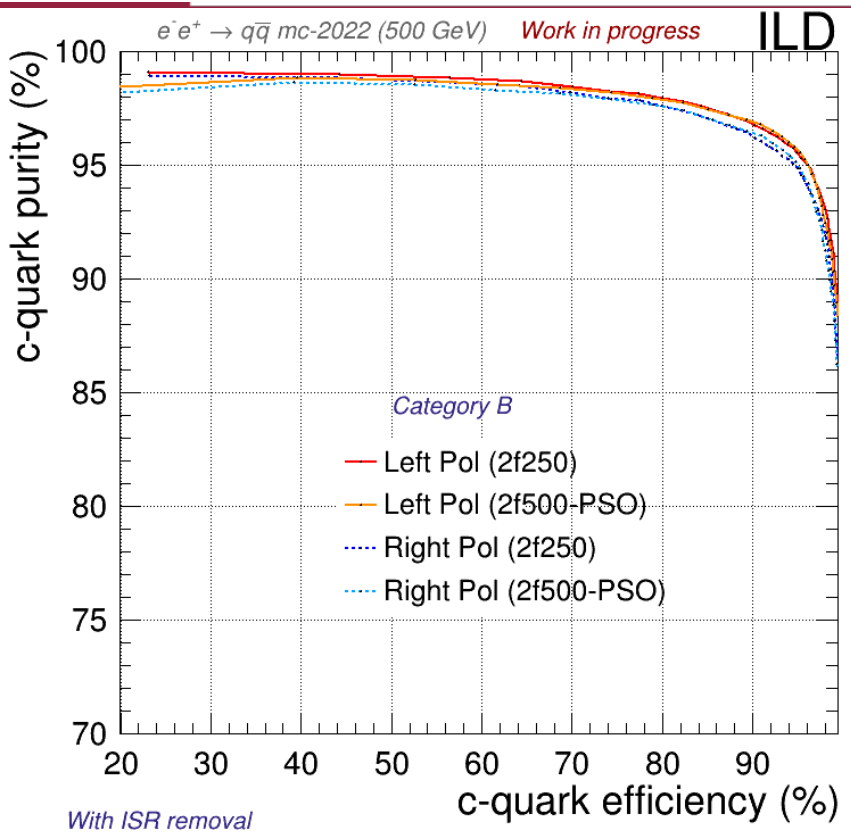
PSO Performance (500 GeV – cat. A)





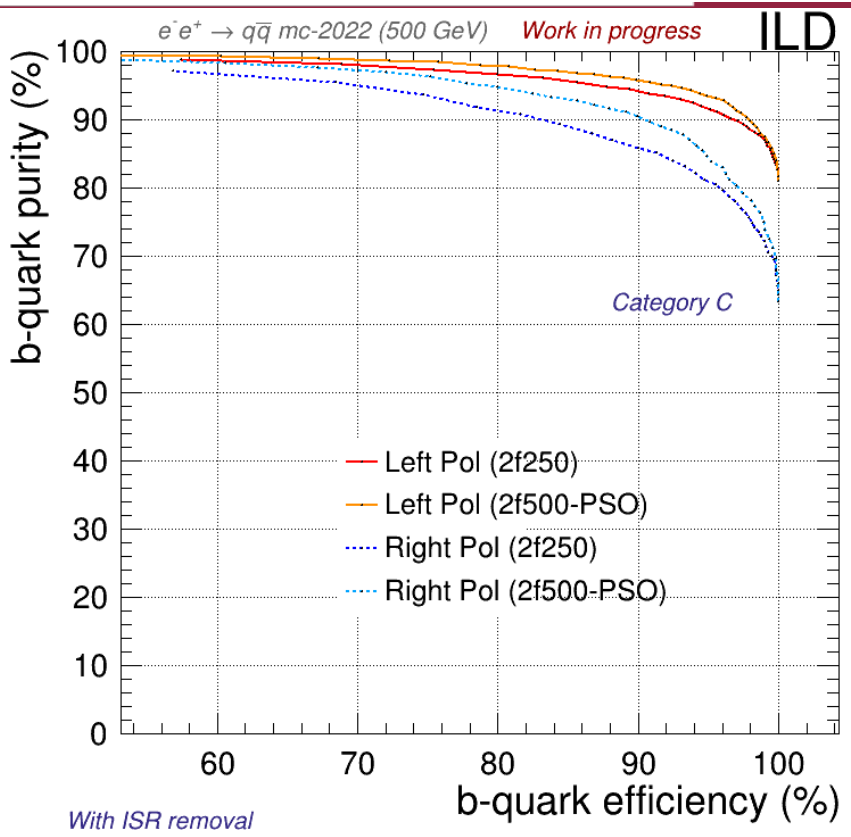
PSO Performance (500 GeV – cat. B)

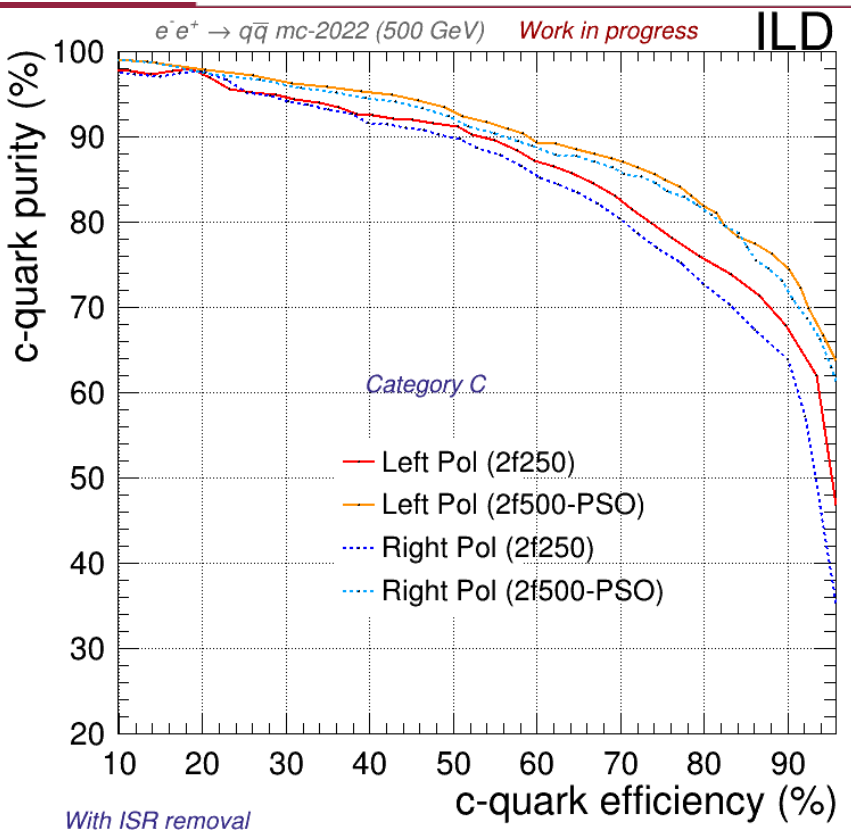




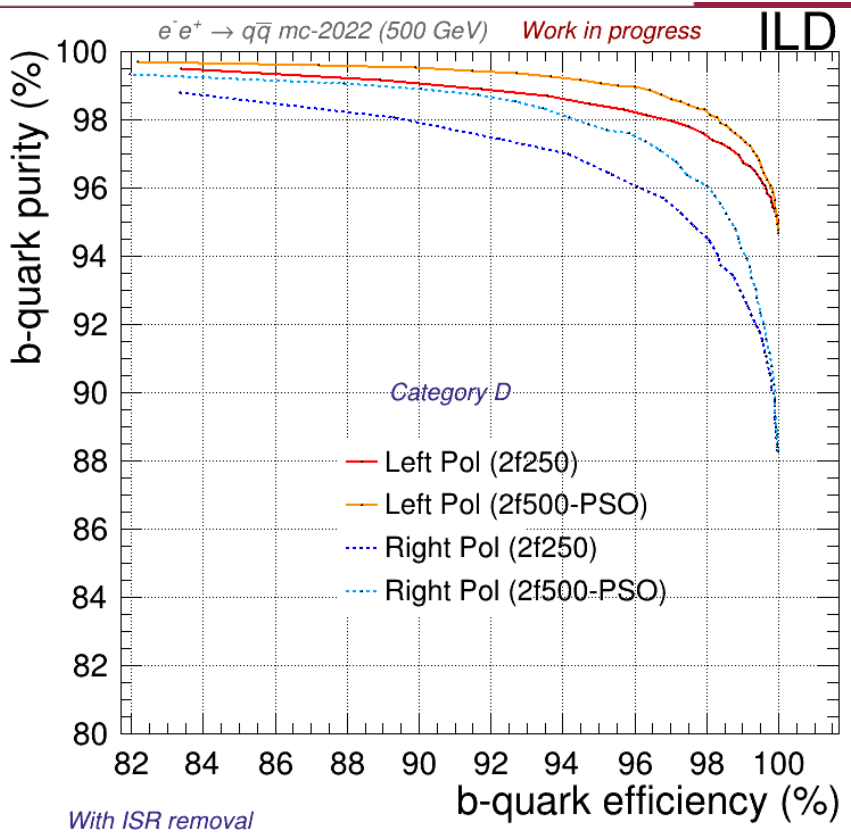
This category does not perform well and it's being re-optimized

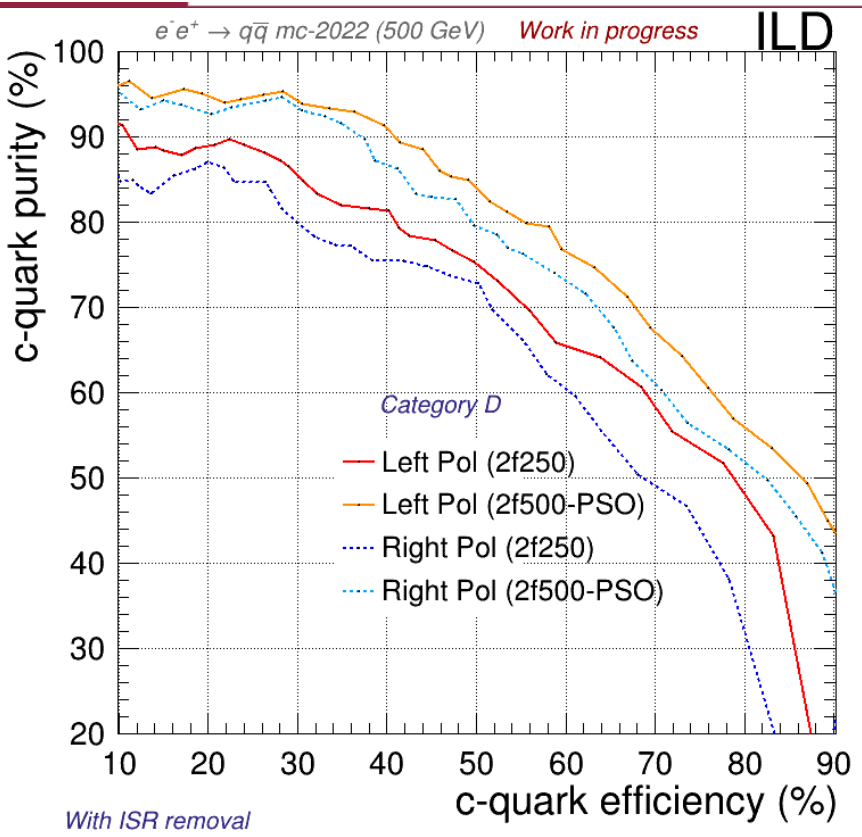
PSO Performance (500 GeV – cat. C)



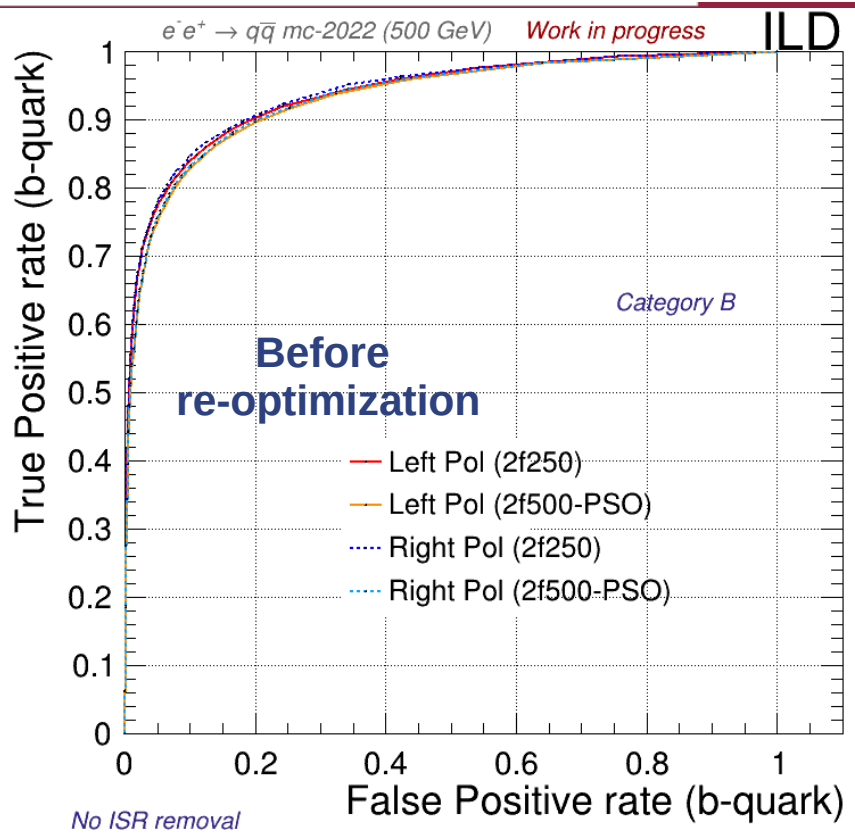


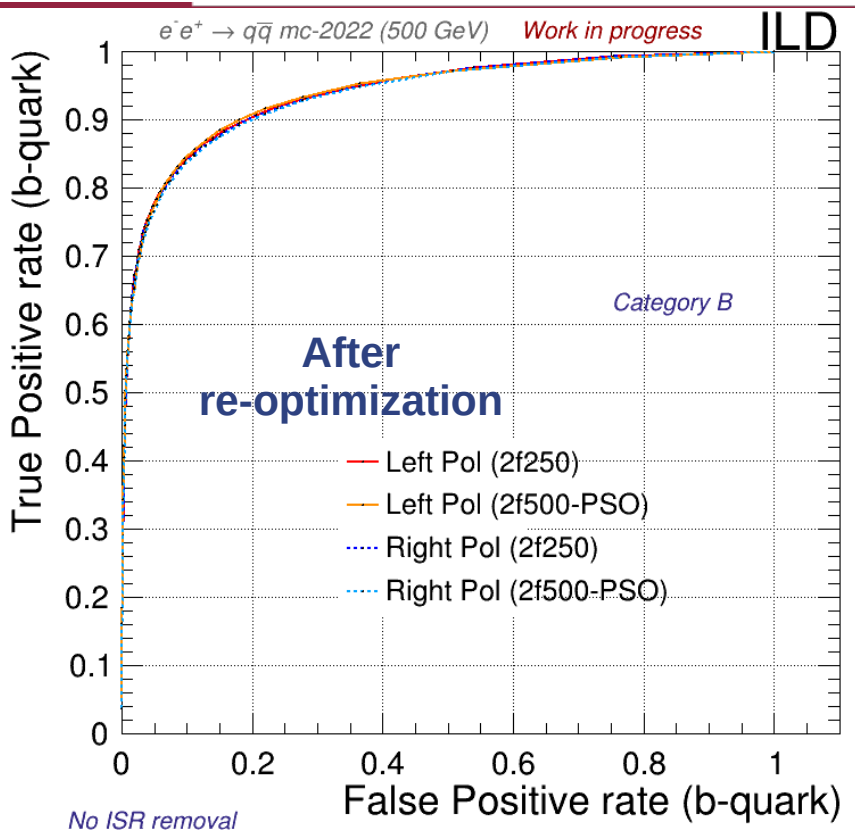
PSO Performance (500 GeV – cat. D)





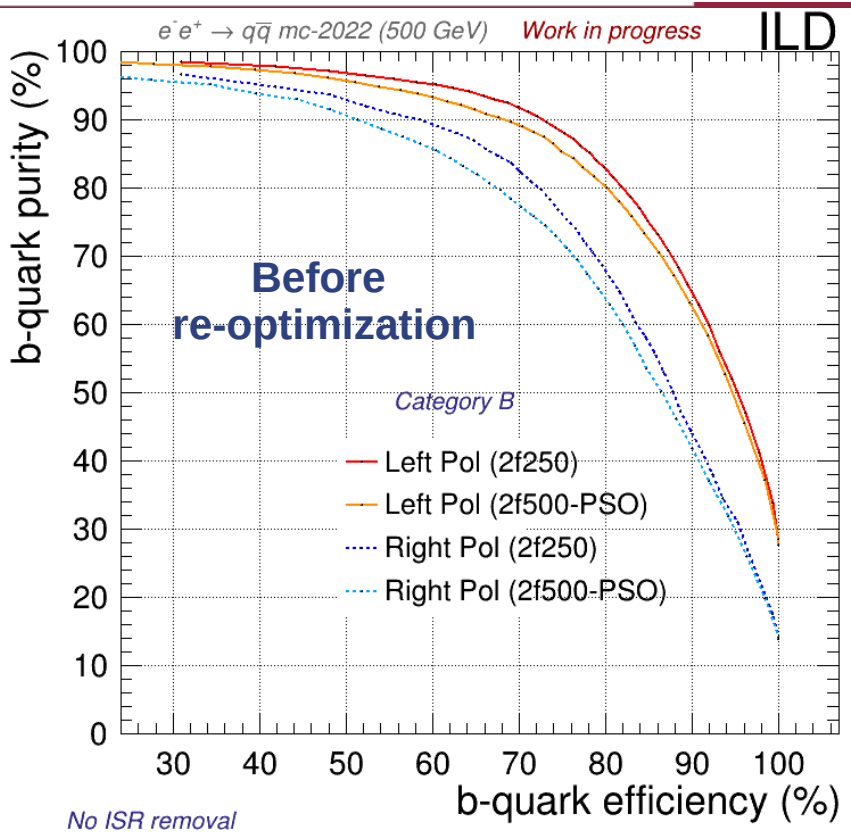
PSO Performance – Polarization & cat. B

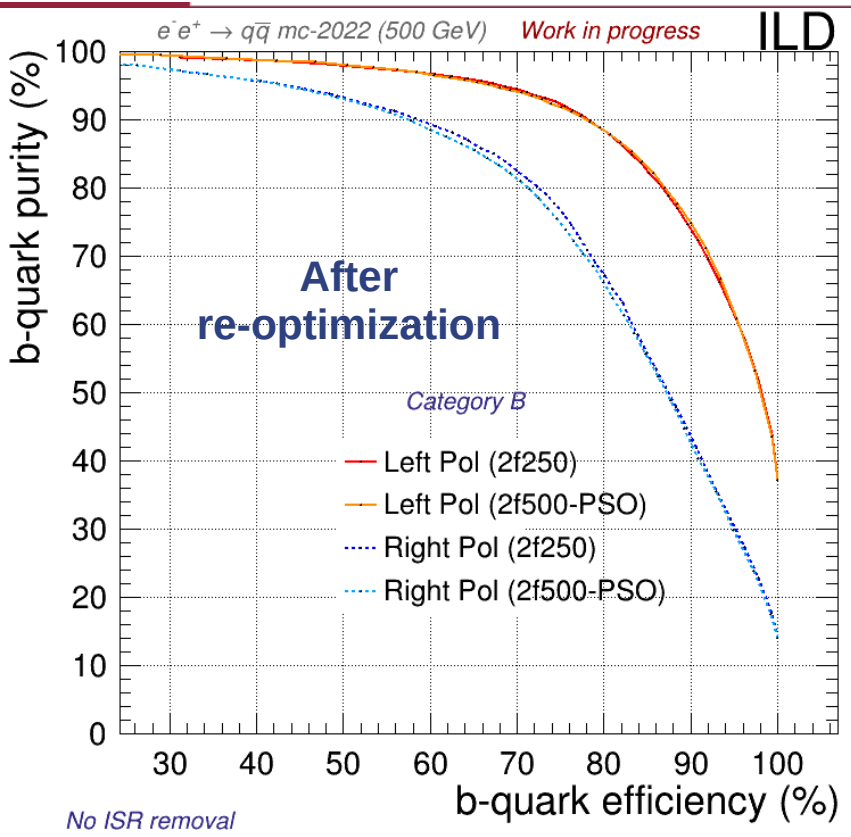




Test done in different samples: Focus only in the difference between old & new weights

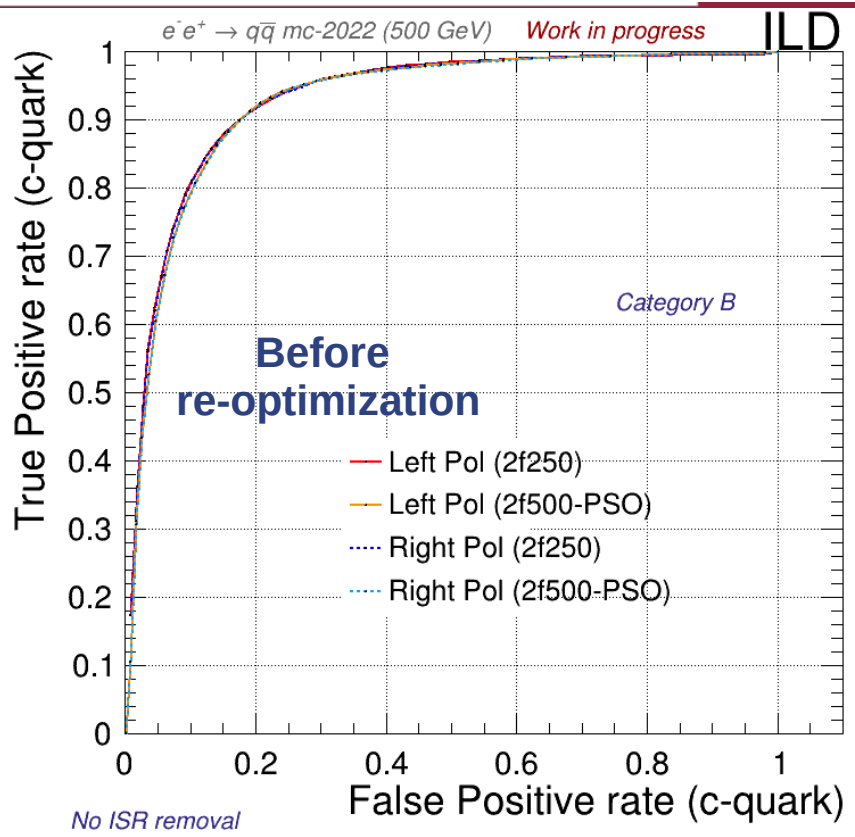
PSO Performance – Polarization & cat. B

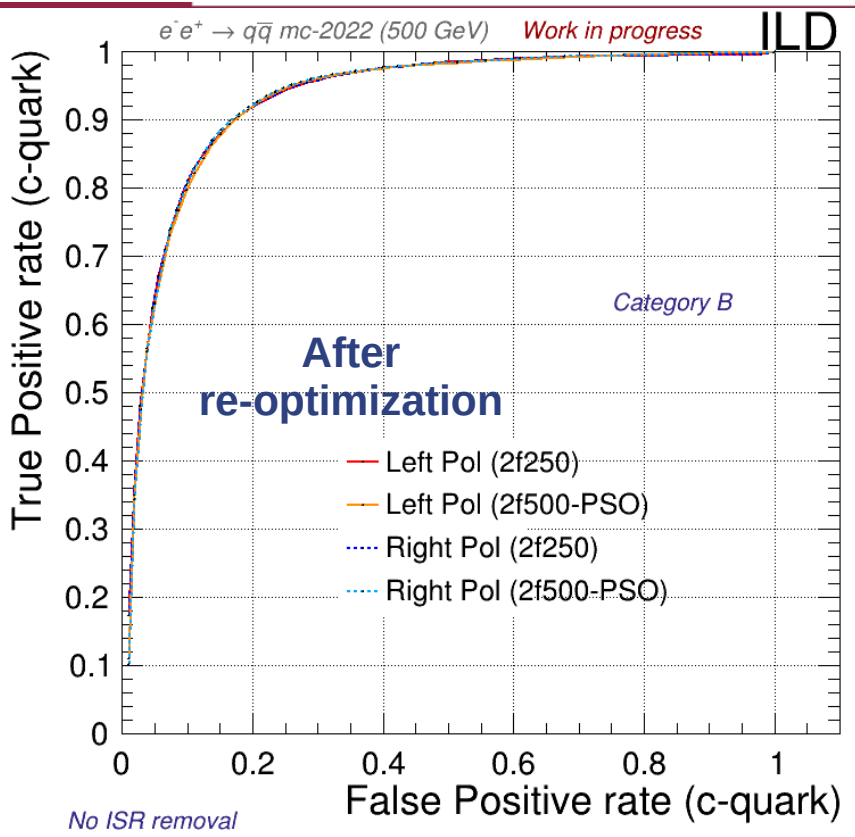




Test done in different samples: Focus only in the difference between old & new weights

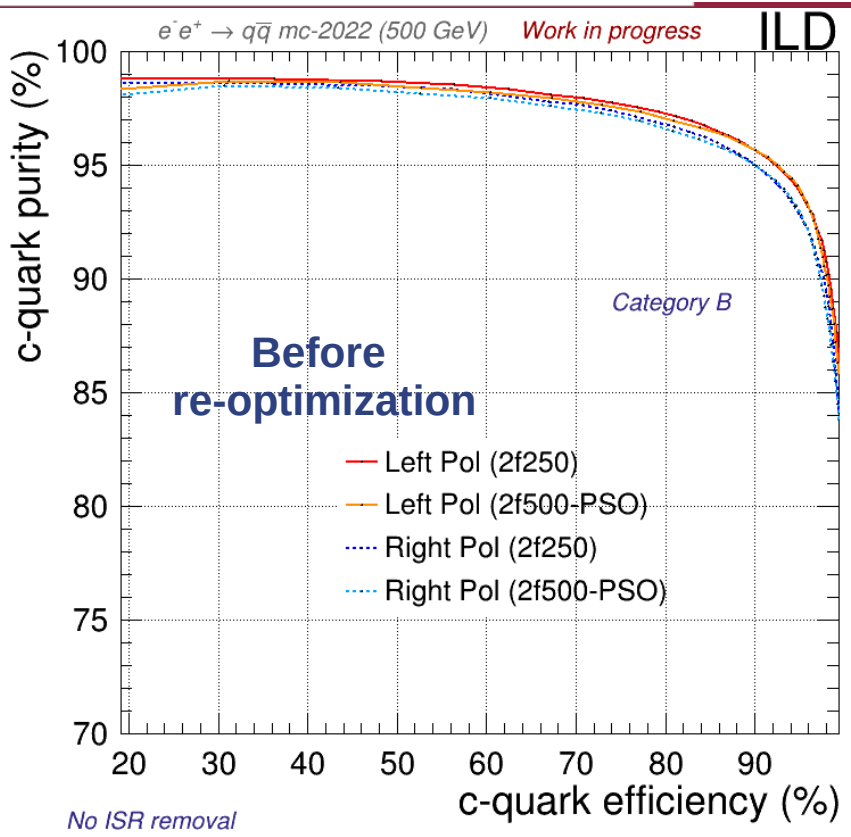
PSO Performance – Polarization & cat. B

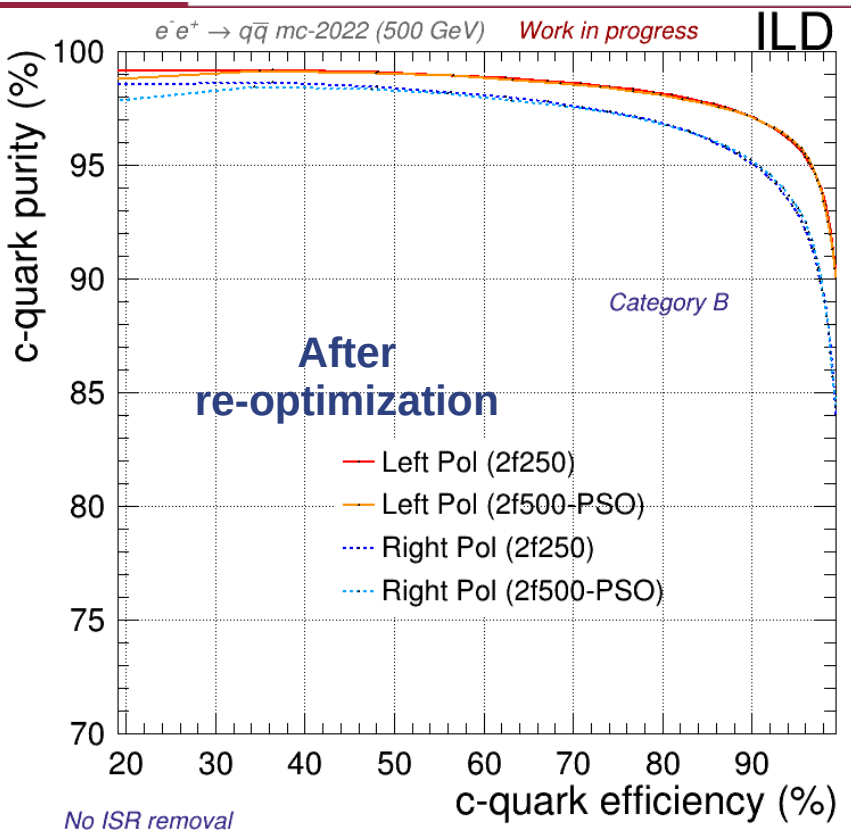




Test done in different samples: Focus only in the difference between old & new weights

PSO Performance – Polarization & cat. B





Test done in different samples: Focus only in the difference between old & new weights