University of Zurich[UZH]

Physik-Institut

Masterhesis: Implementation of Monte Carlo fitter and Selection of $B \to K^*\mu\mu$ decays in LHCb Run2 data.

Supervisors: Prof. Dr. Nicola Serra, Prof. Dr. Marcin Chrzsąszcz

Author: Oliver Dahme

The work presented in this thesis consists of two parts, which are all intended to be used in the contents of data analysis of data taken by the LHCb detector, especially to analyse the decay $B \to K^*\mu\mu$.

1. A new fitting routine has been developed as a more transparent and reliable alternative to the broadly used routine Minuit.
2. A test of boost decision tree algorithms has been performed. boost decision trees are algorithms used to separate signal from background. The test in this thesis was performed to find the best boost decision tree algorithm in order to select events with the decay $B \to K^*\mu\mu$. In addition data from the Run II of the LHCb detector was used to reweight the Monte Carlo simulation of events with the decay $B \to K^*\mu\mu$.

Contents

Introduction

This thesis consists of two parts: The implementation of a new fitting routine and the selection of the $B \to K^* \mu\mu$ decay in the raw data recorded by the LHCb detector.
In physics a fit determs free parameters of a theory by using experimental data. For example the theory to describe an exponential decay is defined as:

$$N(t) = N_0 e^{-\lambda t} \tag{1}$$

where N_0 is the number of particles present at the beginning ($t = 0$), $N(t)$ the number of particles at time t and λ the decay constant. The goal of such a theory is to predict the amount of particles present after a time t has passed. But for this one needs the decay constant λ. This is a free parameter of this theory. To get the λ parameter an experiment is designed which measures the number of particles in a certain time interval, lets say every second.
To get the decay constant this measurment is now fitted to the data. The fit will minimize a so called negative log liklihood function, which depends on the the theory and on every measurment. The only variable in this function is λ. The value for λ at the global minimum of the negative log liklihood function is the one where the theory matches the measurment. To stretch that point imagine one would plot the data points and the graph of the equation 1 wiht the λ from the fit. If the theory is correct the graph of equation 1 should touch every data point, like it does in figure 4. As part of this thesis an algorithm has been implemented into CERN's Data Analysis Framework [3], to perform such fits.
The second part of this thesis is about selecting the $B \to K^* \mu\mu$ out of the data obtained by the LHCb experiment. Selecting means to distinguish the decay from all other decays in contained in the data. This is done with boost decision trees, a form of algorithm from the Machine Learning area. They are able to find thresholds on parameters in the data. For example the decay $B \to K^* \mu\mu$ happens if the flight distance of the B meson is between 0.5 and 2 centimeters. Then all the decays in the data with a flight distance outside this range can be cut off and only the $B \to K^* \mu\mu$ decays remain. To fully analyse a decay not only the data from the detector is important, but also the efficiencies of the detector have to be considered. To get these a Monte Carlo simluation of the detector is perfomed. But since the simulation and the data taking are not done at the same time. The simulation might differ from the data, because for example some part of the detector has gone offline during data taking. To compensate this effect the Monte Carlo simulation has to be matched to the real data. This match is done by applying a certain weight to each Monte Carlo Event. The weights are dertermed by comparing the simulation with the data taken by the detector.
Outline: Chapter one decribes the new fitting routine in detail. Chapter two describes the $B \to K^* \mu\mu$ decay and how the selection and Reweighting for the analysis of this decay is done. Chapter three gives a quick summery over the thesis and an outlook what could be done in the future.

1. Fitting Routine

1.1. Likelihood principle

Let's assume a random variable x_i follows a probability density function (short pdf) $f(x_i|a_j)$. From theory one knows the functional form of f but not the values of so called parameters of intrests (POI) denoted as a_j. In most common physics application the x_j is a given data point from the set $\{x_i\}$. The goal is to obtain an estimate of each unknown parameter a_k by fitting $f(x_i|a_j)$ to the data points. To obtain the values of POI several methods have been developed, the most popular ones are the Maximum Likelihood method, the Chi-square method and the Method of moments. In this section the Likelihood method is presented; one starts by writing down the so called likelihood function defined as:

$$L(a_j) = f(x_1|a_j) \cdot f(x_2|a_j) \cdot \ldots \cdot f(x_N|a) = \Pi_{i=1}^{N} f(x_i|a_j) \tag{2}$$

The value of $L(a_j)$ represents how well the POI suits the data points. Therefore, one needs to find the maximum of $L(a_j)$. The POI values that maximise the likelihood function are then treated as estimators of the POI.
From technical point of view multiplying many floating point numbers is numerically unstable. Furthermore rounding errors are added up per multiplication. To solve both problems one introduces the Log-Likelihood function and changes its sign. The new function is called negative log-likelihood (short nnl) function and is defined as:

$$nll(a_j) = \ln[L(a_j)] = -\ln[f(x_1|a_j)] - \ln[f(x_2|a_j)] - \ldots = -\Sigma_{i=1}^{N} \ln[f(x_i|a_j)] \tag{3}$$

Just as important as getting an estimate of the POI is to calculate the uncertainty of the estimate. The Likelihood method also provides an estimate of the uncertainty. Consider a $nll(a)$ with just one parameter a, the estimate will be denoted as a^* and $L(a^*)$ as L^*. For one parameter the nll becomes Gaussian for a large number of x_js.

$$\ln(L) \approx \ln(L^*) + \frac{1}{2} \frac{\partial^2 \ln(L)}{\partial a^2}|_{a=a^*} (a - a^*)^2 \tag{4}$$

$$\Rightarrow \Delta a^2 = (a - a^*)^2 = -\frac{1}{\frac{\partial^2 \ln(L)}{\partial a^2}|_{a=a^*}} \tag{5}$$

In physics one usually writes this as: $a = a^* \pm \Delta a$. When this holds one gets Δa at $\ln(L^*) - 0.5$ as shown in Fig. 1. If the Likelihood function is not parabolic one gets asymmetric uncertainties like shown in Fig. 3. Note that the uncertainty interval given by $\ln(L^*) - 0.5$ does not always give a 68% confidence interval. More information on Likelihood fits can be found in Craig Blockers talk about Likelihood Fits at [1].

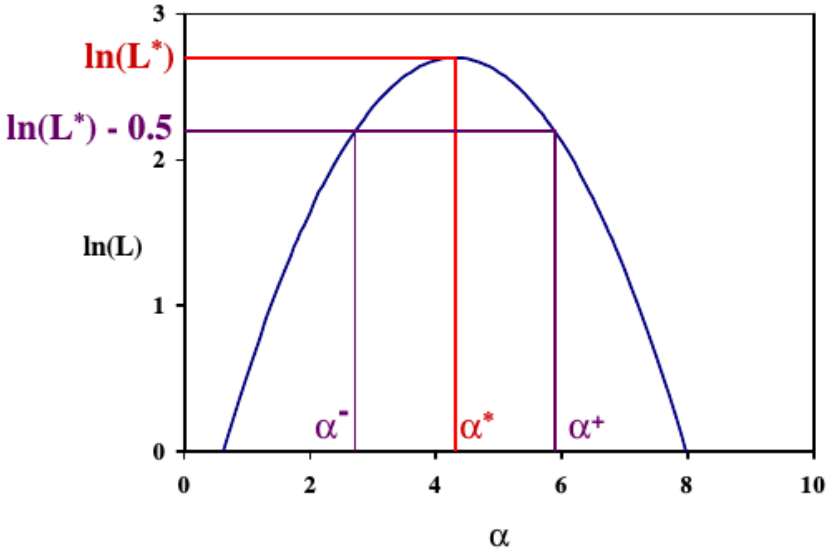


Figure 1: Example of Log-Likelihood function with symmetric uncertainty: $\alpha = \alpha^* \pm \Delta\alpha$. Plot taken from [1].

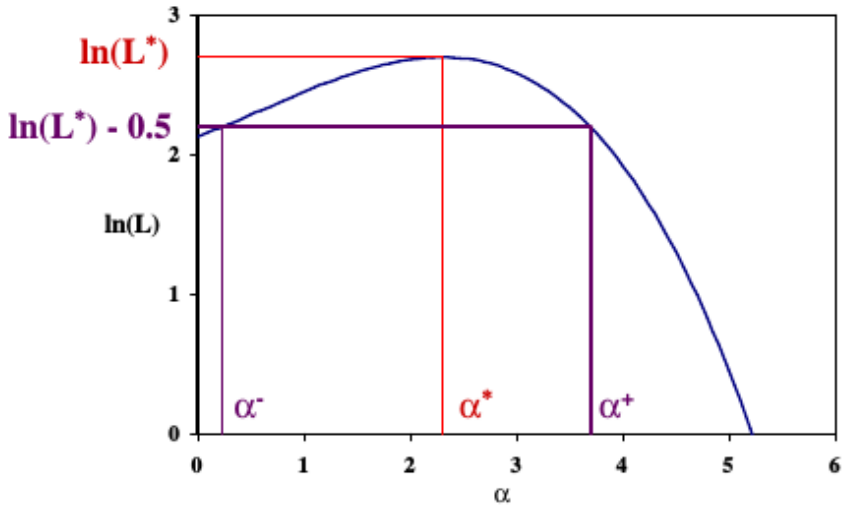


Figure 2: Example of Log-Likelihood function with asymmetric uncertainty: $\alpha = \alpha^{*+\alpha^+}_{-\alpha^-}$. Plot taken from [1].

1.2. Minuit

The Minuit package performs minimization and analysis of the shape of a multi parameter function. It is intended to be used on Chisquare or likelihood functions (see chapter 1.1) for fitting data and finding parameter errors and correlations. It offers several options:

1. MIGRAD
2. HESS

3. MINOS

which will be presented shortly in the next few sections. For further information consider the MINUIT Reference Manual [2]

MIGRAD

MIGRAD is a variable-metric method with inexact line search, a metric updating scheme, and checks for positive-definiteness. Its main weakness is that is depends heavily on knowledge of the first derivatives and fails if they are inaccurate. MIGRAD uses the current estimate of the covariance matrix of the function to determine the current search direction. The search directions is only guaranteed to be downhill if the covariance matrix is positive-definite. If not MIGRAD makes a positive-definite approximation by adding an appropriate constant along the diagonal as determined by the eigenvalues of the matrix. That happens for example if MIGRAD is in a non-physical region or if the problem is under determined or because of numerical inaccuracies.

HESS

HESS is a method in Minuit to estimate symmetric errors of the minimisation. Therefore, HESS calculates the full second-derivative matrix by finite differences and inverts it. If the error matrix is not positive-definite, diagnostics are printed and the algorithm attempts to form a positive-definite approximation, because most algorithms like MIGRAD need a positive-definite "working matrix".

MINOS

MINOS is a method in Minuit to estimate asymmetric errors of the minimisation. Therefore, it follows the function from the minimum up to the point where it crosses the (minimum + UP) value. Where UP is a user defined value typically 0.5 since 0.5 gives a 68% confidence interval in most cases, like shown in figure (3). Notice that Minuit works with a negative log-likelihood while the plot shows a positive log likelihood.

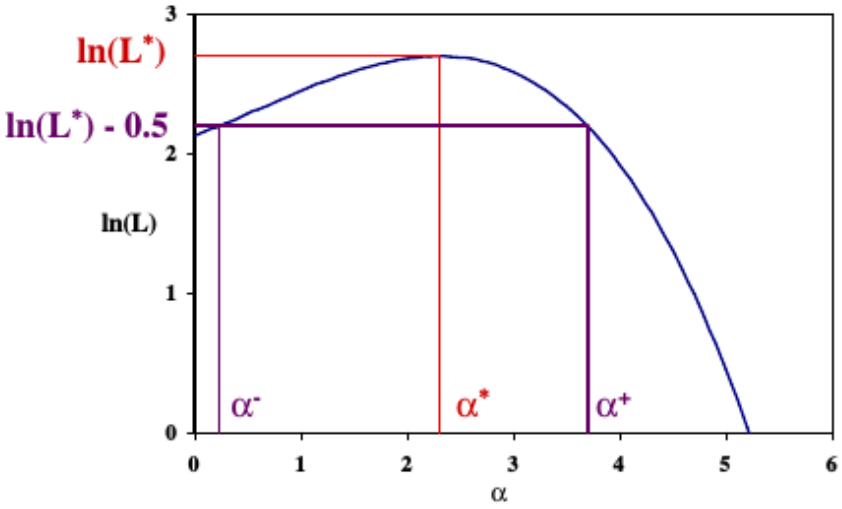


Figure 3: Example of Log-Likelihood function with asymmetric uncertainty: $\alpha = \alpha^{*+\alpha^+}_{-\alpha^-}$. Plot taken from [1].

1.3. Monte Carlo Markov Chain (MCMC)

A Markov Chain is a sequence of random events $X_1, X_2, ...$, where the conditional distribution of X_{n+1} only depends on X_n. The space where the X_i take values is called the state space of the Markov chain, written $x_1, ..., x_n$ If the conditional distribution is independent of n the Markov Chain has stationary transition probabilities. The joint distribution of a Markov chain is determined by:

- The marginal distribution of X_1, called the initial distribution
- The conditional distribution of X_{n+1} called the transition distribution.

If the state space is finite then the initial distribution can be associated with a vector $\lambda = (\lambda_1, ..., \lambda_n)$ defined by the following Probability Pr:

$$Pr(X_i = x_i) = \lambda_i \tag{6}$$

The transition probabilities can be associated with a matrix P having elements p_{ij} defined by:

$$Pr(X_{n+1} = x_j | X_n = x_i) = p_{ij} \tag{7}$$

That only holds if the state space is finite. Most Markov chains of interest in MCMC have an uncountable state space. The initial distribution then becomes an unconditional distribution and the transition probability distribution a conditional probability distribution.

1.3.1. Theory

Suppose one wishes to calculate an expectation

$$\mu = E(g(X)), \tag{8}$$

where g is a real-valued function on the state space. Often there is no exact method that can compute the expectation value. Suppose it is possible to simulate $X_1, X_2, ...$ independent identically distributed having the same distribution as X. Let's define an estimator:

$$\hat{\mu}_n = \frac{1}{n}\Sigma_{i=1}^{n} g(X_i) \tag{9}$$

Introducing the notation $Y_i = g(X_i)$ the Y_i are independent identically distributed with mean μ and variance,

$$\sigma^2 = var(g(X)) \tag{10}$$

The $\hat{\mu}_n$ is the sample mean of the Y_i and according to the Central Limit Theorem

$$\hat{\mu}_n \approx N\left(\mu, \frac{\sigma^2}{n}\right). \tag{11}$$

$$\hat{\sigma}_n^2 = \frac{1}{n}\Sigma_{i=1}^{n}(g(X_i) - \hat{\mu}_n)^2 \tag{12}$$

1.3.2. Implementation

Basic

The most basic implementation of a Markov Chain Monte Carlo program is presented here:

Algorithm 1 basic MCMC pseudo code

```
Initialize x
loop
    Generate random change to x
    Output x
```

It is important that x is the entire state of the program. If for example the movement of a particle is simulated one could apply random changes to the velocity vector at each time step, but if sometimes the changes are not random the process is not Markov any more and all the math described in section 1.3 does not hold any more. The state where random changes are applied to x is called an update mechanism. This report focuses on update mechanisms that preserve a specified distribution. That means the distribution before the update is the same after the update. From that one can construct Markov chains to sample that distribution. An update mechanism is called elementary, when the mechanism is not made up of parts. Since there is not much structure in algorithm 1 most simulation can be fit into this format.

Metropolis-Hastings

Suppose that the specified distribution has unnormalized density h. This means that h is a positive constant multiplied by a probability density. Therefore, h is a non-negative function that integrates (for continuous states) or sums (for discrete states) to a value that is finite and non-zero. The Metropolis-Hastings update does the following:

– The current state x is proposed to move to y with a conditional probability density given x denoted as $q(x, \cdot)$
– Calculate the Hastings ratio

$$r(x, y) = \frac{h(y) \cdot q(y, x)}{h(x) \cdot q(x, y)}. \tag{13}$$

– Accept the proposed move to y with the probability

$$a(x, y) = min(1, r(x, y)). \tag{14}$$

The Hastings ratio (Equation 13) is undefined if $h(x) = 0$, thus one has to ensure that $h(x) > 0$ in the initial state. If $h(y) = 0$ there is no problem since $r(x, y)$ also becomes zero and the probability to accept y just becomes zero. Note that the proposed y must satisfy $q(x, y) > 0$ because $q(x, \cdot)$ is the conditional density of y given x. Hence $h(x) > 0$ the denominator of the Hastings ratio is always non-zero and well defined. The numerator does not have to be nonzero, because if $h(y) = 0$ than y is an impossible value of the desired distribution, and if $q(y, x) = 0$ than x is an impossible proposal when y is the current state.
At this point it is important to stress out that these properties are very fortunate since the Metropolis Hastings update automatically does the right thing, almost surely rejecting such proposals. Therefore, it is not nessecary to ensure that the proposals have to be possible values of the desired distribution. The only thing necessary, is to assure that the unnormalized density function h works with every proposal and gives $h(y) = 0$ if y is an impossible proposal.

1.4. Monte Carlo Markov Chain fitter

Decay rate distributions in particle physics depend on the angles between the final state particles and the energies of the particles. In order to determine the angular coefficients (see section 2.2) of the decay rate the experimentally obtained decay rates are fitted. So far this has been done with the MINUIT [4] algorithm. But as explained in detail in section 1.4, the MINUIT algorithm produces random errors without explanation. Furthermore MINUIT is not transparent: It is not clear, how the fit is performed, and if it really has found the global minimum.
The new algorithm implemented as part of this thesis, is called RooMCMarkovChain. It is based on a Monte Carlo Markov Chain (see section 1.3), where a certain link in the chain respresents a point on the negative log-likelihood function of the fit. The chain is designed to follow the global minimum and to jump out of local minima.
RooMCMarkovChain provides two things: A fitting parameter set, for example the angular coefficents of a decay rate, at the global minimum. And one-dimensional projections of the negative log-likelihood function in the vicinity of the global minimum. They can be used for error estimations and sent along side papers which describe, how the fit has been performed. The RooMCMarkovChain will be part of the roofit package of CERN's ROOT Data analysis framework [3].

The features of the RooMCMarkovChain class will be shown by fitting the following probability density function (pdf):

$$g(x) = \frac{1}{\sqrt{2\pi\sigma_1^2}} e^{-\frac{(x-\mu_1)^2}{2\sigma_1^2}} + f \cdot \frac{1}{\sqrt{2\pi\sigma_2^2}} e^{-\frac{(x-\mu_2)^2}{2\sigma_2^2}} \tag{15}$$

It is so called double gaus pdf, which is just the sum of two gaussian pdfs with a fractional parameter f. Where μ_1 and μ_2 are mean1 and mean2 of the two gaus and σ_1 and σ_2 the standart derviations. The result of the fit is compared with the result of the Minuit algorithm. Therfore 1000 points were simulated fowllowing the distribution 15:

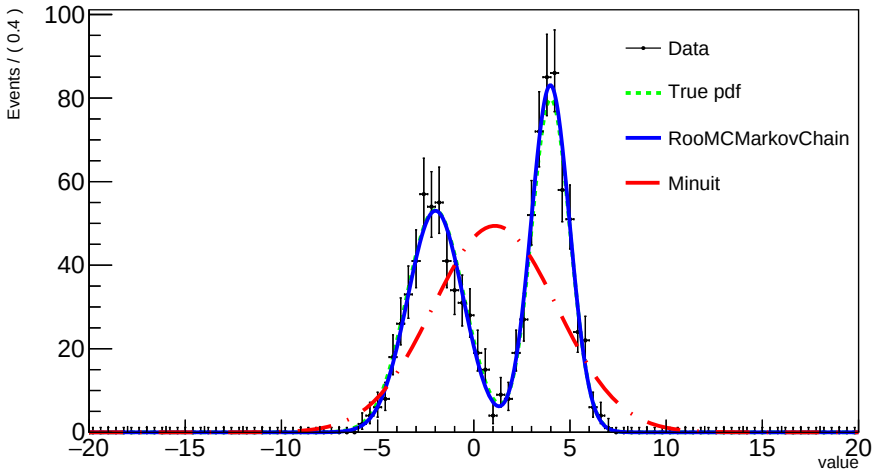


Figure 4: Fit of double gaus pdf (see equation 15) with RooMCMCarkovchain and with Minuit. The true pdf has the follwing values: $\mu_1 = -2$, $\sigma_1 = 1.5$, $\mu_2 = 4$, $\sigma_2 = 1$ and the fraction $f = 0.5$. The Minuit fit fails, because of a unknown error. The fits gets right if the starting values for the parameters are changed.

Figure 4 shows that the Minuit routine fails to fit the distribution. The reasons are unknwon, but the fit gets right if the starting values of the parameters are changed. Intensive testing prooved RooMCMarkovChain less sensitive to starting values. But one should analysis with the number of parameters usually used in particle physics, with up to 50 parameters.
The terminal output, for the fit in figure 4, gives the estimated parameters with error interval and correlation coefficents of each parameter pair.

```
RooFit v3.60 -- Developed by Wouter Verkerke and David Kirkby
                Copyright (C) 2000-2013 NIKHEF, University of California & Stanford University
                All rights reserved, please read http://roofit.sourceforge.net/license.txt

Starting Monte Carlo Markov Chain Fit with 12000 points and cutoff after 7000 points
1% 2% 3% 4% 5% 6% 7% 8% 9% 10% 11% 12% 13% 14% 15% 16% 17% 18% 19% 20% 21% 22% 23% 24% 25% 26%
% 33% 34% 35% 36% 37% 38% 39% 40% 41% 42% 43% 44% 45% 46% 47% 48% 49% 50% 51% 52% 53% 54% 55% 5
 62% 63% 64% 65% 66% 67% 68% 69% 70% 71% 72% 73% 74% 75% 76% 77% 78% 79% 80% 81% 82% 83% 84% 85
91% 92% 93% 94% 95% 96% 97% 98% 99% 100%
NO.     NAME    VALUE           ERROR
1       frac    5.14084e-01     1.47307e-02
2       mean1   3.98243e+00     5.07205e-02
3       mean2    -1.99256e+00   7.51154e-02
4       sigma1  9.98988e-01     3.87089e-02
5       sigma2  1.44724e+00     5.86681e-02

CORRELATION COEFFICIENTS
NO.     1       2       3       4       5
1       1.000   -0.072  -0.043  0.095   -0.060
2       -0.072  1.000   0.131   -0.151  0.136
3       -0.043  0.131   1.000   -0.135  0.194
4       0.095   -0.151  -0.135  1.000   -0.171
5       -0.060  0.136   0.194   -0.171  1.000
```

Figure 5: Terminal output of the RooMCMarkovChain fit. Note that RooMCMarkovChain confused the two means, because for the algorithm the two pdfs are equal.

This gives a good initial overview of results after the fit has finished. The error calculation can be set to assume gaussian or non-gaussian errors.

In addition several other properties of the parameters can be obtainded:

1. The 1 dimensional profile of the negative log likelihood function for a given parameter.

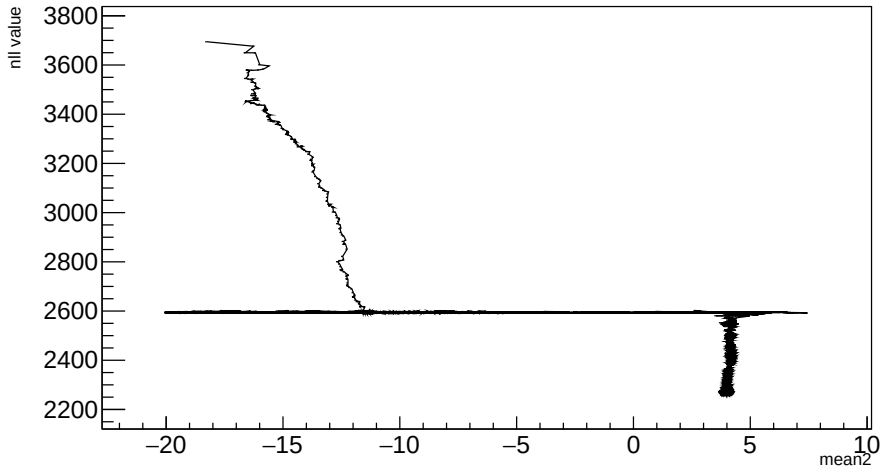


Figure 6: Profile of the negative log likelihood function for $\mu_2 = mean2$ in equation 15. There is a local minimum at the nll value of 2600, since the algorithm scanned that region very briefly.

From this plot on can see how the algorithm reached the minimum of the negative log likelihood function and if there are local minima.

2. The walk distribution of a given parameter.

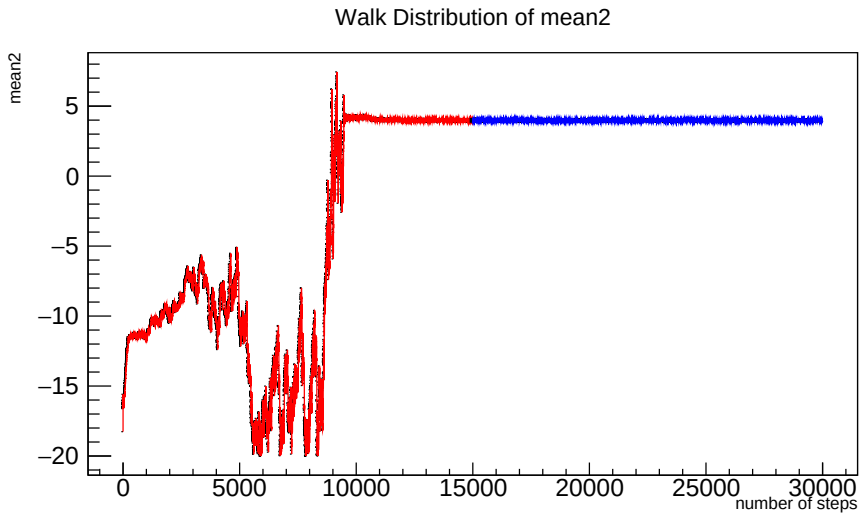


Figure 7: Walk distribution of $\mu_2 = mean2$ in equation 15. The red points are not considered for the error calculation, the number of points beeing cut of is defined by the user.

This plot is very important for handling the RooMCMarkovChain class. Since the error calculation is based on the variance of the walk distribution, cutting of points in the begginning greatly reduces this variance. The user has to choose how many points are cut off. Plotting the walk distribution of all parameters helps choosing the right amount. The right amount would be at the point where there is only the oszillation arround the global minimum left. This has to be done by the user since there is now way to predict how many steps RooMCMarkovChain needs to reach the global minimum.

3. The walk distribution of a given parameter as a histogramm.

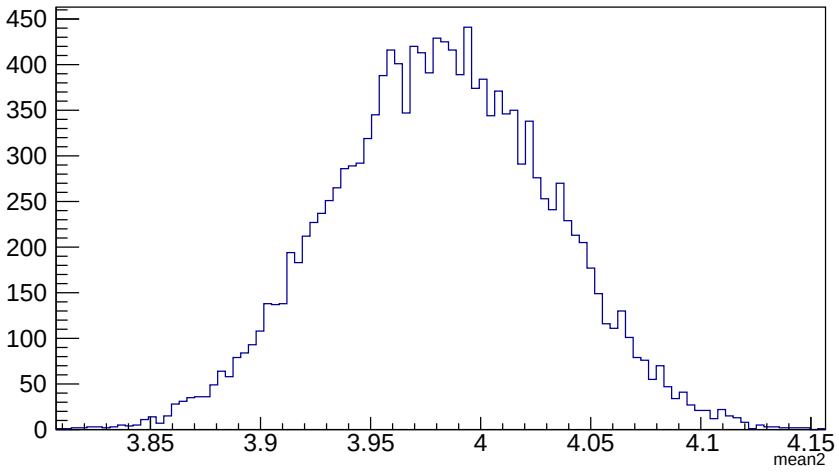


Figure 8: Walk distribution of $\mu_2 = mean2$ in equation 15 as a histogramm.

This plot can be used to check, which error strategy should be used. If the walk distribution histogramm is gaussian, gaussian errors can be assumed.

4. The scatterplot between two given parameters.

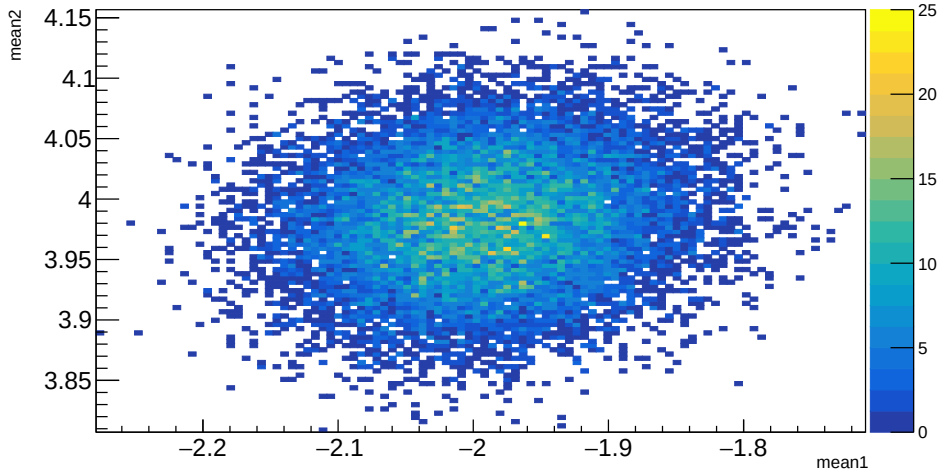


Figure 9: Scatterplot between μ_1 and μ_2 in equation 15. The round shape states that there is no correlation between the two parameters.

This plot shows the correlation between two parameters graphically.

2. The $B \to K^* \mu \mu$ decay

This section describes the motivation to analyse this decay, then the operators needed to calculate the different couplings are derived and in the last section the effetive Wilson coefficients are derived. This section is mostly theoretical and should give an overview over the decay.

2.1. Motivation

According to the Standard Model the three leptons electron (e), muon (μ), and tau (τ), differ only in their masses. Therefore, in high energy regions (> 1 TeV) where masses become negligible, these leptons should behave the same. This phenomena is called lepton universality.

Recent experimental results of the LHCb collaboration [5] suggest a violation of the lepton universality: They analysed the decay of the B meson with a Kaon and 2 muons in the final state, and the decay of the B with a Kaon and 2 electrons in the final state.

The B0 is a pseudoscalar meson made of an anti-b and a d-quark. K_0^* (892) is a vector meson made of an anti-s and a d-quark which promptly decays to a charged kaon and a pion.

The LHCb collaboration measured the fractions of four B0 decays with a b to s quark transition and two leptons in the final state:

- $B_0 \to K_0^* \mu^+ \mu^-$
- $B_0 \to K_0^* J/\psi (\to \mu^+ \mu^-)$
- $B_0 \to K_0^* e^+ e^-$

– $B_0 \to K_0^* J/\psi(\to e^+e^-)$

The following double ratio of these branching fractions are considered as a well defined test of lepton universaility and reduce systematic uncertainties.

$$R_{K*0} = \frac{\mathcal{B}(B^0 \to K^{*0}\mu^+\mu^-)}{\mathcal{B}(B^0 \to K^{*0}J/\psi(\to \mu^+\mu^-))} \Big/ \frac{\mathcal{B}(B^0 \to K^{*0}e^+e^-)}{\mathcal{B}(B^0 \to K^{*0}J/\psi(\to e^+e^-))} \,,$$
$$R_{K*0} = \begin{cases} 0.66^{+0.11}_{-0.07}(\text{stat}) \pm 0.03(\text{syst}) \text{for} 0.045 < q^2 < 1.1\,\text{GeV}^2/c^4, \\ 0.69^{+0.11}_{-0.07}(\text{stat}) \pm 0.05(\text{syst}) \text{for} 1.1 < q^2 < 6.0\,\text{GeV}^2/c^4. \end{cases} \tag{16}$$

Where q^2 is square of the invariant di-muon mass. The Standard Model predicts this ratio R_{K*0} to be one. But LHCb found values for R_{K*0} below one, implying that decays into electrons are favored with respect to decays into muons. The mesurement shows a 2.1-2.3 and 2.4-2.5 σ deviation in the two q^2 regions, respectively. To investigate further the same measurment is performed with new data from the LHCb detector. The goal of this thesis is to contribute to this analysis.

2.2. Kinematics

In this section the decay itself and its kinematics are explained: The Decay is a flavor changing neutral current (FCNC) with four charged particles in the final state. The FCNC is a current, which changes the flavor of a fermion without changing its electric charge. The four particles in the final stage are:
The K^+ and π^- from the K^* decay and two leptons from the loop or box diagrams:

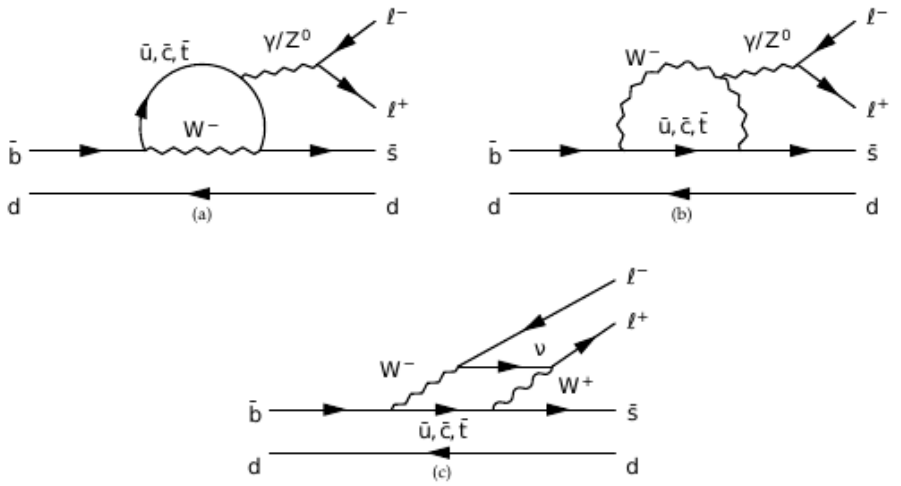


Figure 10: Feynman diagrams for decay $B(\bar{b}, d) \to K^*(\bar{s}, d) l^+ l^-$ at lowest order

The kinematics and directions of the final state particles from the decay are defined by the three angels θ_K, θ_L and ϕ, shown in figure 11 and the invariant di-muon mass square q^2.

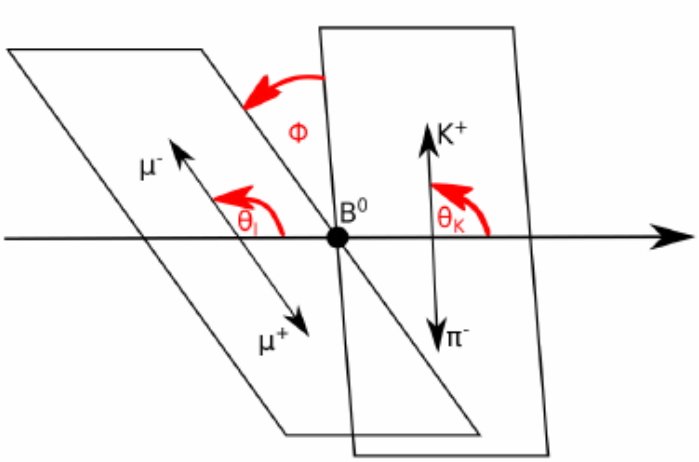


Figure 11: Angels defining the decay $B^0 \to K^{*0} \mu \mu$

The differential decay rate for the B^0 Meson can be expressed in terms of these variables:

$$\begin{aligned}
\frac{d^4\Gamma}{d\cos\theta_L d\cos\theta_K d\phi dq^2} &= \frac{9}{32\pi} I(q^2, \theta_L, \theta_K, \phi) \\
\text{with: } I(q^2, \theta_L, \theta_K, \phi) &= I_1^S \sin^2(\theta_K) + I_1^C \cos^2\theta_K + \left(I_2^S \sin^2\theta_K + I_2^C \cos^2\theta_K\right) \cos^2\theta_L \\
&\quad + I_3 \sin^2\theta_K \sin^2\theta_L \cos 2\phi + I_4 \sin 2\theta_K \sin 2\theta_L \cos\phi \\
&\quad + I_5 \sin 2\theta_K \sin\theta_L \cos\phi \\
&\quad + \left(I_6^S \sin^2\theta_K + I_6^C \cos^2\theta_K\right) \cos\theta_L + I_7 \sin 2\theta_K \sin\theta_L \sin\phi \\
&\quad + I_8 \sin 2\theta_K \sin 2\theta_L \sin\phi + I_9 \sin^2\theta_K \sin^2\theta_L \sin 2\phi
\end{aligned} \tag{17}$$

This decay rate is defined as the probability per unit time that the particle will decay. The I_i can be obtained by fitting this decay rate to the kinematic variables obtained by the detector. This fit is usually done by a standart fitting routine called MINUIT [4]. In section 1.4 an alternative is presented.
For completness the next two sections desribe some of the theory concerning the decay: The operators and wilson coefficients.

2.3. Operators for $B \to X_s l^+ l^-$ decays

In this section all the operators of the B0 decay into a Meson containing a strange quark X_s and two charged leptons l^+, l^- are derived from the effective Lagrangian. These operators are used to describe the creation or destruction of particles by applying them to a quantum field. This section is supposed to give an overview. More detailed information can be found in ref [6].

The effective Lagrangian for $B \to X_s l^+ l^-$ decays has the form:

$$\begin{aligned}\mathcal{L}_{eff} &= \mathcal{L}_{QCD,QED}(u,d,s,c,b,e,\mu,\tau) \\ &+ \frac{4G_F}{\sqrt{(2)}} \left[V_{us}^* V_{ub}(C_1^c P_1^u + C_2^c P_2^u) + V_{cs}^* V_{cb}(C_1^c P_1^c + C_2^c P_2^c)\right] \\ &+ \frac{4G_F}{\sqrt{(2)}} \sum_{i=3}^{10} \left[(V_{us}^* V_{ub} + V_{cs}^* V_{cb}) C_i^c + V_{ts}^* V_{tb} C_i^t\right] P_i.\end{aligned} \tag{18}$$

The first term in equation 18 contains the kinetic terms of the light SM particles as well as their QCD and QED interactions. The remaining two terms consist of $\Delta B = -\Delta S = 1$ local operators of dimension $(d \leq 6)$, which contain those light fields. The mass of the s quark can be negleted in comparisson with the b mass. One gets the following operators:

$$\begin{aligned}
\mathcal{O}_1^u &= (\bar{s}_L \gamma_\mu T^a u_L)(\bar{u}_L \gamma^\mu T^a b_L), & \mathcal{O}_6 &= (\bar{s}_L \gamma_{\mu_1}\gamma_{\mu_2}\gamma_{\mu_3} T^a b_L) \sum_q (\bar{q}\gamma^{\mu_1}\gamma^{\mu_1}\gamma^{\mu_1} T^a q), \\
\mathcal{O}_2^u &= (\bar{s}_L \gamma_\mu u_L)(\bar{u}_L \gamma^\mu b_L), & & \\
\mathcal{O}_1^c &= (\bar{s}_L \gamma_\mu T^a c_L)(\bar{c}_L \gamma^\mu T^a b_L), g & \mathcal{O}_7 &= \frac{e}{g^2} m_b (\bar{s}_L \sigma^{\mu\nu} b_R) F_{\mu\nu}, g \\
\mathcal{O}_2^c &= (\bar{s}_L \gamma_\mu c_L)(\bar{c}_L \gamma^\mu b_L), & \mathcal{O}_8 &= \frac{1}{g} m_b (\bar{s}_L \sigma^{\mu\nu} T^a b_R) G^a_{\mu\nu}, \\
\mathcal{O}_3 &= (\bar{s}_L \gamma_\mu b_L) \sum_q (\bar{q}\gamma^\mu q), & & \\
\mathcal{O}_4 &= (\bar{s}_L \gamma_\mu T^a b_L) \sum_q (\bar{q}\gamma^\mu T^a q), & \mathcal{O}_9 &= \frac{e^2}{g^2} (\bar{s}_L \gamma_\mu b_L) \sum_l (\bar{l}\gamma^\mu l), \\
\mathcal{O}_5 &= (\bar{s}_L \gamma_{\mu_1}\gamma_{\mu_2}\gamma_{\mu_3} b_L) \sum_q (\bar{q}\gamma^{\mu_1}\gamma^{\mu_1}\gamma^{\mu_1} q), & \mathcal{O}_{10} &= \frac{e^2}{g^2} (\bar{s}_L \gamma_\mu b_L) \sum_l (\bar{l}\gamma^\mu \gamma_5 l),
\end{aligned} \tag{19}$$

where $\sum_q$ and $\sum_l$ denote the sums over light quarks and all leptons, respectivly.

2.4. Wilson coefficients

In this section the Wilson coefficents are derived from the effective Hamiltonion of the deacy. The effective Hamiltonian for $b \to s\mu^+\mu^-$ transitions can be written as:

$$H_{eff} = -\frac{4G_F}{\sqrt{2}} \left(\lambda_t H_{eff}^{(t)} + \lambda_u H_{eff}^{(u)}\right) \tag{20}$$

The λ_i can be expressed with CKM combinations $\lambda_i = V_{ib} V_{is}^*$.

$$H_{eff}^{(t)} = C_1 \mathcal{O}_1^c + C_2 \mathcal{O}_2^C + \sum_{i=3}^{6} C_i \mathcal{O}_i + \sum_{i=7,8,9,10,P,S} (C_i \mathcal{O}_i + C_i' \mathcal{O}_i') \tag{21}$$

$$H_{eff}^{(u)} = C_1(\mathcal{O}_1^C - \mathcal{O}_1^u) + C_2(\mathcal{O}_2^C - \mathcal{O}_2^u). \tag{22}$$

The contribution of $H_{eff}^{(u)}$ has a double Cabibbo supression and is therefore usually dropped. It is kept here since it is sensitive to complex phases of decay amplitudes. The operators $\mathcal{O}_{i \leq 6}$ are the same as for general $B \to X_s l^+ l^-$ decays, see equation 19. The remaining ones are given by:

$$\begin{aligned}
\mathcal{O}_7 &= \frac{e}{g^2} m_b (\bar{s}\sigma_{\mu\nu} P_R b) F^{\mu\nu}, & \mathcal{O}'_7 &= \frac{e}{g^2} m_b (\bar{s}\sigma_{\mu\nu} P_L b) F^{\mu\nu}, \\
\mathcal{O}_8 &= \frac{1}{g} m_b (\bar{s}\sigma_{\mu\nu} T^a P_R b) G^{\mu\nu a}, & \mathcal{O}'_8 &= \frac{1}{g} m_b (\bar{s}\sigma_{\mu\nu} T^a P_L b) G^{\mu\nu a}, \\
\mathcal{O}_9 &= \frac{e^2}{g^2} (\bar{s}\sigma_\mu P_L b)(\bar{\mu}\gamma^\mu \mu), & \mathcal{O}'_9 &= \frac{e^2}{g^2} (\bar{s}\gamma_\mu P_R b)(\bar{\mu}\gamma^\mu \mu), \\
\mathcal{O}_{10} &= \frac{e^2}{g^2} (\bar{s}\gamma_m u P_L b)(\bar{\mu}\gamma^\mu \gamma_5 \mu), & \mathcal{O}'_{10} &= \frac{e^2}{g^2} (\bar{s}\gamma_\mu P_R b)(\bar{\mu}\gamma^\mu \gamma_5 \mu), \\
\mathcal{O}_S &= \frac{e^2}{16\pi^2} m_b (\bar{s} P_R b)(\bar{\mu}\mu), & \mathcal{O}'_S &= \frac{e^2}{16\pi^2} m_b (\bar{s} P_L b)(\bar{\mu}\mu), \\
\mathcal{O}_P &= \frac{e^2}{16\pi^2} m_b (\bar{s} P_R b)(\bar{\mu}\gamma_5 \mu), & \mathcal{O}'_P &= \frac{e^2}{16\pi^2} m_b (\bar{s} P_L b)(\bar{\mu}\gamma_5 \mu),
\end{aligned} \tag{23}$$

where m_b denotes the running b mass in the $\overline{MS}$ scheme and g is the strong coupling constant and $P_{L,R} = (1 \pm \gamma_5)/2$. In the Standart Modell the primed Operators with opposite chirality to the unprimed operators vanish or are highly suppresd as are the $\mathcal{O}_S$ and $\mathcal{O}_P$. The contributions of $\mathcal{O}_{1,2,3,4,5,6}$ are neglected, since they are either heavily constrained or their impact turns out to be generically very small. For example in the left-right symmetric models or throughout gluino contributions in a general Minimal Supersymmetric Standard Model.

The C_i coefficients in the equations 21 and 22 are called Wilson coefficients. They encode short-distance physics and New Physics effects. For the calculation a matching scale $\mu = m_W$ is chosen, in a pertubative expansion in powers of $\alpha_s(m_W)$. Then the Wilson coefficents are evolved down to scales $\mu = m_b$ according to the solutions of the renomalization group equations. Contributions by New Physics enter through $C_i(m_W)$, while the low scales are determined by the Standart Modell. To allow a more organized expansion of the Wilson coefficients in pertubation theory the factors $16\pi^2/g^2 = 4\pi/\alpha_S$ are included into the definitions of the operators $\mathcal{O}_{i\geq 7}$. All the C_i expand as:

$$C_i = C_i^{(0)} + \frac{\alpha_s}{4\pi} C_i^{(1)} + \left(\frac{\alpha_s}{4\pi}\right)^2 C_i^{(2)} + O(\alpha_s^3) \tag{24}$$

where $C_i^{(0)}$ is the tree-level contribution, which is quale to zero for all operators except $\mathcal{O}_2$ and $C_i^{(n)}$ denotes the n-loop contributions. Before discussing the Wilson coefficents in details, lets look at the Operators again; the operators $\mathcal{O}'_S$ and $\mathcal{O}'_P$ are given in terms of conserved currents. They carry no scale-dependence. They do not mix with other operators and their Wilson coefficents are at the matching scale. $\mathcal{O}_9$ is also given by conserved curents. It mixes with $\mathcal{O}_{1,2,3,4,5,6}$ via a virtual photon decaying into $\mu^+\mu^-$. In addition their is a scale depedence from the factor $1/g^2$. This dependence is also present in C_{10} which otherwise would be scale independent.

C_7 and C_9 always appear in a particular combination with other Wilson coefficents in matrix elements. Therfore effective coefficients are defined:

$$\begin{aligned}
C_7^{eff} &= \frac{4\pi}{\alpha_s} C_7 - \frac{1}{3} C_3 - \frac{4}{9} C_4 - \frac{20}{3} C_5 - \frac{80}{9} C_6, \\
C_8^{eff} &= \frac{4\pi}{\alpha_s} C_8 + C_3 - \frac{1}{6} + 20 C_5 - \frac{10}{3} C_6, \\
C_9^{eff} &= \frac{4\pi}{\alpha_s} C_9 + \mathcal{Y}(q^2), \\
C_{10}^{eff} &= \frac{4\pi}{\alpha_s} C_{10}, \\
C_{7,8,9,10}^{\prime eff} &= \frac{4\pi}{\alpha_s} C'_{7,8,9,10},
\end{aligned} \tag{25}$$

$$\begin{aligned} \text{where } \mathcal{Y}(q^2) = & h(q^2, m_c)\left(\frac{4}{3}C_1 + C_2 + 6C_3 + 60C_5\right) \\ & - \frac{1}{2}h(q^2, m_b)\left(7C_3 + \frac{4}{3}C_4 + 76C_5 + \frac{64}{3}C_6\right)_{env} \\ & - \frac{1}{2}h(q^2, 0)\left(C_3 + \frac{4}{3}C_4 + 16C_5 + \frac{64}{3}C_6\right) \\ & + \frac{4}{3}C_3 + \frac{64}{9}C_5 + \frac{64}{27}C_6. \end{aligned} \tag{26}$$

The function $h(q^2, m_q)$ comes from the fermion loop and for completness is presented in equation 27 below. More details are explained in reference [7].

$$\begin{aligned} h(q^2, m_q) &= -\frac{4}{9}\left(\ln\frac{m_q^2}{\mu^2} - \frac{2}{3} - z\right) - \frac{4}{9}(2+z)\sqrt{|z-1|} \cdot \begin{cases} \arctan\frac{1}{\sqrt{z-1}} & z > 1 \\ \ln\frac{1+\sqrt{1-z}}{\sqrt{z}} - \frac{i\pi}{2} & z \leq 1 \end{cases} \\ z &= \frac{4m_q^2}{q^2} \end{aligned} \tag{27}$$

2.5. The LHCb Experiment

The Large Hadron Collider beauty experiment (LHCb) is one of four large experiments based at the CERN laboratory near Geneva in Switzerland. In this section the experimental setup of the detector is presented, which recorded the data used in this thesis. The LHCb Experiment is situated at the Large Hadron Collider (LHC). The LHC consists of a 27-kilometre ring of superconducting magnets and is located in an underground tunnel at CERN. The protons in the LHC are accelerated to have a kinetic energy of 7 TeV, which allows a collision energy in the LHCb detector of 13 TeV. In the year 2016 the LHCb had a recorded luminosity of $1906\,\text{pb}^{-1}$. For this thesis 1'575'210 potential $B \to K^* \mu\mu$ events are used and refered as raw LHCb data. LHCb is dedicated to flavor physics. It investigates rare decays and CP violation in beauty and charm hadrons.

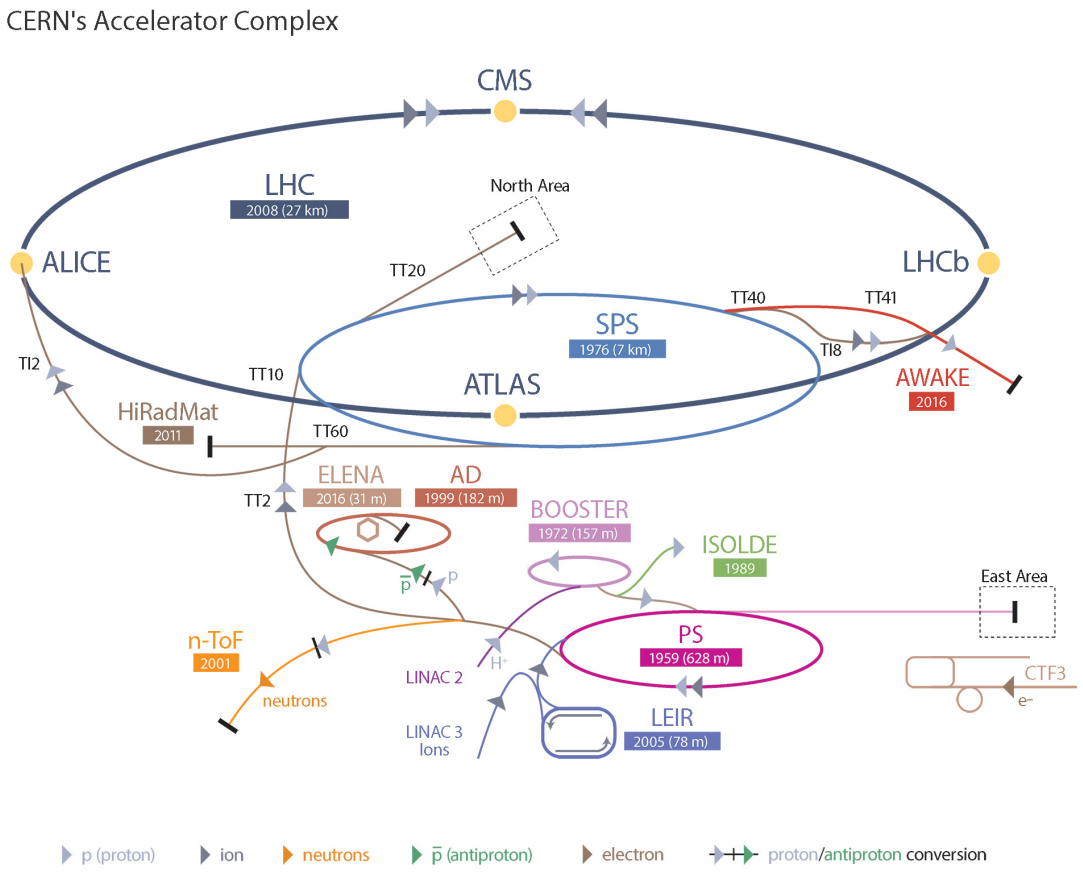


Figure 12: CERN's Accelerator Complex [8]: he protons are injected in the linear accelerator LINAC2. Then they are pre-accelerated in 3 synchrotons (BOOSTER,PS,SPS), where the protons reach a kinetic energy of 450 GeV. This is their entering energy of the LHC which accelerates them further up to 7 TeV. They collide inside the four detectors: CMS, ATLAS, LHCb and ALICE.

2.5.1. The LHCb Detector

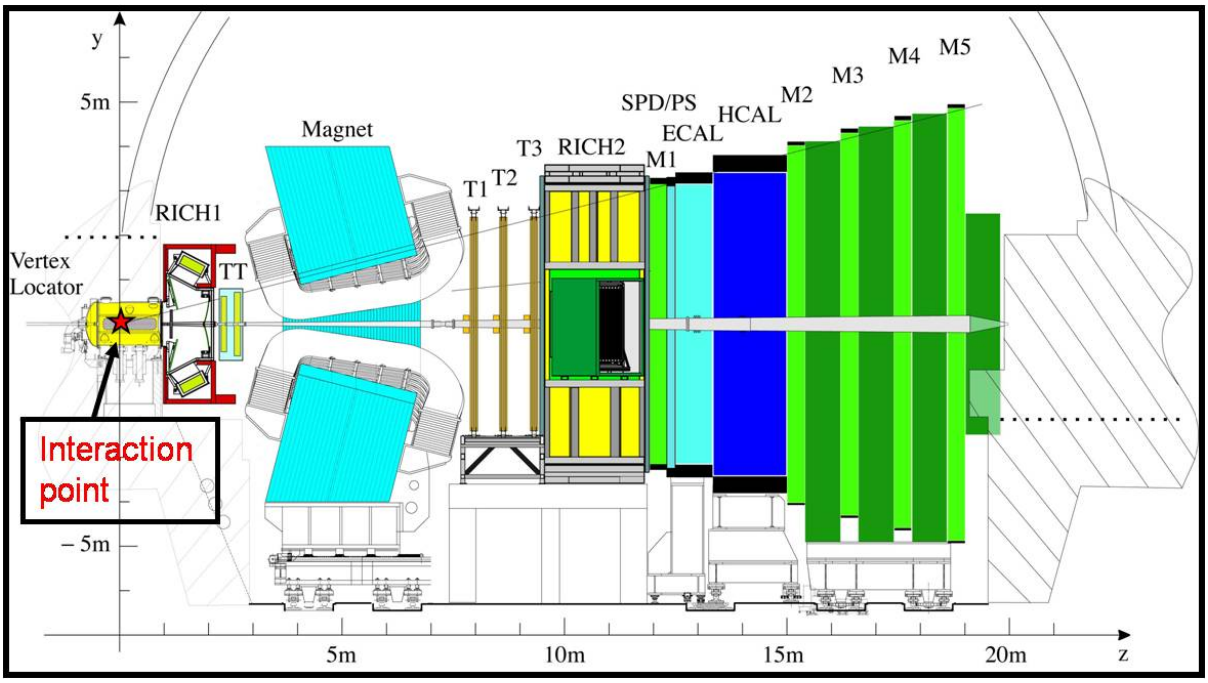


Figure 13: Basic layout of the LHCb detector [9]. The interaction point is inside the vertex detector and the beam pipe passes through the center. The different sub-detectors are the two Ring Imaging Cherenkov Detectors (RICH1 and RICH2), the tracking stations (TT and T1 to T3), the Scintillator Pad Detector (SPD), the pre-shower Electromagnetic CALorimeter (ECAL), the Hadronic CALorimeter (HCAL) and the Muon stations (M1-M5).

VErtex LOcator (VELO) [10] : picks out B mesons from the multitude of other particles produced. This is a complex task since B mesons have a very short lifetime spent close to the beam. The VELO's silicon detector elements must be placed at a distance of just five millimeters to the interaction point. To prevent damage to the detector during beam injection, it is mechanically moved to a safe distance. VELO measures B mesons indirectly by detecting its decay particles.It has a resolution of 10 microns.
Ring Imaging Cherenkov (RICH) detectors [11] : The RICH detectors measure the emission of Cherenkov radiation, which happens when a charged particle passes through a medium faster than light does. The shape of the light cone depends on the particle's velocity, enabling the detector to determine its speed.
Magnet [12] : The big magnet of the LHCb experiment weights 27 tons and is mounted inside a 1,450 tons steel frame. This powerful magnet forces the charged particles to change their trajectory. From the curvature of their paths, their momentum is calculated.
Trackers [13] : The LHCb's tracking system consists of a series of four large rectangular stations, each covering an area of 40 m^2. While flying through this area charged particles will leave signals, which are used to estimate their trajectory. The trajectory is used to link the signals left in other detector elements to the corresponding particle. In the LHCb experiment two different tracker technologies are used: The silicon tracker placed close to the beam pipe uses silicon micro-stripes. If a charged particle passes such a stripe, it collides with the silicon atoms, liberating electrons and creating an electric current, which is then recorded. The outer tracker situated further from the beam pipe consists of gas-filled tubes. The gas is ionized when a charged particle hits the gas molecules, producing electrons. They reach an anode

wire situated in the centre of each tube. The position of the track is found by timing: how long electrons need to reach it.

Calorimeters [14] : Calorimeters stop particles as they pass through, measuring the amount of energy lost. In LHCb there are two different types: The elctromagnetic calorimeter responsible for light particles like electrons and photons and the hadronic calorimeter responsible for heavier particles containing quarks. Both have a sandwich-like structure with alternating layers of metal and plastic plates. If a particles hits a metal plate it produces a shower of secondary particles. These will excite polystyrene molecules in the plastic plates, which then emit ultraviolet light. The energy lost by the particle in the metal plate is proportional to the amount of UV light produced in the plastic plates.

Muon System [15] : The Muon System consists of 5 rectangular stations, which cover an area of 435 m^2. Each station has chambers filled with three gases: carbon dioxide, argon and tetrafluoromethane. Passing muons react with the mixture, and electrodes detect the result.

2.5.2. The LHCb trigger system

The rate of events at the LHCb interaction point is 40 MHz; the rate to have a B meson contained in the detector is only 15 kHz. But the offline computing power just allows 2 kHz to be recorded. The LHCb trigger system aims to fill this 2 kHz with interesting B decays and important control decays like J/ψ decays. [22]. The trigger has two levels:

The **Level Zero (L0)** trigger reduces the beginning 40MHz to 1 MHz. To get this high rate it can only rely on fast sub-detectors as the calorimeters and the muon system. The L0 trigger looks for events with high transverse momentum with respect to the particle beam axis (pT),. Particles from a B decay have this attribute, since B Mesons are always produced almost parallel to the beam axis. In addition the L0 trigger performs a simplified vertex reconstruction using the signals of two silicon layers of the VELO in order to identify events with multiple proton-proton collisions. They are rejected because for this kind of events it is much more difficult to reconstruct B meson decays.

The **High Level Trigger (HTL)** s an algorithm that runs on a farm of 1000 16-core computers. It has two stages: HLT1 which reduces the event rate to a few tens of kHz and HLT2 which reduces the rate to the 2 kHz which are recorded. HLT1 gets all the candidates of the L0 trigger and uses the full detector information on them to search for particles with a high impact parameter with respect to the proton-proton collisions. These particles are most likely decay products from B mesons, because of its relatively long life-time. They typically fly 1 cm away from the collision point before decaying resulting in a high impact parameter for the decay products. HLT2 does a complete reconstruction of the events. It starts with the track of the VELO and connects them to the tracks in the other sub-detectors. Most important are displaced vertices, since they are strong indicator for B decays. The selection is divided into two parts. The inclusive selection searches for resonance decays like D^* or J/ψ. The exclusive selection is designed to provide the highest possible efficiency to fully reconstruct B decays of interest. It uses all information available such as mass and vertex quality and intermediate resonances.

2.6. Selection

2.6.1. Inroduction to decision tree learning

The selection of events out of the raw detector data is done by boost decision trees. A decision tree takes a set of input features and splits input data recursively based on those features. Each split at a node is chosen to maximize information gain or minimize entropy. The infromation gained by the split is the difference in entropy befor a and after the split. Entropy is maximal for a 50:50 split in minimal for a 1:0 split. The splits are created recursively and splitting continues until some stop condition is met, like a certain depth of the tree or if no more information is gained. An example of a decision tree is given in figure 14.

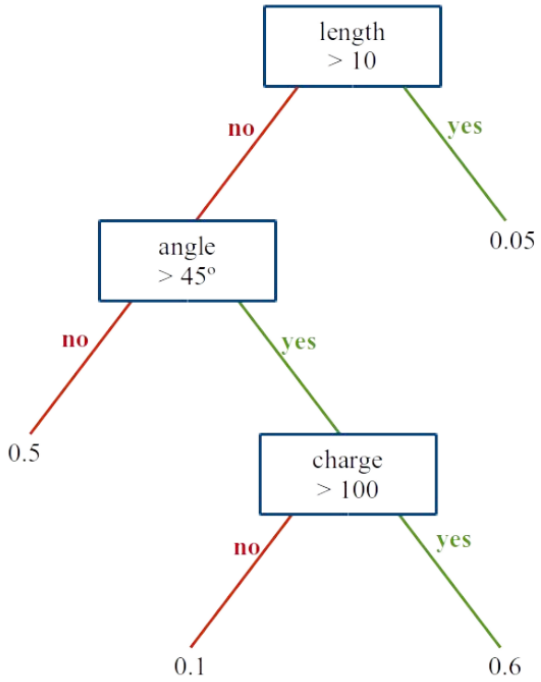


Figure 14: Example of a decision tree. It is structured in Nodes (length, angle, charge) which represent the input features and in Leaves, which can be seen as terminal nodes. They represent a class label, a probability to be signal for example.

The technique used in this thesis is called tree boosting. It is an advancement of the decision tree method explained above. Tree boosting combines many week tees into a strong boost decision tree. For that matter each tree is created iteratively and the tree's output ($h(x)$) is given in a weicht (w) relative to its accuracy. Th output of a boost decision tree is a weighted sum:

$$\hat{y}(x) = \sum_t w_t h_t(x) \tag{28}$$

After each iteration each data sample is given a weight based on its misclassification. Misclassification means that the decision tree found the wrong answer. For example it gets the atributes of a muon as input und decides that it is an electron. The more often a data sample is misclassifed, the more important it becomes since the weights accumulate. The goal of tree bossting is the minimization the

objective function:

$$O(x) = \sum_i l(\hat{y}_i, y_i) + \sum_t \Omega(f_t) \tag{29}$$

where $l(\hat{y}_i, y_i)$ is the loss function, it represents the distance between truth and prediction of the i th data sample. $\Omega(f_t)$ is the regulariization function, it penalizes the complexity of the t th tree.
There are many different ways to archive boosting, or in other words to iteratively adding trees together to minimize the loss function. The ones used in this thesis are described in the following references: Ada Boost [16], uBoost [18], xgb [19], sk_bdtg [20], sk_bdt [21].

2.6.2. Test of different boost decision trees

The first step of analysis in particle physics is to is to distinguish the one decay one wants to analyze from everything else. Here the signal produced by the decay $B \to K^* \mu\mu$ shall be separated from everything else detected and recorded in the experiment. In order to find this decay in the raw data of a detector, one needs thresholds on experimental parameters. These parameters shall be different or in the best case unique to the decay. They can be all kind of parameters like vertex locations, momenta or angles between trajectories. For the $B \to K^* \mu\mu$ decay these parameters would be:

- Decay vertex location for reconstructed particles (ENDVERTEX)
- Primary vertex location (OWNPV)
- Impact parameter (IP_OWNPV)
- Flight distance (FD_OWNPV)
- The cosine of the angle between primary vertex and decay vertex and recorded momentum (DIRA_OWNPV)

The names in the bracets are the labels given to these parameters in the data structure of the ROOT Data Anlysis Framework [3]. Then the thresholds on these parameters must be defined, which define the signal-region. For example the flight distance has to be between 0.2 milimeters and 2 centimeters. To obtain these thresholds, boost decision trees are used. Boost decision trees are a type of algorithm from the Machine Learning area. They can be trained to distinguish different types of data. In this case signal data and background data. Finally, each event in the data gets a probability assigned to belong to the signal events. So after the selection one can just take all the events with a probability over lets say 80%. One has now eliminated 80% of the background and can continue cleaning up the signal but thats not a part of this thesis. The goal of this part was to find the best boost decision tree to be used in the $B \to K^* \mu \mu$ decay analysis.
Seven different boost decision trees were tested. The boost decision tree is given a training set consisting of Monte Carlo simulated events containing only $B \to K^* \mu \mu$ decays and randomly choosen real detector data, containing all kinds of decays. In a training set the data is labeled. The Monte Carlo events are labeld with 1 for signal and the real data with 0 for background. The boost decision tree will now train, that means the algorithm tries to label the data into signal and background by choosing different thresholds on the parameters mentioned above. After the training the thresholds get fixed. Now one has to test the boost decision tree to check if everything worked fine. To do so a second set of data is prepared similar to the training set, just that the labels are now hidden from the tree. Since boost decision trees are not perfect, they do not assign 0 and 1 to each event but rather a probability to be signal between 0 and 1. A good boost decision tree will now label the signal events with a high probability and background with a low probabillity. A bad boost decision tree will assign 0.5 probability to each event. With a 0.5 probability no information was gained by the selection, because having a 50:50 chance is as good as guessing. In case of succesfull selection this probability assignements can be used

as a threshold themselves to subtract the background.
Another property of boost decision trees is, that the data used in the training, can no longer be used in the analysis, because the boost decision tree knows the training set too well and a bias is introduced if one reuses the training data in the actual analysis. That would mean that the part of the real detector data which is needed to train the boost decision tree would be lost for further analysis. To avoid such a waste of data a technique called k-folding is used.
K-folding means that the no training set is created. But the data Monte Carlo Mix with signal events labeled with 1 and background events labeled with 0 is split into k equal parts, containing randomly choosen events. Now to perform the selection in one part all the other parts are used for training, while the labels of the one part are of course hidden. After iterating over all parts, one has performed a selection on all the data without loosing any. The only disadvantage are the addidtional computer resources needed, since the boost decision tree has now to be trained k times instead of one time.

The following list of boost decsision trees where tested and compared in terms of perfomance:

- Ada Boost [16]
- uGB [17] + knnAda (k-nearest neghbor AdaBoost)
- uBoost [18]
- uGB [17] + Fl (flatness loss)
- xgb [19]
- sk_bdtg [20]
- sk_bdt [21]

The test was performed with 30000 events from the 2016 LHCb $B \to K^* \mu \mu$ data and 10000 events from the Monte Carlo simulation. To compare boost decision trees the so called ROC (receiver operating charactertic) curves are used. The ROC curve shows the true positive rate against the false positive rate. The true positive rate is the rate of signal events that have been selected correctly as signal events, while the false postive rate ist the rate of background events selected as signal events. On figure 15 a ROC curves of the boost decision trees above is displayed. The data used for training the boost decsison tree algorithms can not be reused in the analysis, because the algorithm knows the data to well. To avoid a waste of data the sample of 40000 events to test the boost decision trees was cut into ten equal parts, also called folds. Then to perform the selection in one fold, the oder nine folds were used for training. In figure 15 the ROC curve of one of these folds is displayed, the other nine can be found in the appendix A.
It turns out that all the boosted decision trees selected the data mostly correctly with just some minor variances. The ROC curve is in that case not a good tool to compare the different boost decsison trees.

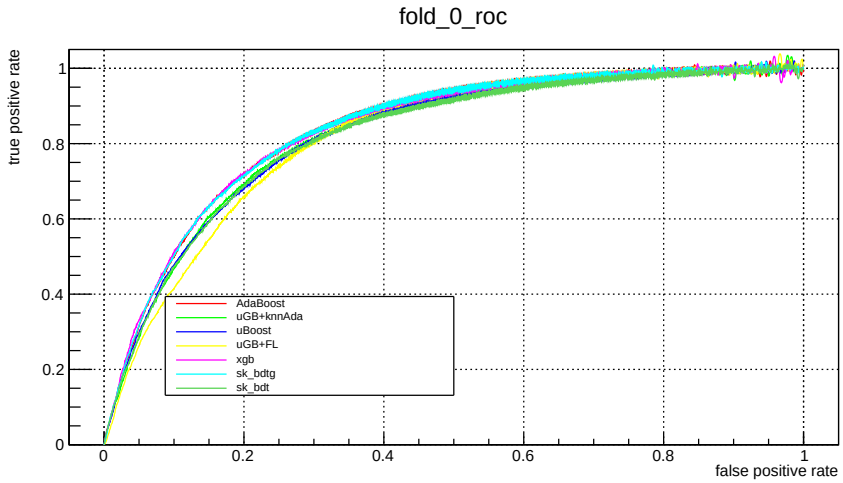


Figure 15: ROC curve of the first fold. One can see that all the boost decision trees are competetive in terms of selecting signal events correctly

The next step is to check for correlations between the assigned probabilities and the kinematic variables and the angels between trajectories. The kinematic variabels and angels as explained in section 2.2 are used to obtain the angular coefficients of the decay rate. If there is a correlation between the assigned probabilities from the boost decision tree and these variables, a peak is artificialy added into the distribution of these variables. Imagine a positive correlation between the B mass in the range 5000 to 6000 GeV and the probability to be signal from the boost decision tree. After cutting away the background there will be an artificial peak in the B mass distribution between 5000 and 6000 GeV. Normally such a peak suggest a new particle in this range. This is to avoid at all cost since it will compremise the whole analysis later on.
In the following the correlation between the probability to be signal assigned by the boost decision tree and the B mass are shown. The correlation plots for the different angles can be found in appendix B.

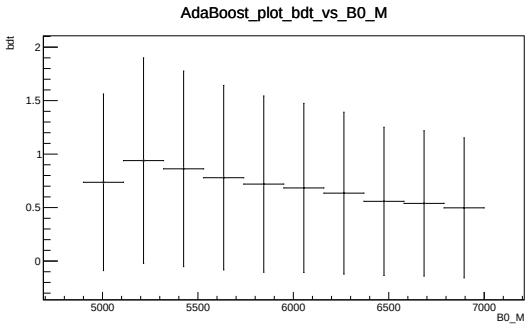


Figure 16: This plot showes the probabilities assign by the boost decision tree as explained in the above paragraph, labeld with bdt on the Y achsis. On the X achsis the values for the B mass are shown in MeV.W The horizontal errorbars indicate the length of each bin, while the vertical errorbars represent the standart derivation of the probabilities. One can see in this plot the correlation between the boost decision tree probabilities and the B mass for the AdaBoost algorithm. There is an obvius correlation from the second bin to the fourth bin.

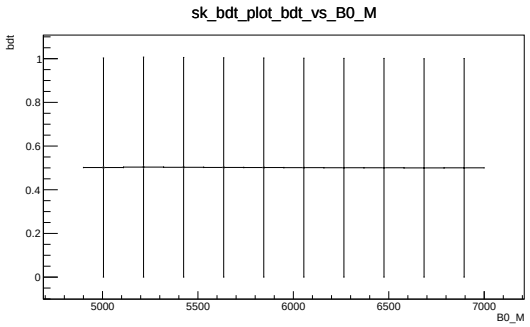


Figure 17: This plot shows the correlations between the probabilities to be signal assigned by the sk_bdt algorithm and the B mass in MeV. There is no correlation.

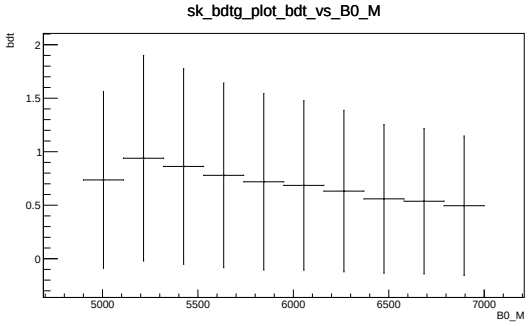


Figure 18: This plot shows the correlations between the probabilities to be signal assigned by the sk_bdtg algorithm and the B mass in MeV. There is a obvius correlation from the second bin to the fourth bin.

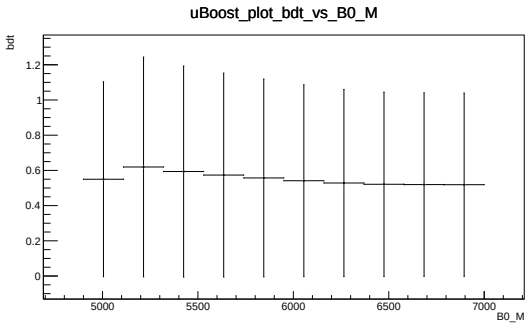


Figure 19: This plot shows the correlations between the probabilities to be signal assigned by the uBoost algorithm and the B mass in MeV. There is just a very small correlation compared to the other algorithms.

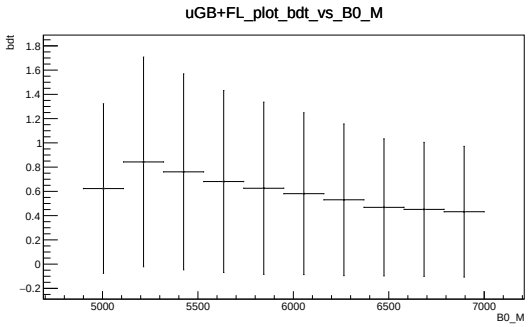


Figure 20: This plot shows the correlations between the probabilities to be signal assigned by the uGB+FL algorithm and the B mass in MeV. There is an obvius correlation from the second bin to the fourth bin.

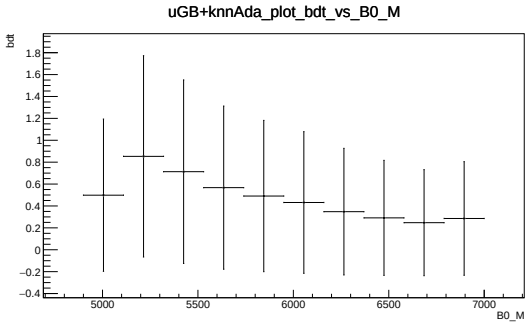


Figure 21: This plot shows the correlations between the probabilities to be signal assigned by the ugb+knnAda algorithm and the B mass in MeV. There is a obvius correlation from the second bin to the fourth bin.

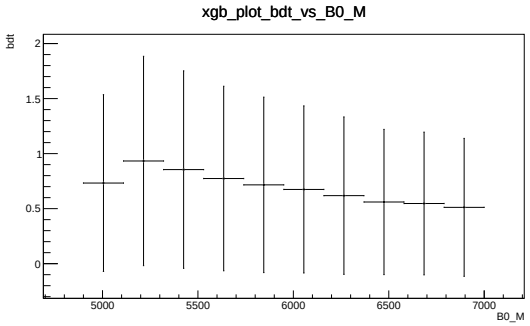


Figure 22: This plot shows the correlations between the probabilities to be signal assigned by the xgb algorithm and the B mass in MeV. There is an obvius correlation from the second bin to the fourth bin.

Result: The two algorithm with the least correlation are the sk_bdt and the uBoost algorithms. Therfore they most suitable to seperate between signal and background in the $B \to K^* \mu \mu$ data. But the uBoost algorithm is a little bit better in the ROC curve (figure 15). Therefore it was choosen to perform the selection.

2.7. Reweighting

Reweighting is a method to match the Monte Carlo data to the real detector data, to extract quantities not measurable by the detector itself like efficiencies. The match is done by applying weights to the Monte Carlo events.
As an initial weight the sWeights see equation 49 are used. The parameters used for reweighting are applied in the following order:

- number of tracks: 'nTracks'
- transversal momentum of the B Meson: 'B p_T'
- quality of the $K\pi\mu\mu$ vertex. 'B vertex χ^2'

The new weights derived by the difference in data and simulation of these parameters for the very clean channel $K^* \to J/\Psi K^*$ [22], are used to reweight the $B \to K^* \mu\mu$ Monte Carlo sample.

2.7.1. Results of the $B \to K^* \mu \mu$ reweighting

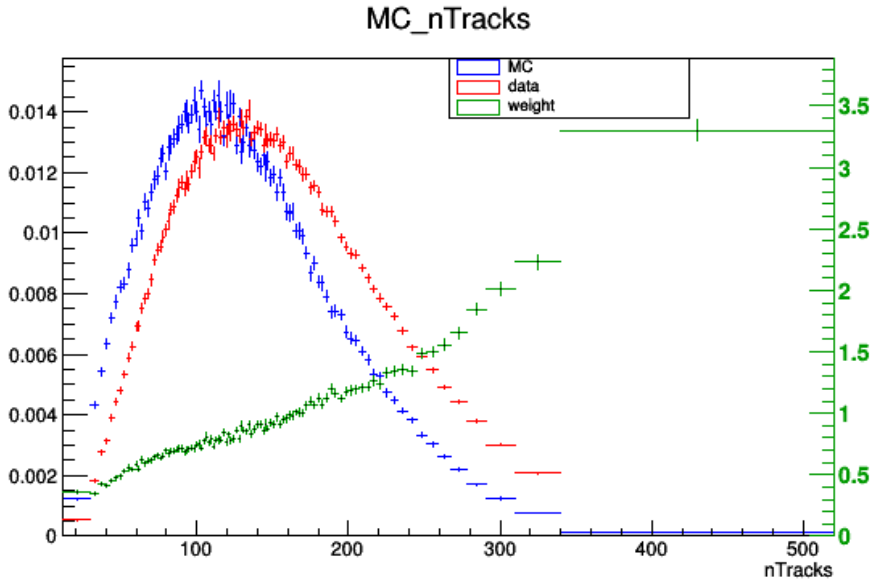


Figure 23: This plot shows the Monte Carlo Simulated data in blue, the detector data in red and in green the best weights to reweight the Monte Carlo data in order to match the detector data. The x- axes are the values for the number of tracks in an event (nTracks). On the left y-axes the fraction of events having nTracks is plotted. On the right y-axes the magnitude of the weights is shown.

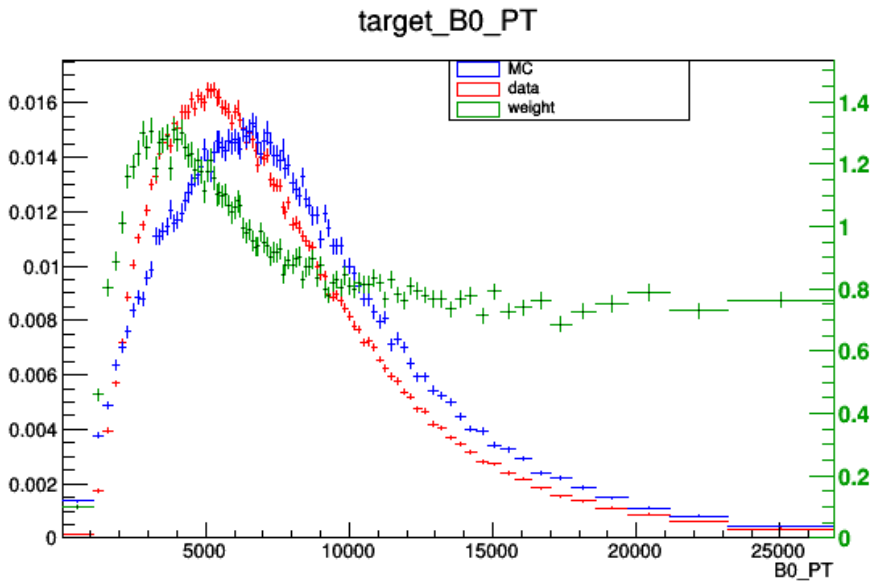


Figure 24: Same plot as in figure 23, for the transversal momentum of the B meson (B0_PT) in MeV.

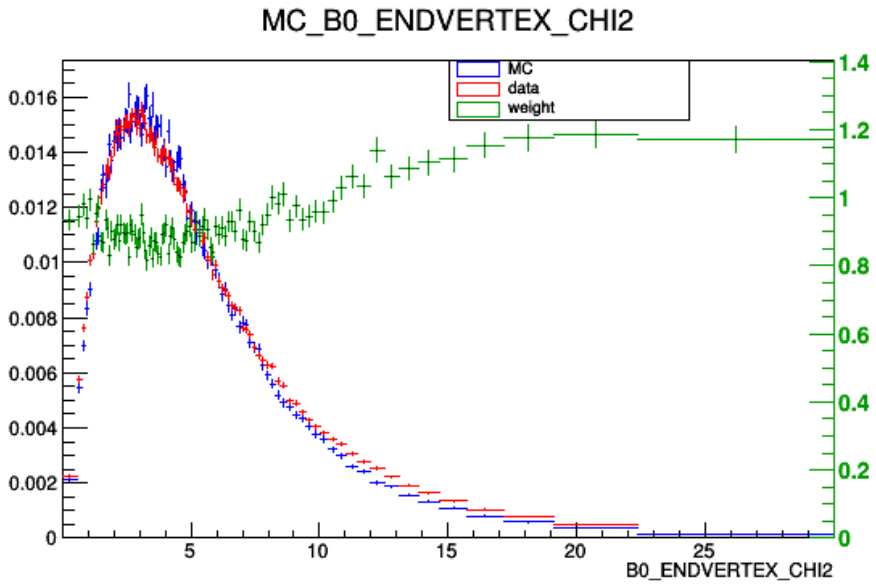


Figure 25: Same plot as in figure 23, for the quality of the $K\pi\mu\mu$ vertex (B0_ENDVERTEX_CHI2).

After applying those weights to the Monte Carlo data, the distributions of all the event parameters will change. To check if the simulation really matches the data. Some comparison plots have been made. One is presented here, while many others can be found in appendix C.

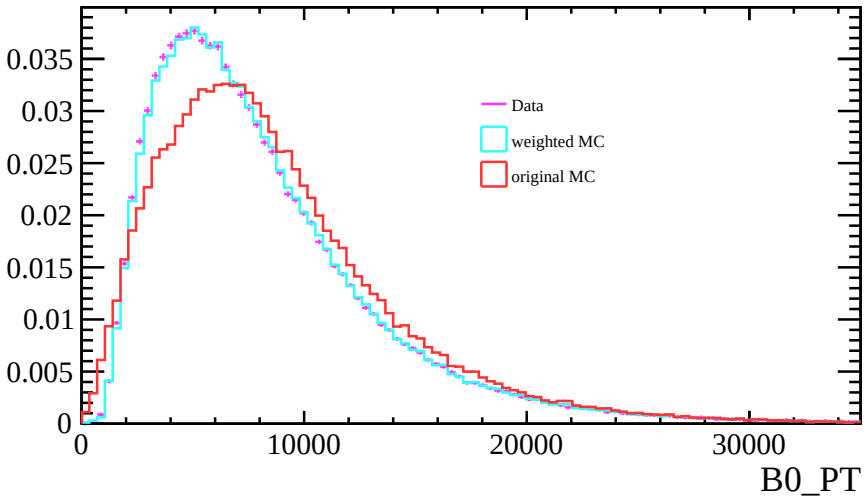


Figure 26: This plots shows the distribution of the B transversal momentum. The X achsis shows the B transversal momentum (B0_PT) in MeV. The violet points represent the distribution of (B0_PT) in the detector data. The red line represents the distribution of (B0_PT) before reweighting the Monte Carlo data. The blue line represents the distribution (B0_PT) after reweighting the Monte Carlo data.

2.7.2. SPlot

In this section the SPlot Technique is explained. It is used to separate two or more merged distributions and has been used to get the initial weights for the re-weighting in this chapter.

Likelihood method

Consider an analysis of a data sample, which consists of several types of events. These types represent signal components and background components, for example from different experiments. The log-liklihood of such a data sample is expressed as:

$$L = \sum_{i=1}^{N} \ln \left[\sum_{j=1}^{N_S} N_j f_j(y_i) \right] - \sum_{i=j}^{N_S} N_j \tag{30}$$

- N = total number of events
- N_S = number of types
- N_i = expected average number of events for type i
- y = set of diciminating variables
- f_j = PDF of the ith type
- $f_j(y_i)$ = value of PDF for event y_i

– x = control variable, not a part of L by construction

The yields N_i and the free parameters of the PDF are obtained by maximizing the above log-likelihood (eq 30).

${}_{in}Plot$ technique

Consider a variable x which can be expressed as a function of the discriminating variables y used in the fit. Furthermore a fit has been performed to determine the yields N_i for all types. From the knowledge of the PDF and the values of Ni a naive weight can be defined as:

$$P_n(y_i) = \frac{N_n f_n(y_i)}{\sum_{k=1}^{N_s} N_k f_k(y_i)} \tag{31}$$

which leads the the x-distribution $\tilde{M}_n$ defined by:

$$N_n \tilde{M}_n(\bar{x}) = \sum_{i \subset \delta x} P_n(y_i) \tag{32}$$

where sum $\sum_{i \subset \delta x}$ contains alle events for which x_i lies in the interval centered on $\bar{x}$ and of total width δx. Therefore $N_n \tilde{M}_n(\bar{x}) \delta x$ is the x-distribution of the histogrammed events, using the weights of eq 31. With this procedure one can on average reproduce the true distribution $\mathbf{M}_n(x)$. One can even replace the sum in eq 32 by an integral:

$$\left\langle \sum_{i \subset \delta x} \right\rangle \to \int dy \sum_{j=1}^{N_s} N_j f_j(y) \delta(x(y) - \bar{x}) \delta x \tag{33}$$

Furthermore through identifying the number of events N_i from the fit one gets:

$$\langle N_n \rangle \tilde{M}_n(\bar{x}) = \int dy \sum_{j=1}^{N_s} N_j f_j(y) \delta(x(y) - \bar{x}) P_n(y) \tag{34}$$

$$= \int dy \sum_{j=1}^{N_s} N_j f_j(y) \delta(x(y) - \bar{x}) \cdot \frac{N_n f_n(y)}{\sum_{k=1}^{N_s} N_k f_k(y)} \tag{35}$$

$$= N_n \int dy \delta(x(y) - \bar{x}) f_n(y) \tag{36}$$

$$= N_n \mathbf{M}_n(\bar{x}) \tag{37}$$

One can see that the sum over events of the naive weight P_n provides a direct estimate of the x-distribution for the nth type. But this procedure has a major drawback. Since x is correlated to y, the PDFs of x enter implicity in the definition of the naive weight. Therefore the $\tilde{M}_n$ distributions are a bad estimate for the quality of the fit. These distributions are biased in a difficult way, when the PDFs $f_i(y)$ are not accurate.
Consider for example a data sample where one of the types has events on the tail of the x-distribution. Such events require the true distribution to account for the tail. But since the events are averaged the weights on the tail are going to be very small missing those events in the estimated true distribution. Only the core of the x-distribution can be examined with ${}_{in}Plots$.

${}_{S}Plot$ technique

In the previous section it was shown that if a variable x belongs to a set y of discriminating varibales, the expected x distribution can be reconstruct. Consider now two sets of variables x and y, where x does

not belong to y and which are uncorrelated , hence the total PDFs $f_i(x,y)$ all factorize into products $\mathbf{M}_i(x)f_i(y)$. The equation 37 does not hold anymore because, when summing over the events the x-PDFs $\mathbf{M}_j(x)$ appears:

$$\langle N_n \rangle \tilde{M}_n(\bar{x}) = \int\int dydx \sum_{j=1}^{N_s} N_j \mathbf{M}_j(x) f_j(y) \delta(x-\bar{x}) P_n \tag{38}$$

$$= \int dy \sum_{j=1}^{N_s} N_j \mathbf{M}_j(\bar{x}) f_j(y) \frac{N_n f_n(y)}{\sum_{k=1}^{N_s} N_k f_k(y)} \tag{39}$$

$$= N_n \sum_{j=1}^{N_s} \mathbf{M}_j(\bar{x}) \left(N_j \int dy \frac{f_n(y) f_j(y)}{\sum_{k=1}^{Ns} N_k f_k(y)} \right) \tag{40}$$

$$\neq N_n \mathbf{M}_n(\bar{x}). \tag{41}$$

The correction term

$$N_j \int dy \frac{f_n(y) f_j(y)}{\sum_{k=1}^{N_s} N_k f_k(y)} \tag{42}$$

is not identical to the kroenecker delta δ_{jn}. In fact the $N_n \tilde{M}_n$ distribution obtained by the naive weight is a linear combination of the true distribution $\mathbf{M}_j$.
To go forward one has to realize that the correction term is related to the inverse of the covariance matrix, given by the second derivatives of $-L$, after the minimization.

$$\mathbf{V}^{-1}_{nj} = \frac{\partial^2(-L)}{\partial N_n \partial N_j} = \sum_{i=1}^{N} \frac{f_n(y_i) f_j(y_i)}{\left(\sum_{k=1}^{N_s} N_k f_k(y_i) \right)^2} \tag{43}$$

If one averages and is replacing the sum over events by intergals (eq 33) the varaince matrix reads:

$$\langle \mathbf{V}^{-1}_{nj} \rangle = \int\int dydx \sum_{e=1}^{N_s} N_e \mathbf{M}_e(x) f_e(y) \frac{f_n(y) f_j(y)}{\left(\sum_{k=1}^{N_s} N_k f_k(y) \right)^2} \tag{44}$$

$$= \int dy \sum_{e=1}^{N_s} N_e f_e(y) \frac{f_n(y) f_j(y)}{\left(\sum_{k=1}^{N_s} N_k f_k(y) \right)^2} \cdot \int dx \mathbf{M}_l(x) \tag{45}$$

$$= \int dy \frac{f_n(y) f_j(y)}{\sum_{k=1}^{N_s} N_k f_k(y)} \tag{46}$$

Therefor equation 38 can be rewritten as:

$$\langle \tilde{M}_n(\bar{x}) \rangle = \sum_{j=1}^{N_s} \mathbf{M}_j(\bar{x}) N_j \langle \mathbf{V}^{-1}_{nj} \rangle. \tag{47}$$

To get the distribution of intrest one has to invert this matrix equation:

$$N_n \mathbf{M}_n(\bar{x}) = \sum_{j=1}^{N_s} \langle \mathbf{V}_{nj} \rangle \langle \tilde{M}_j(\bar{x}) \rangle \tag{48}$$

The true distribution of x can still be reconstructed using the naive weight (eq 31), through a linear combination of ${}_{in}Plots$. In other words: When x does not belong to the set y, the weights are not given by equation 31, they are given by a covariance-weighted quantity called sWeight defined by:

$$ {}_sP_n(y_i) = \frac{\sum_{j=1}^{N_s} \mathbf{V}_n j f_j(y_i)}{\sum_{k=1}^{N_s} N_k f_k(y_i)} \tag{49} $$

With the sWeights on can obtain the distribution of the x variable by histogramming the ${}_sPlot$:

$$ N_{ns}\tilde{M}_n(\bar{x})\delta x = \sum_{i \subset \delta x} {}_sP_n(y_i) \tag{50} $$

On average it reproduced the true distribution:

$$ \langle N_{ns}\tilde{M}_n(x) \rangle = N_n \mathbf{M}_n(x) \tag{51} $$

In the case were x is significantly correlated with y, the sPlots from equation 50 can not be compared with the pure distributions of the various types. To solve that problem one can perform a Monte Carlo simulation of the procedure and obtain the expected distributions to which the ${}_sPlots$ should be compared.
More information on ${}_sPlots$ is given in reference [23].

3. Summary

The RooMCMarkovChain fitting routine is currently beeing added [24] to the ROOT Data analysis framework. The best algorithm uBoost from the boost decision tree test will be used to find future $B \to K^* \mu\mu$ decays in the data produced by the LHCb detector. The rewighting done in this thesis will be used for the future analysis of the $B \to K^* \mu\mu$ in the current LHCb data.

References

[1] Likelihood Fits, Craig Blocker, Brandeis, 23.8.2004, *http://physics.bu.edu/neppsr/2004/Talks/Likelihood-Blocker.pdf*

[2] Minuit Reference Manual Version 94.1, *https://root.cern.ch/sites/d35c7d8c.web.cern.ch/files/minuit.pdf*

[3] ROOT Data Analysis Framework *https://root.cern.ch/*

[4] MINUIT Home page, *https://seal.web.cern.ch/seal/snapshot/work-packages/mathlibs/minuit/*

[5] JHEP08 (2017) 055

[6] C. Bobeth, M. Misiak and J. Urban, Nucl. Phys. B 574 (2000) 291 [arXiv:hep-ph/9910220].

[7] arXiv:0811.1214 [hep-ph]

[8] CERN Accelerator Complex, *http://www.stfc.ac.uk/research/particle-physics-and-particle-astrophysics/large-hadron-collider/cern-accelerator-complex/*

[9] Science and Technology Facilities Council article about LHCb , *https://www.ppd.stfc.ac.uk/Pages/LHCb.aspx*

[10] VELO description, *http://lhcb-public.web.cern.ch/lhcb-public/en/Detector/VELO2-en.html*

[11] RICH description, *http://lhcb-public.web.cern.ch/lhcb-public/en/Detector/RICH2-en.html*

[12] Magnet description, *http://lhcb-public.web.cern.ch/lhcb-public/en/Detector/Magnet2-en.html*

[13] Tracker description, *http://lhcb-public.web.cern.ch/lhcb-public/en/Detector/Trackers2-en.html*

[14] Calorimeters description, *http://lhcb-public.web.cern.ch/lhcb-public/en/Detector/Calorimeters2-en.html*

[15] Muon system description, *http://lhcb-public.web.cern.ch/lhcb-public/en/Detector/Muon2-en.html*

[16] A Short Introduction to Boosting, by Yoav Freund and Rovert E. Schapire *http://www.site.uottawa.ca/~stan/csi5387/boost-tut-ppr.pdf*

[17] J.H. Friedmann, " Greedy function approximation: A gradient boosting machine " , 2001

[18] J. Stevens and M. Williams, uBoost: A boosting method for producing uniform selection efficiencies from multivariate classifiers, JINST 8, P12013 (2013). [arXiv:1305.7248]

[19] arXiv:1603.02754 [cs.LG]

[20] Gradient Boosting classifier from the sk_learn python package *http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingClassifier.html*

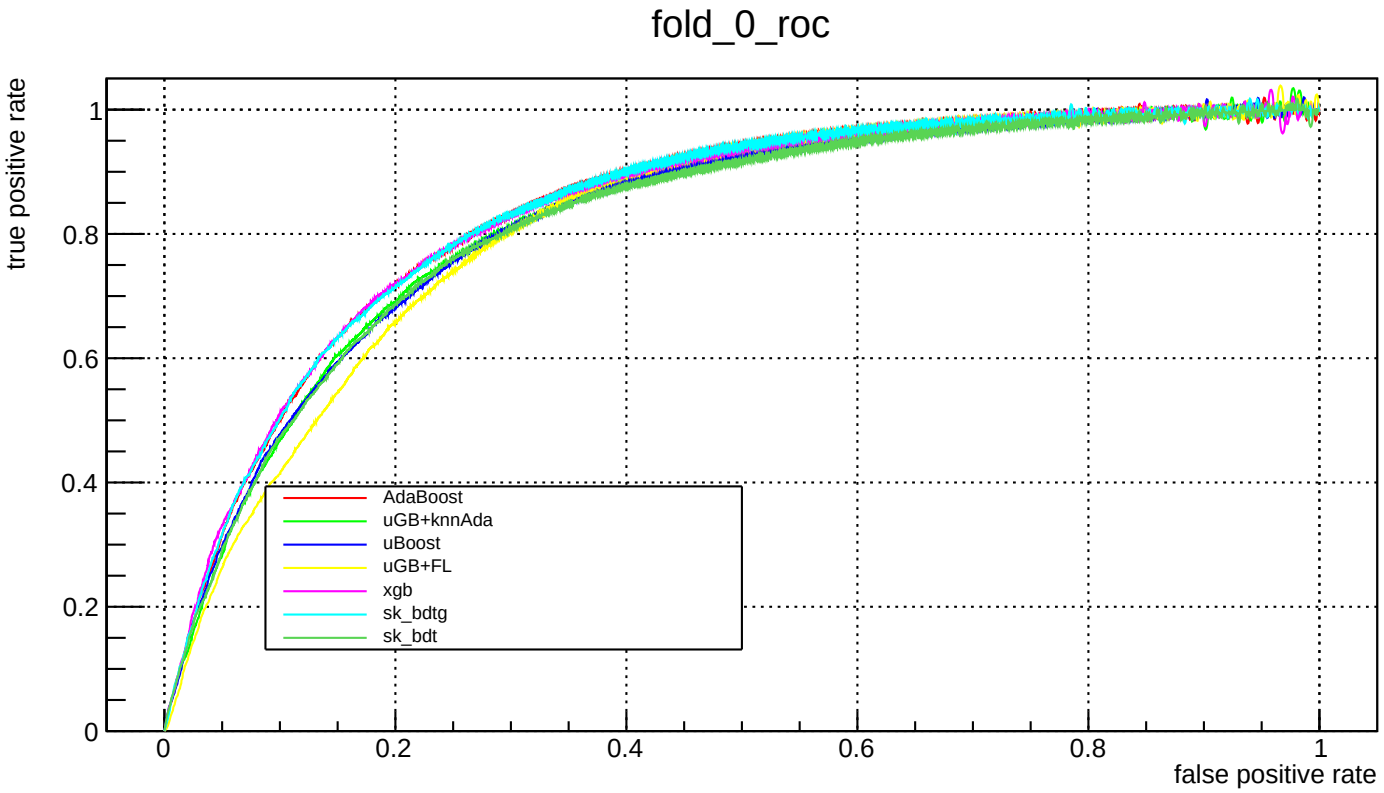
[21] A vaiation of the AdaBoost classifier called AdaBoost-SAMME *http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.AdaBoostClassifier.html*

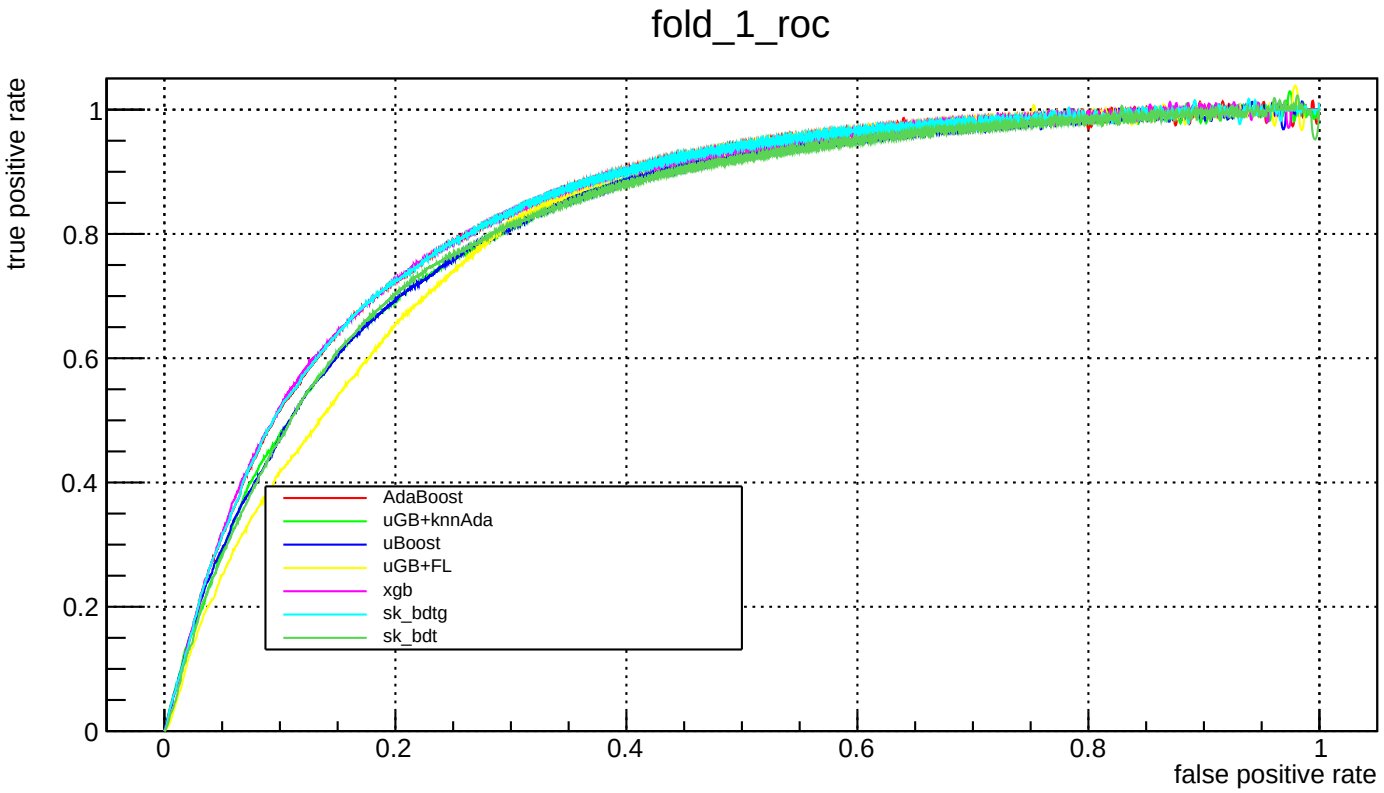
[22] https://arxiv.org/pdf/1103.0423.pdf

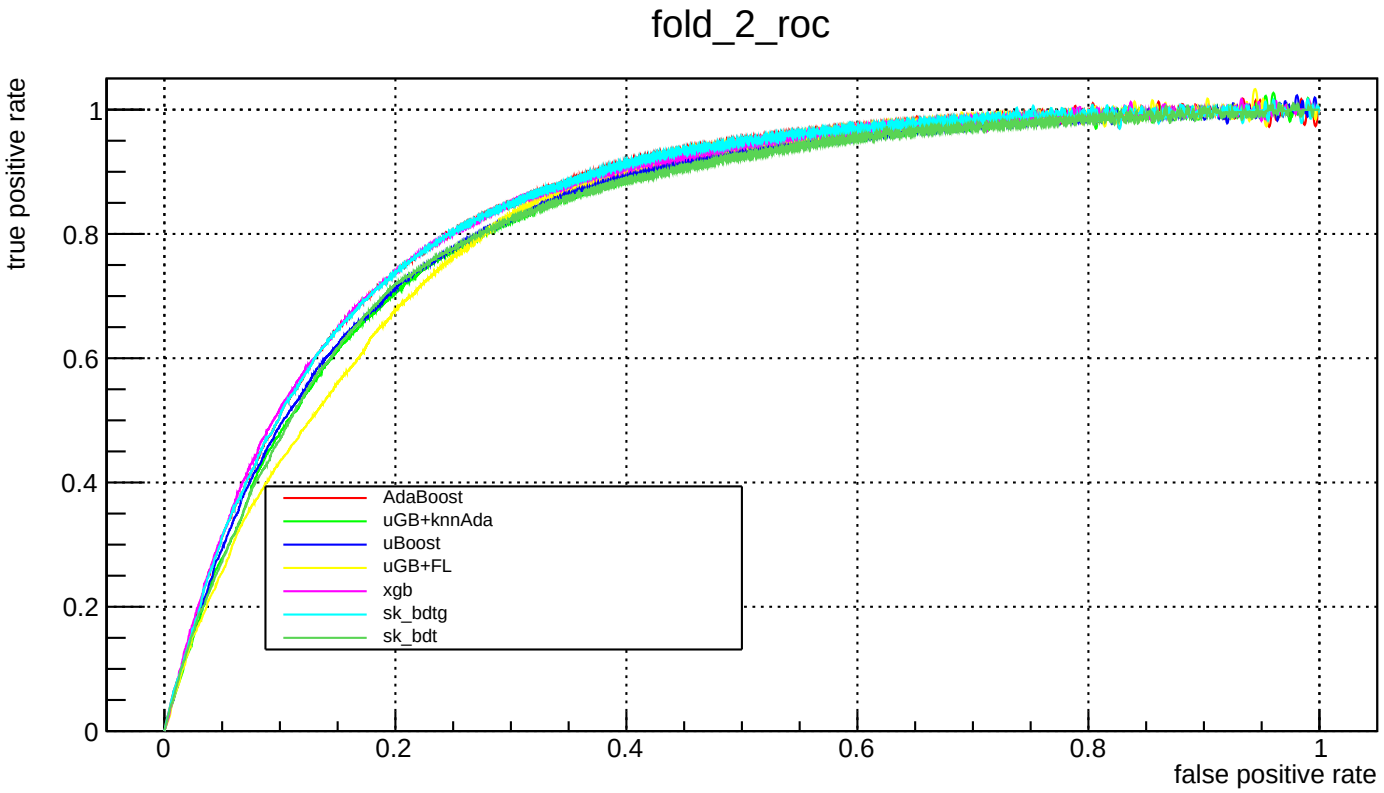
[23] sPlot: a statistical tool to unfold data distributions, arXiv:physics/0402083 [physics.data-an]

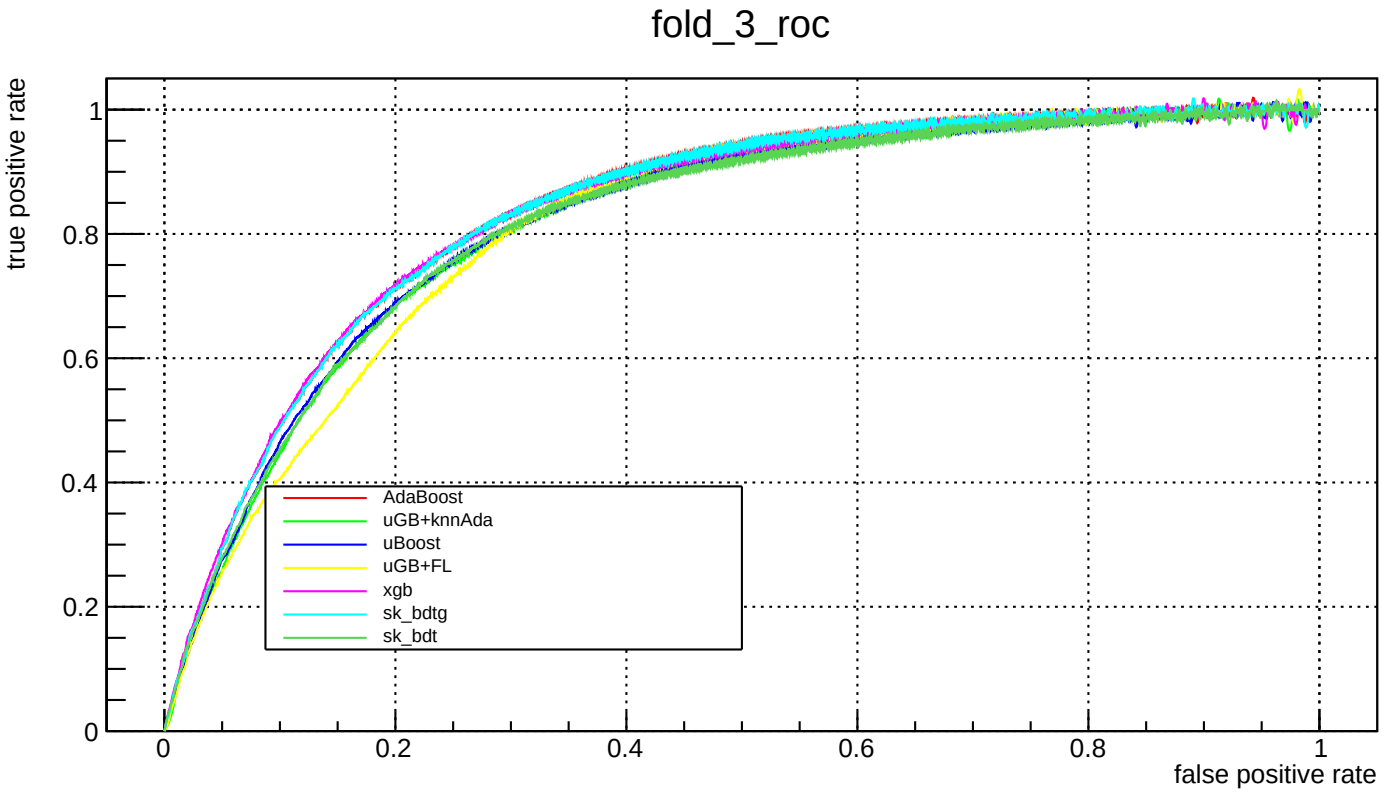
[24] https://github.com/root-project/root/pull/1422

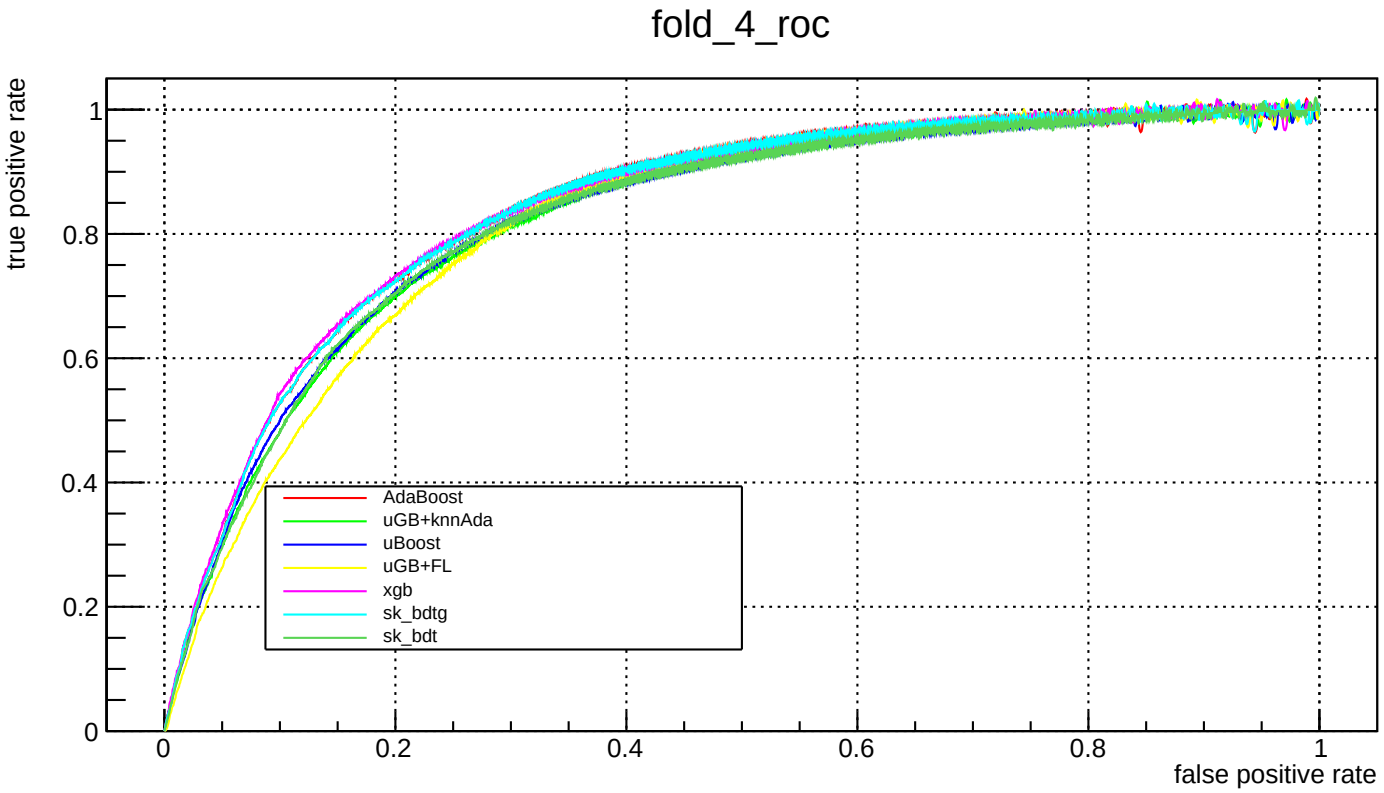
A. ROC curves

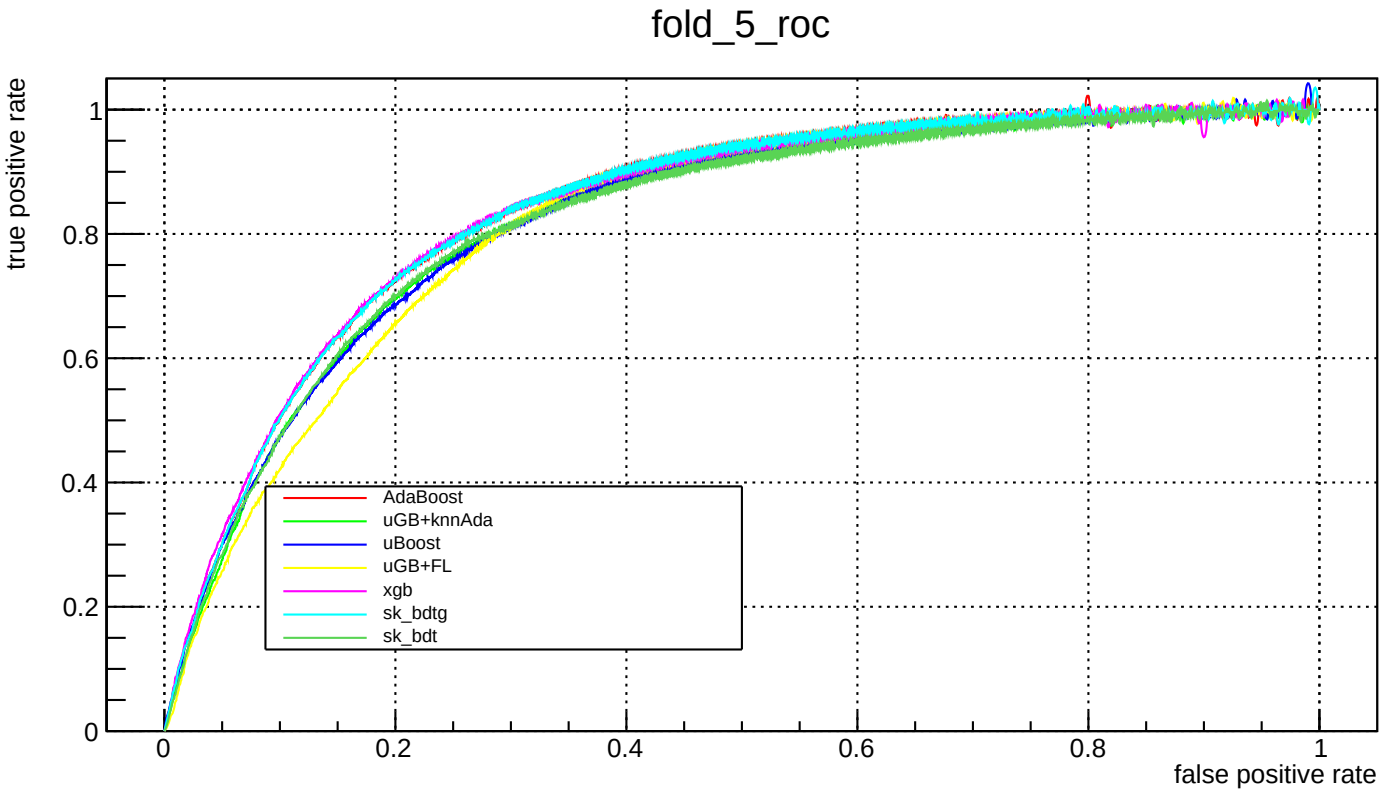

fold_0_roc
true positive rate
false positive rate
0
0.2
0.4
0.6
0.8
1
AdaBoost
uGB+knnAda
uBoost
uGB+FL
xgb
sk_bdtg
sk_bdt

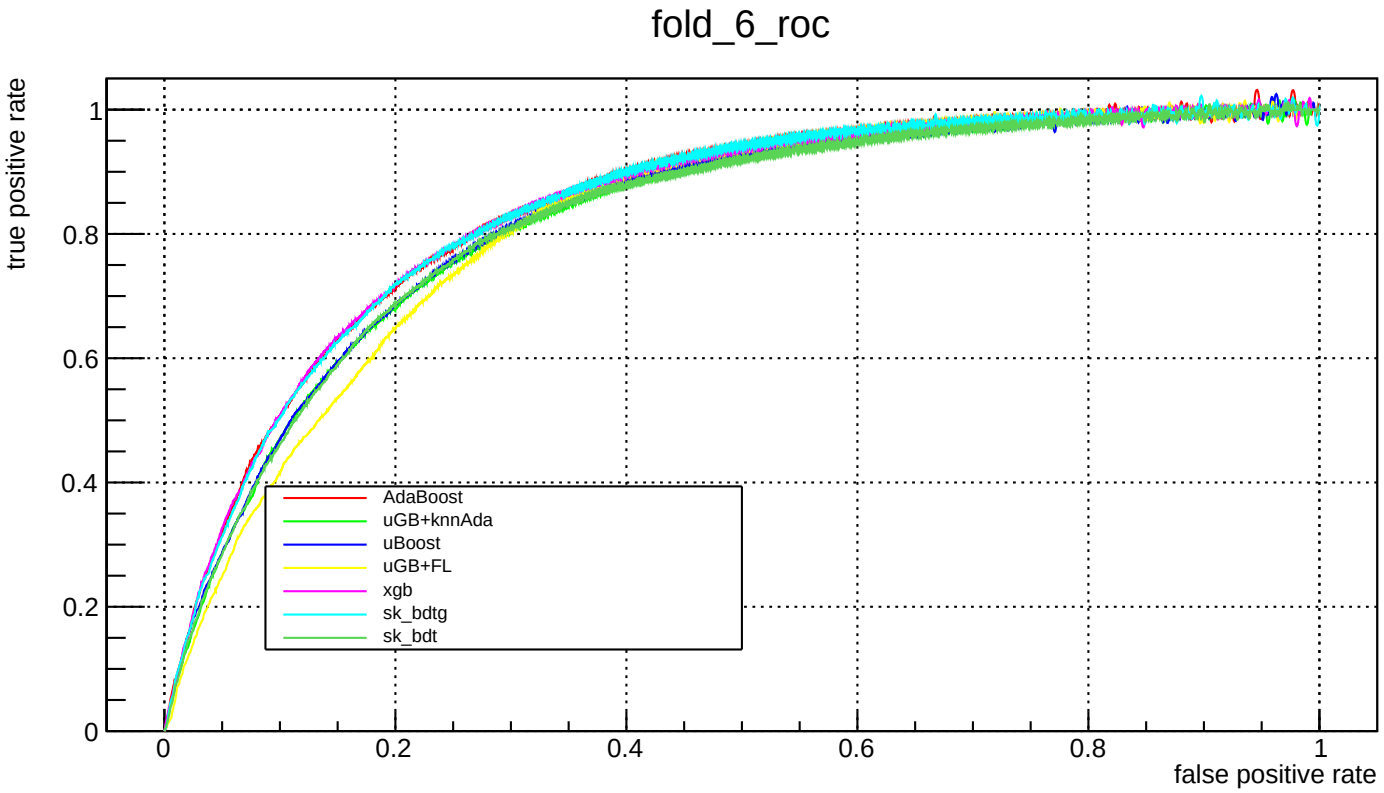

fold_1_roc
true positive rate
false positive rate
0
0.2
0.4
0.6
0.8
1
AdaBoost
uGB+knnAda
uBoost
uGB+FL
xgb
sk_bdtg
sk_bdt

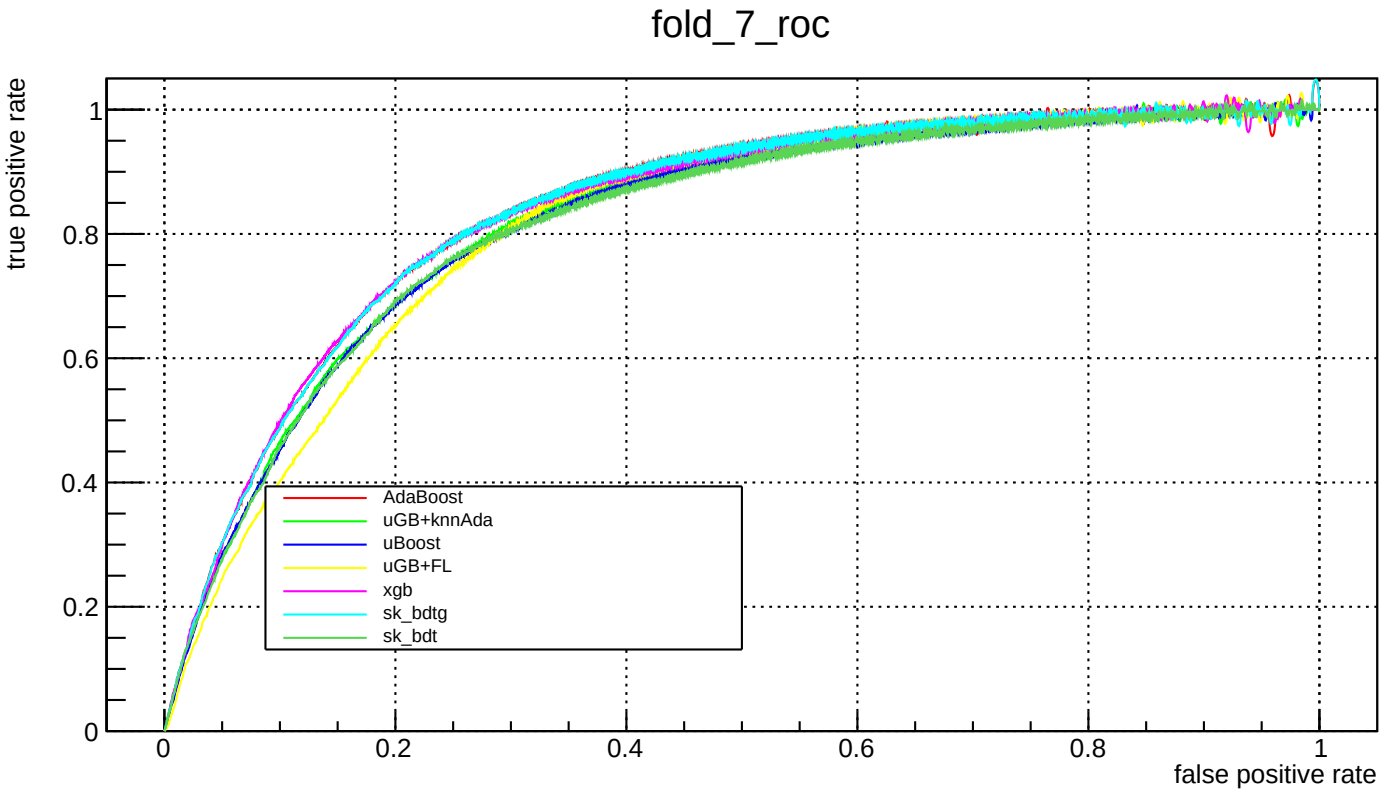

fold_2_roc
true positive rate
false positive rate
0
0.2
0.4
0.6
0.8
1
AdaBoost
uGB+knnAda
uBoost
uGB+FL
xgb
sk_bdtg
sk_bdt

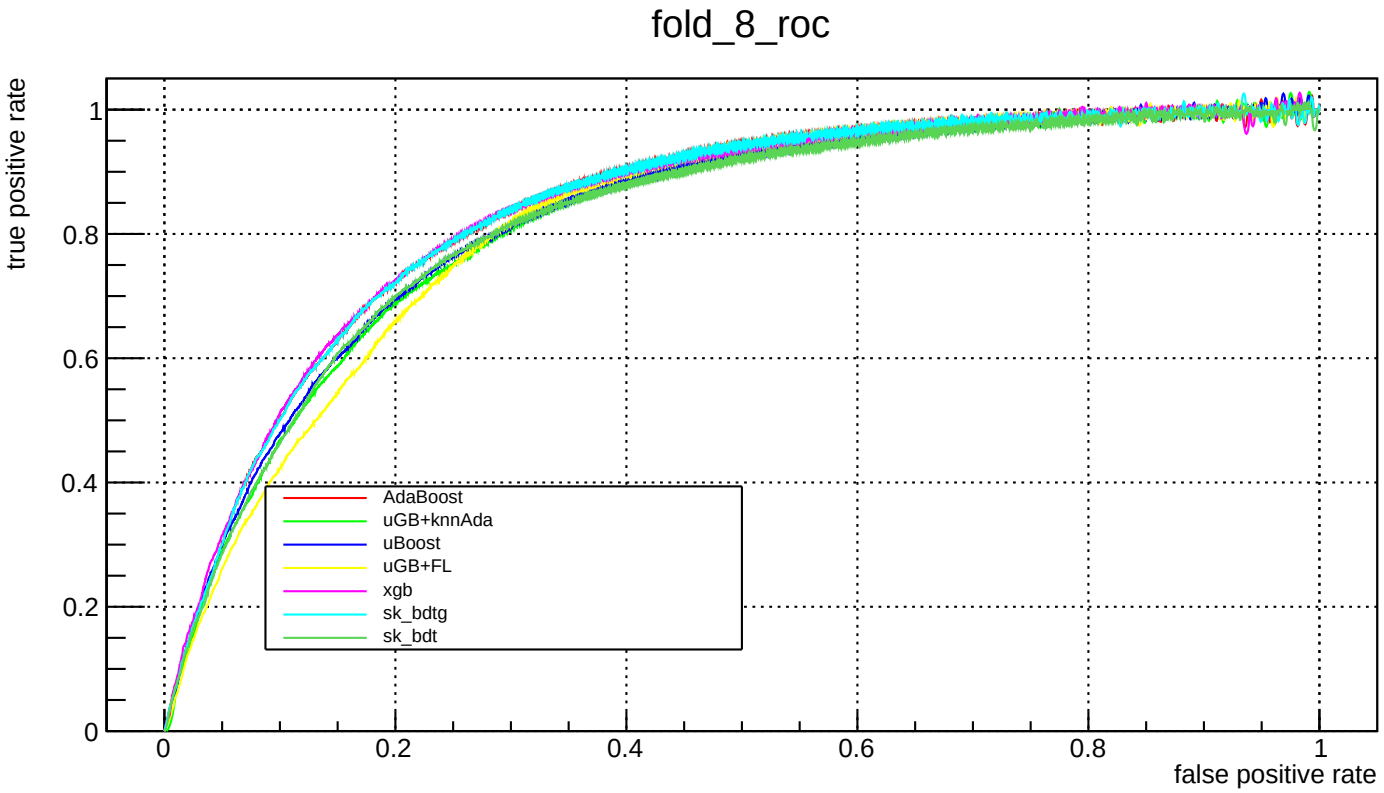

fold_3_roc
true positive rate
false positive rate
0
0.2
0.4
0.6
0.8
1
AdaBoost
uGB+knnAda
uBoost
uGB+FL
xgb
sk_bdtg
sk_bdt

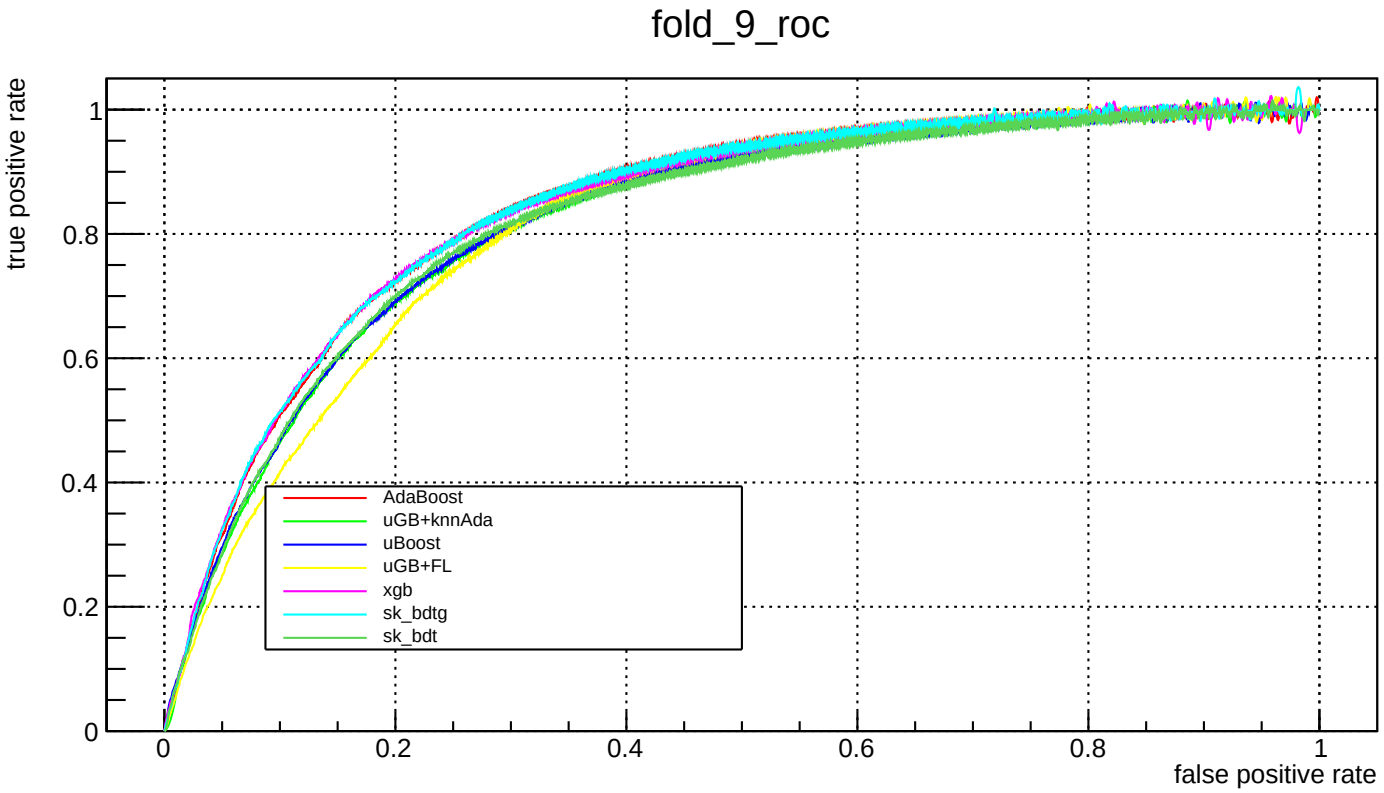

fold_4_roc
true positive rate
false positive rate
0
0.2
0.4
0.6
0.8
1
AdaBoost
uGB+knnAda
uBoost
uGB+FL
xgb
sk_bdtg
sk_bdt


fold_5_roc
true positive rate
false positive rate
0
0.2
0.4
0.6
0.8
1
AdaBoost
uGB+knnAda
uBoost
uGB+FL
xgb
sk_bdtg
sk_bdt

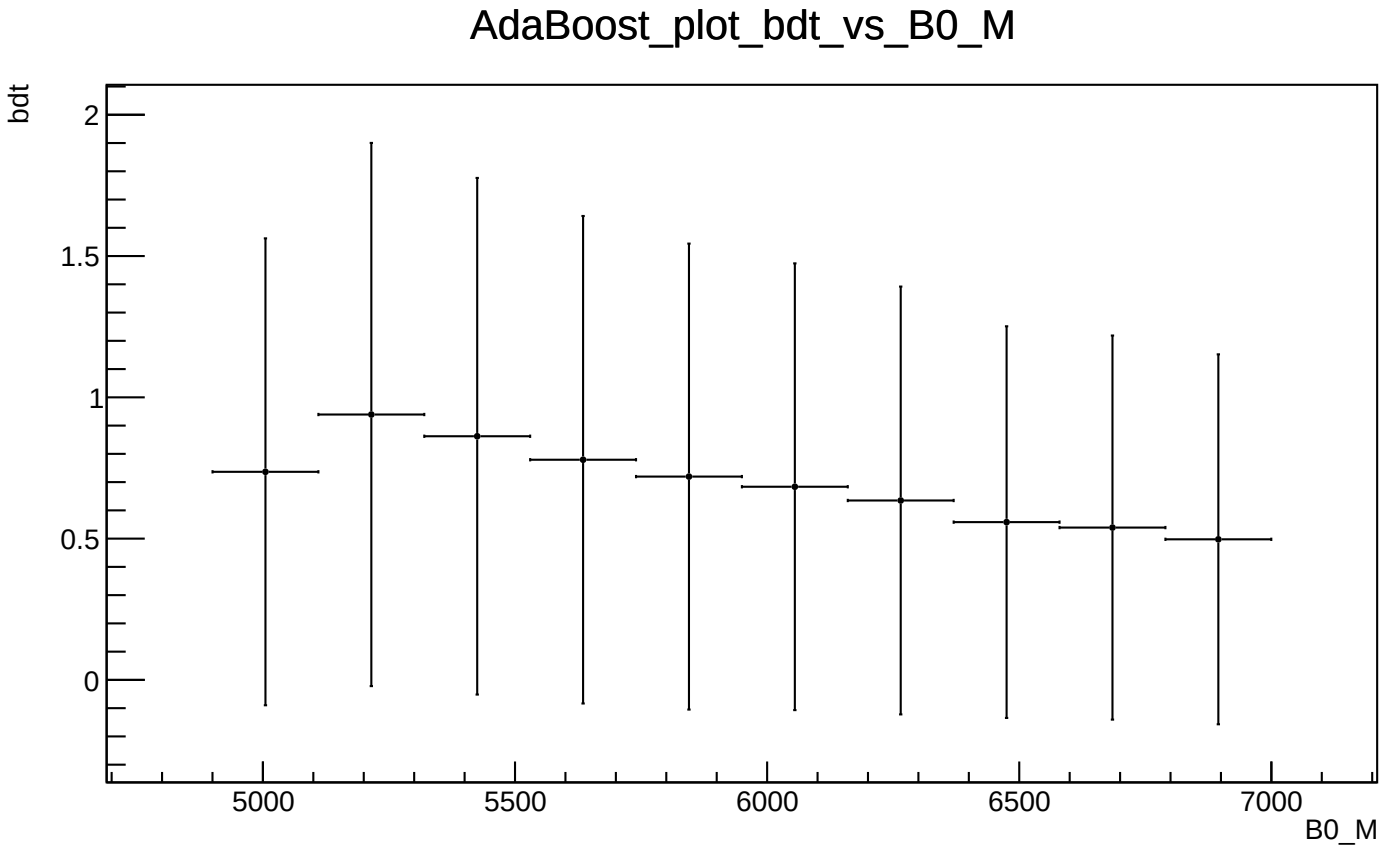

fold_6_roc
true positive rate
false positive rate
0
0.2
0.4
0.6
0.8
1
AdaBoost
uGB+knnAda
uBoost
uGB+FL
xgb
sk_bdtg
sk_bdt

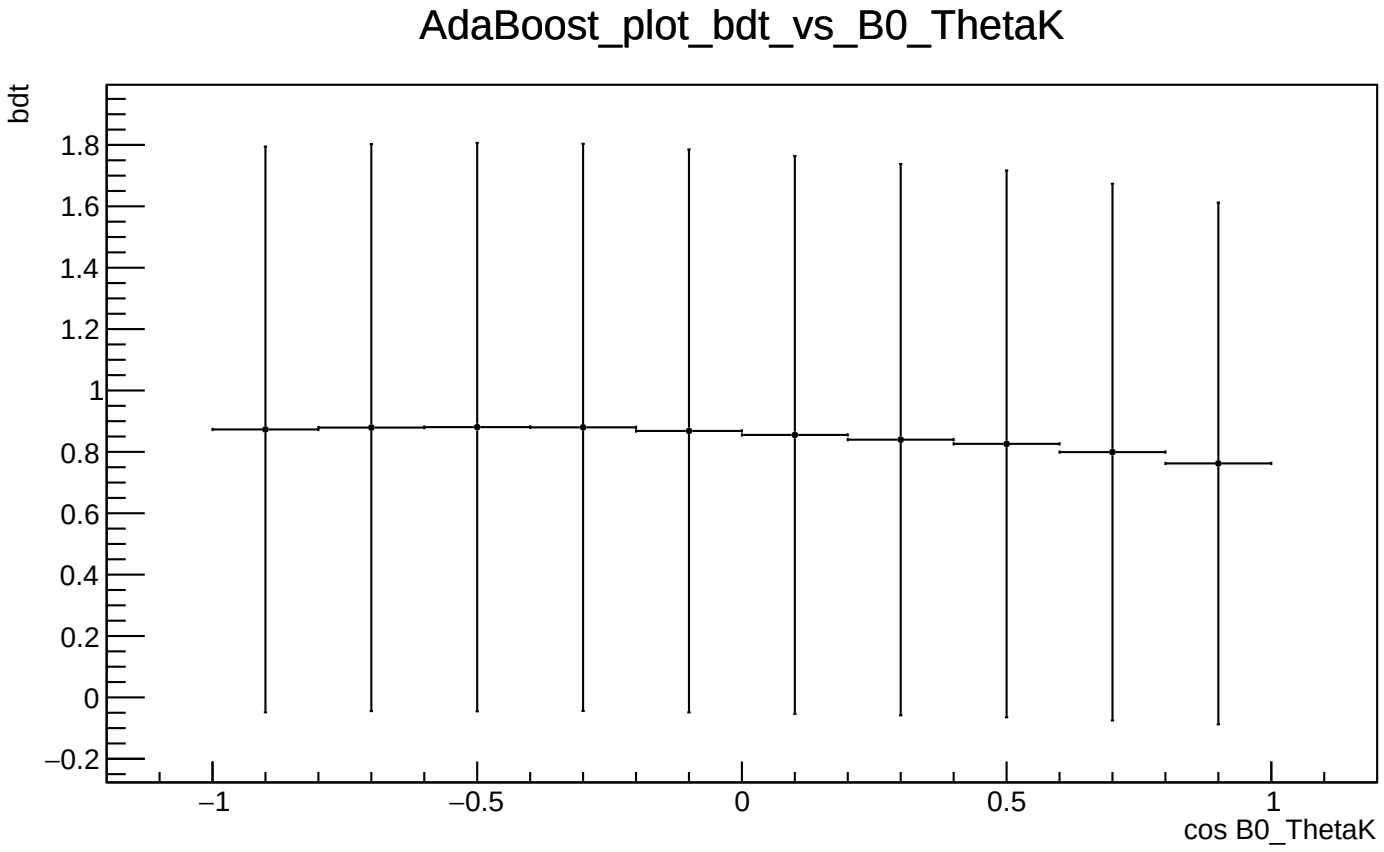

fold_7_roc
true positive rate
false positive rate
0
0.2
0.4
0.6
0.8
1
AdaBoost
uGB+knnAda
uBoost
uGB+FL
xgb
sk_bdtg
sk_bdt

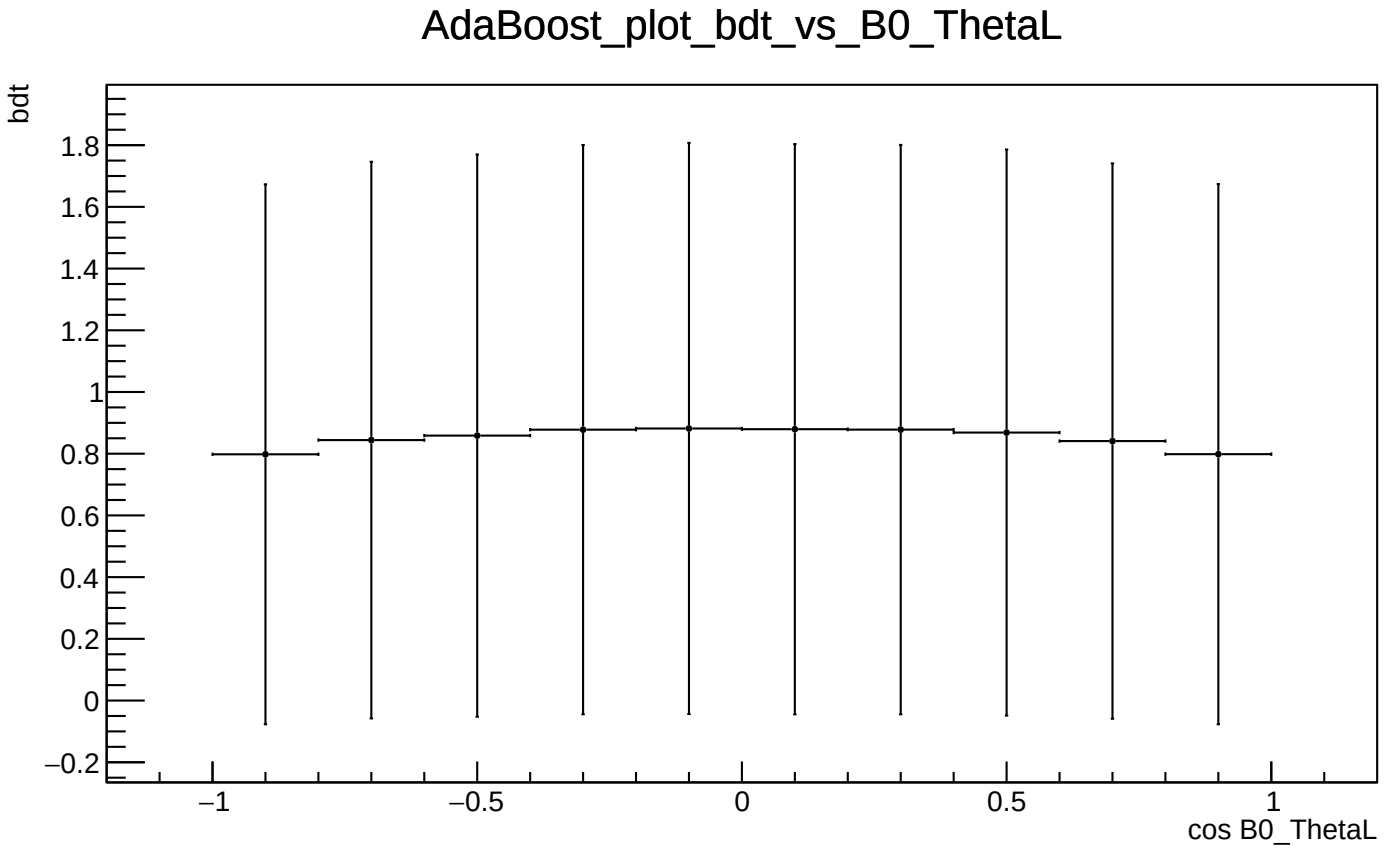

fold_8_roc
true positive rate
false positive rate
0
0.2
0.4
0.6
0.8
1
AdaBoost
uGB+knnAda
uBoost
uGB+FL
xgb
sk_bdtg
sk_bdt

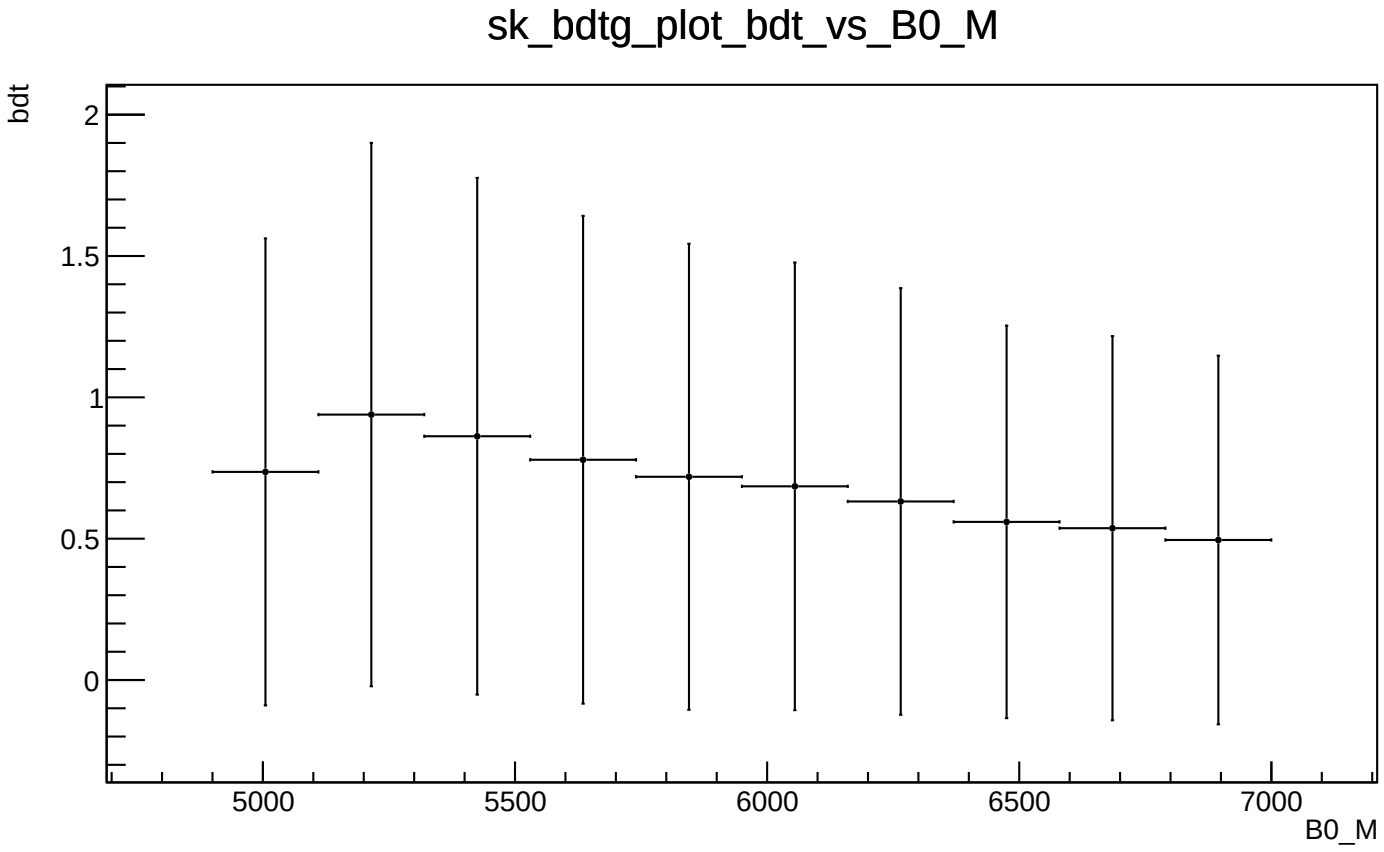

fold_9_roc
true positive rate
false positive rate
0
0.2
0.4
0.6
0.8
1
AdaBoost
uGB+knnAda
uBoost
uGB+FL
xgb
sk_bdtg
sk_bdt

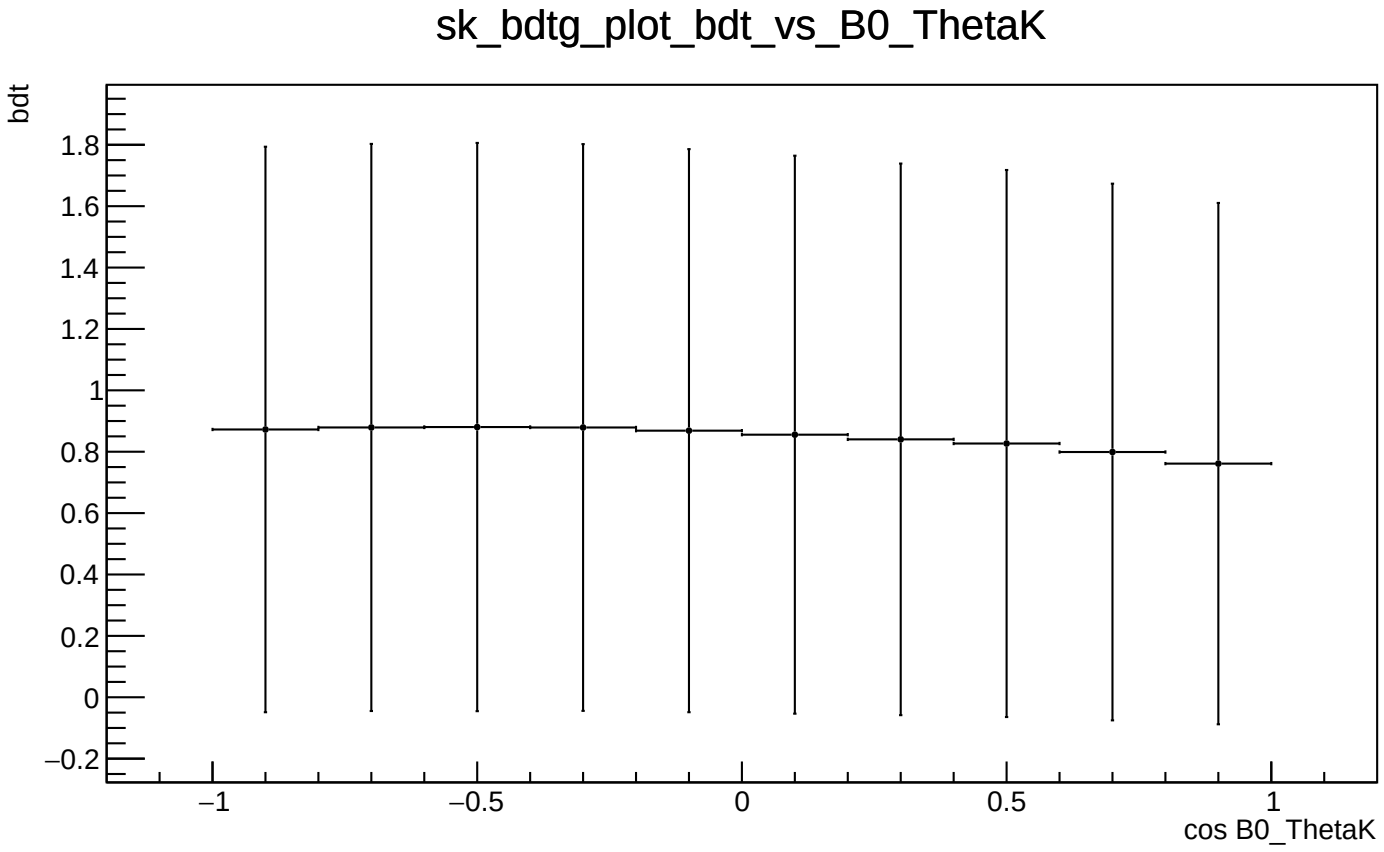
B. Correlation plots

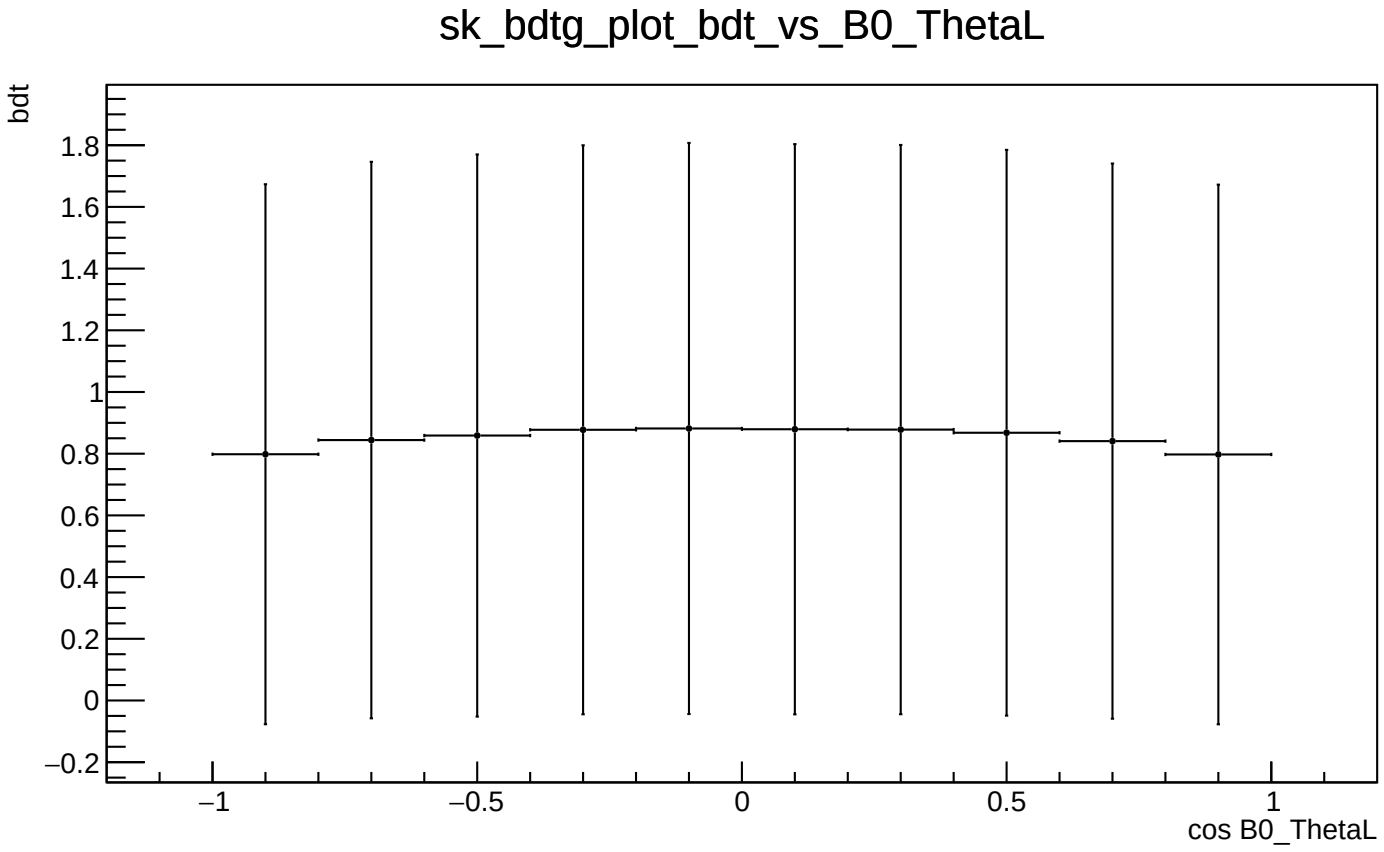

AdaBoost_plot_bdt_vs_B0_M
bdt
2
1.5
1
0.5
0
5000
5500
6000
6500
7000
B0_M

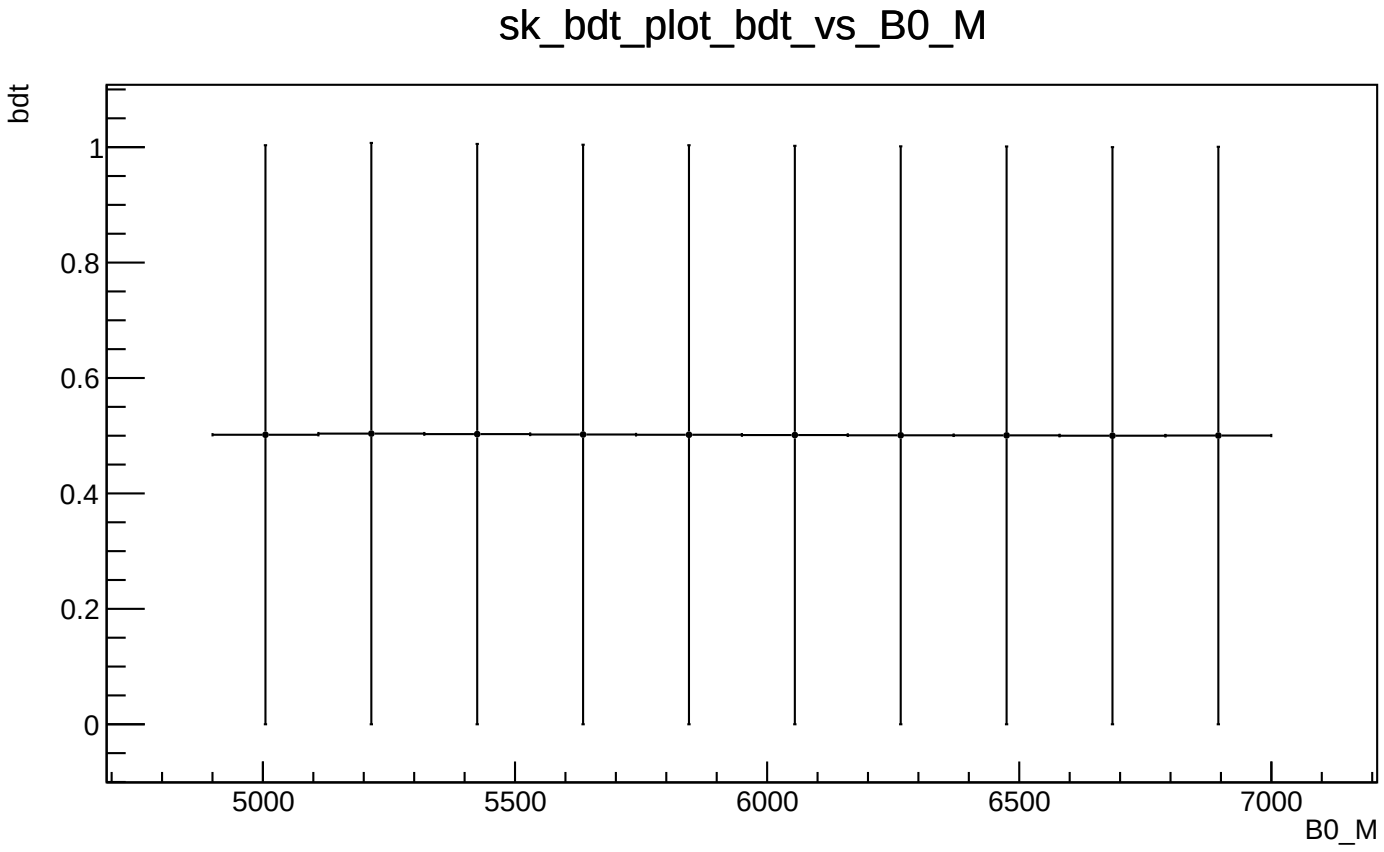

AdaBoost_plot_bdt_vs_B0_ThetaK
bdt
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
−0.2
−1
−0.5
0
0.5
1
cos B0_ThetaK

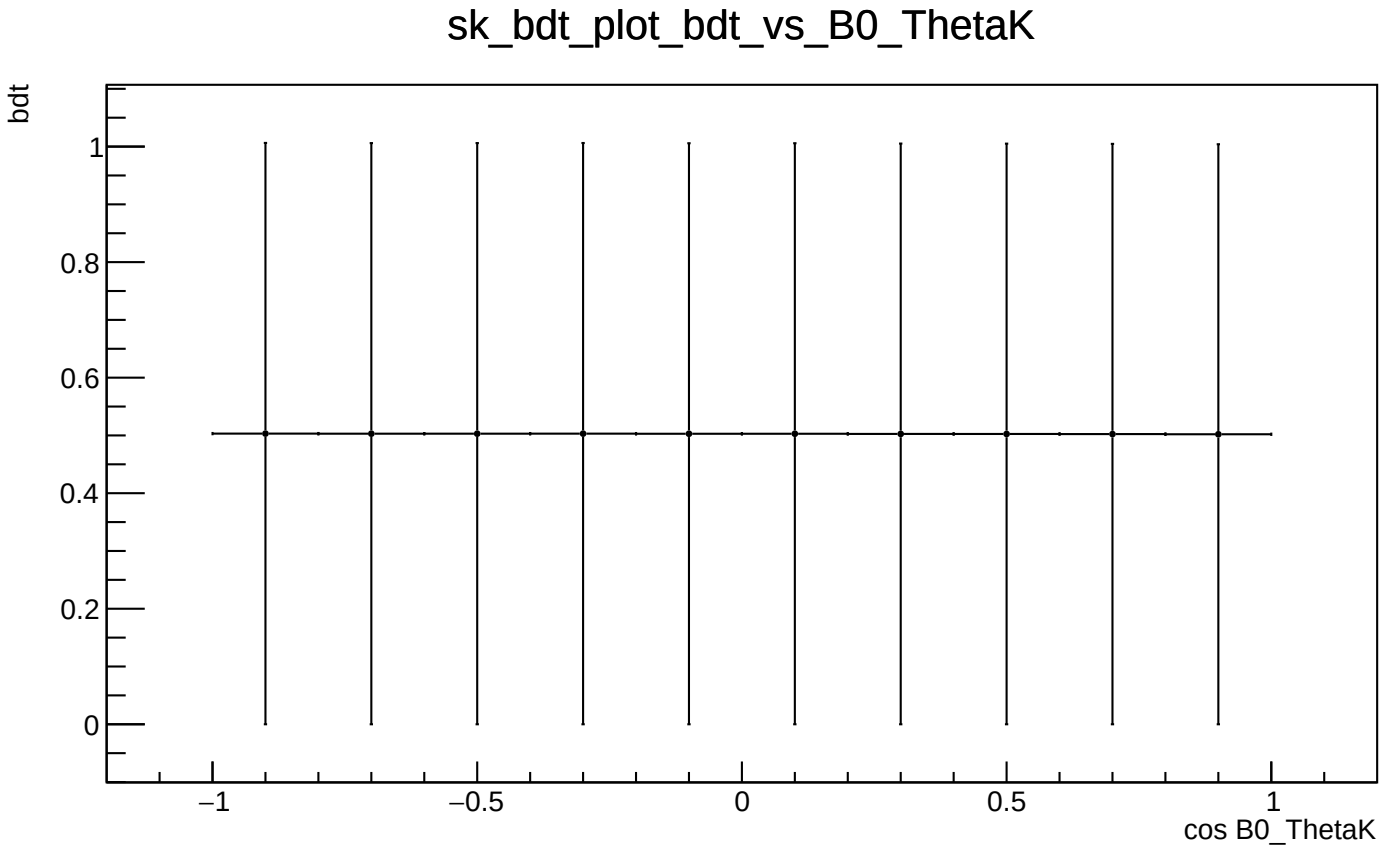

AdaBoost_plot_bdt_vs_B0_ThetaL
bdt
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
−0.2
−1
−0.5
0
0.5
1
cos B0_ThetaL

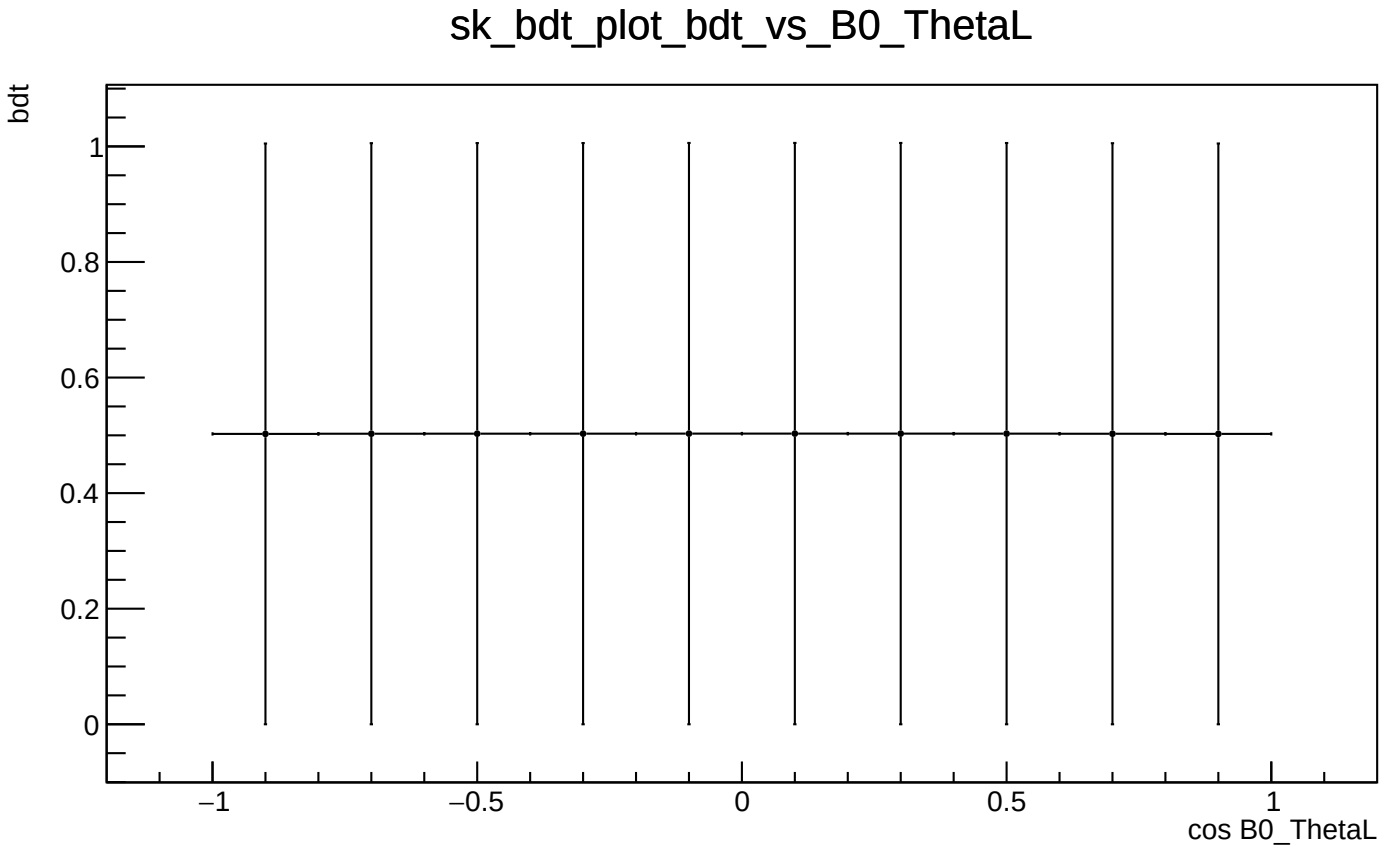

sk_bdtg_plot_bdt_vs_B0_M
bdt
2
1.5
1
0.5
0
5000
5500
6000
6500
7000
B0_M

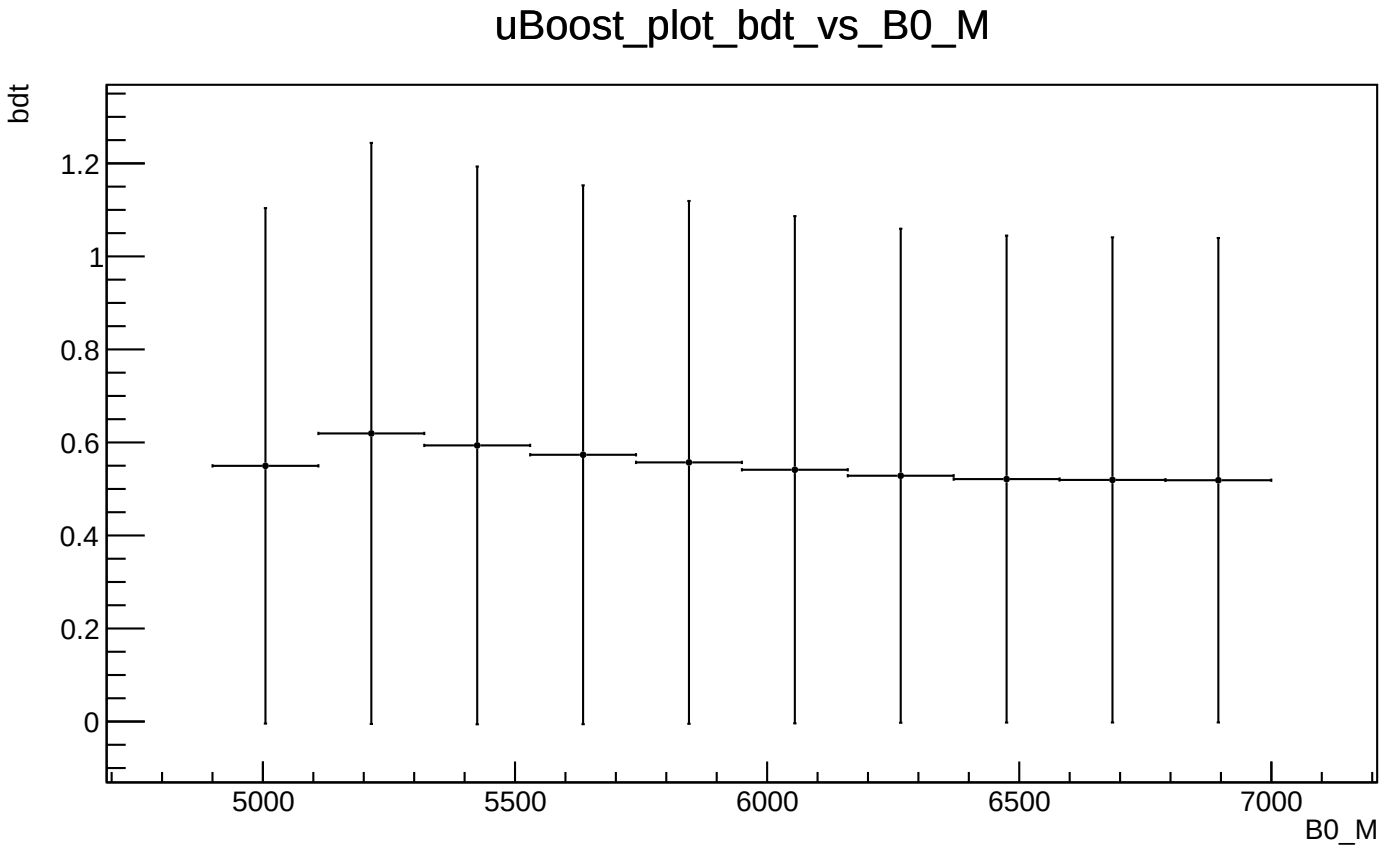

sk_bdtg_plot_bdt_vs_B0_ThetaK
bdt
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
−0.2
−1
−0.5
0
0.5
1
cos B0_ThetaK

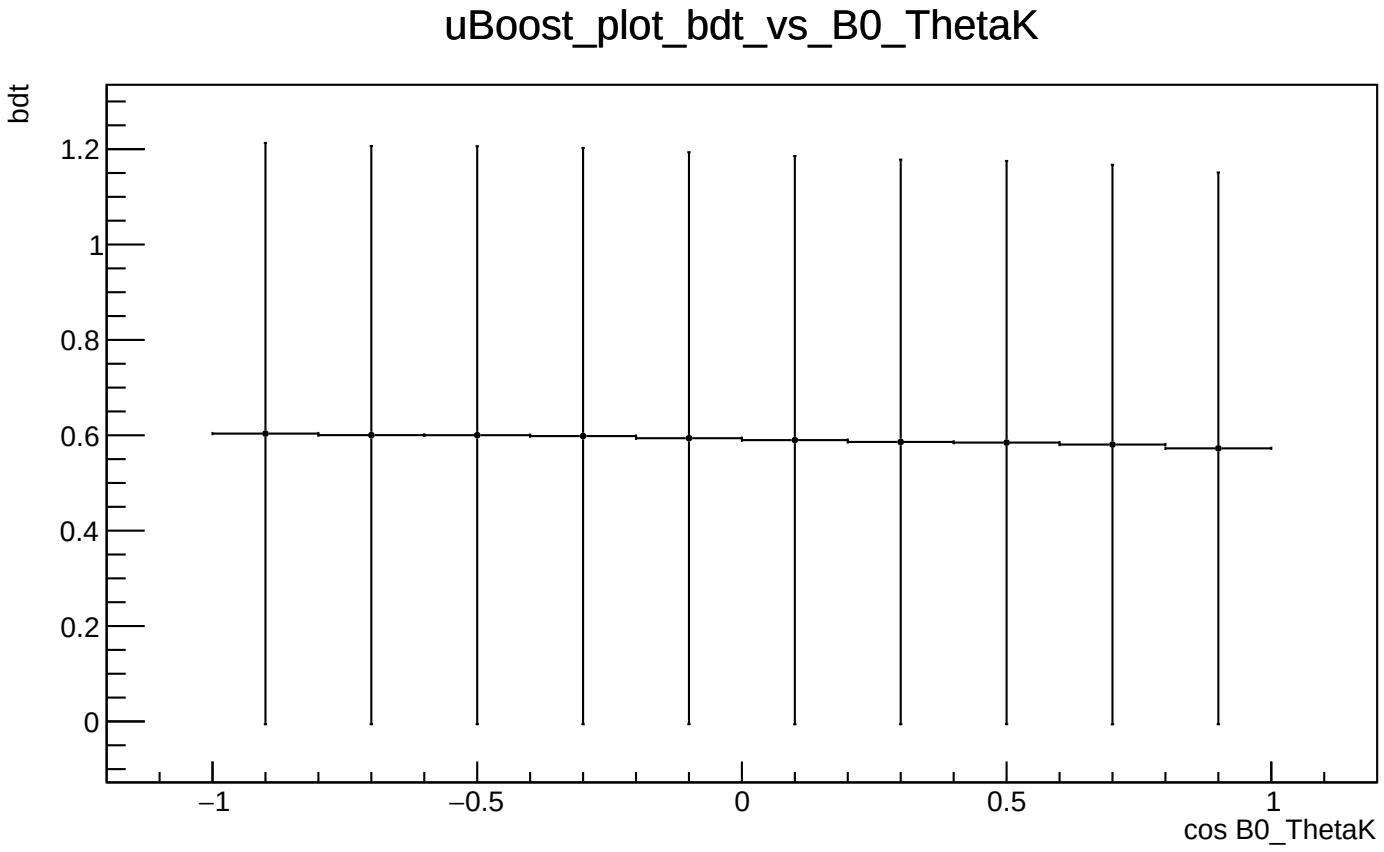

sk_bdtg_plot_bdt_vs_B0_ThetaL
bdt
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
−0.2
−1
−0.5
0
0.5
1
cos B0_ThetaL

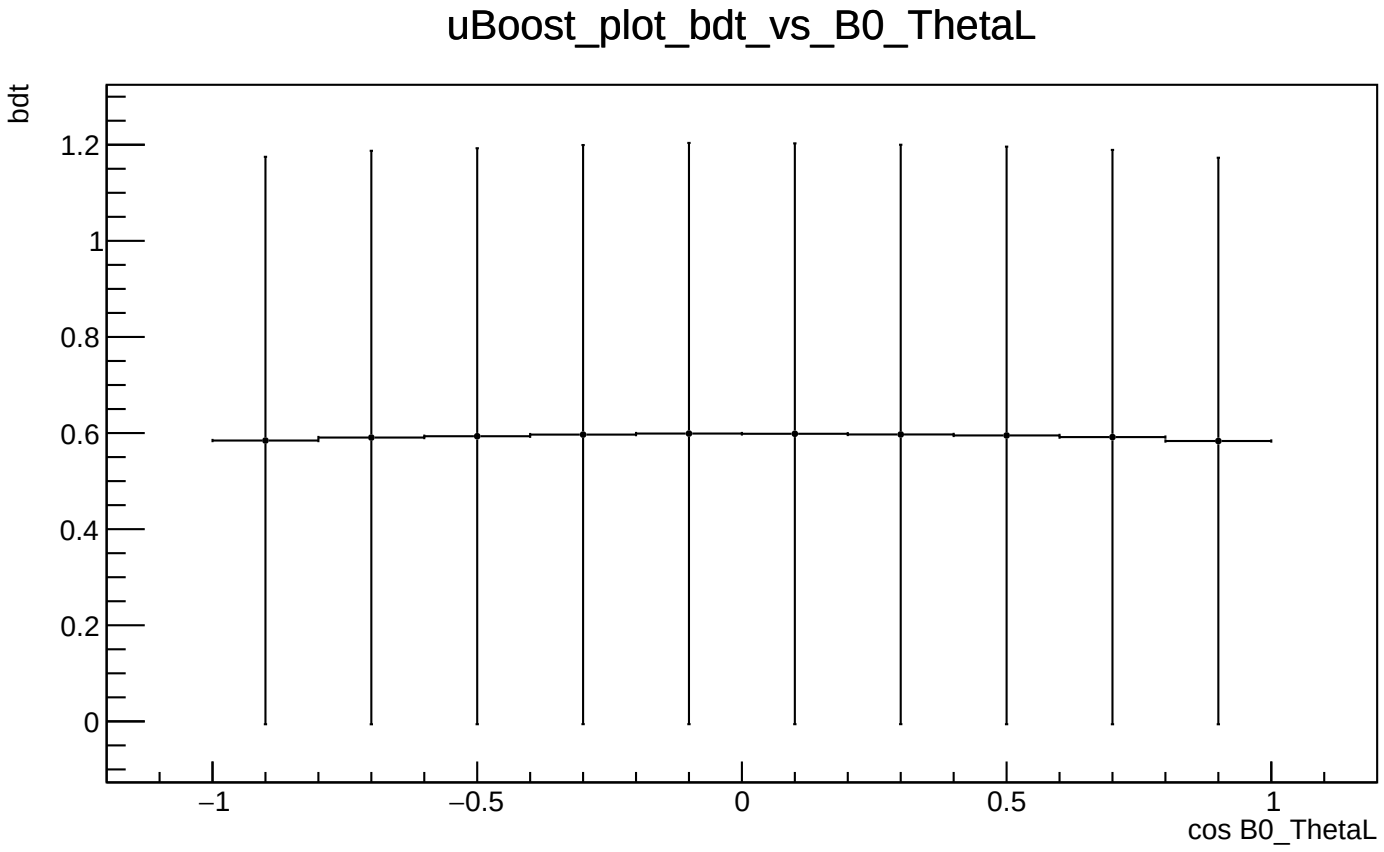

sk_bdt_plot_bdt_vs_B0_M
bdt
1
0.8
0.6
0.4
0.2
0
5000
5500
6000
6500
7000
B0_M

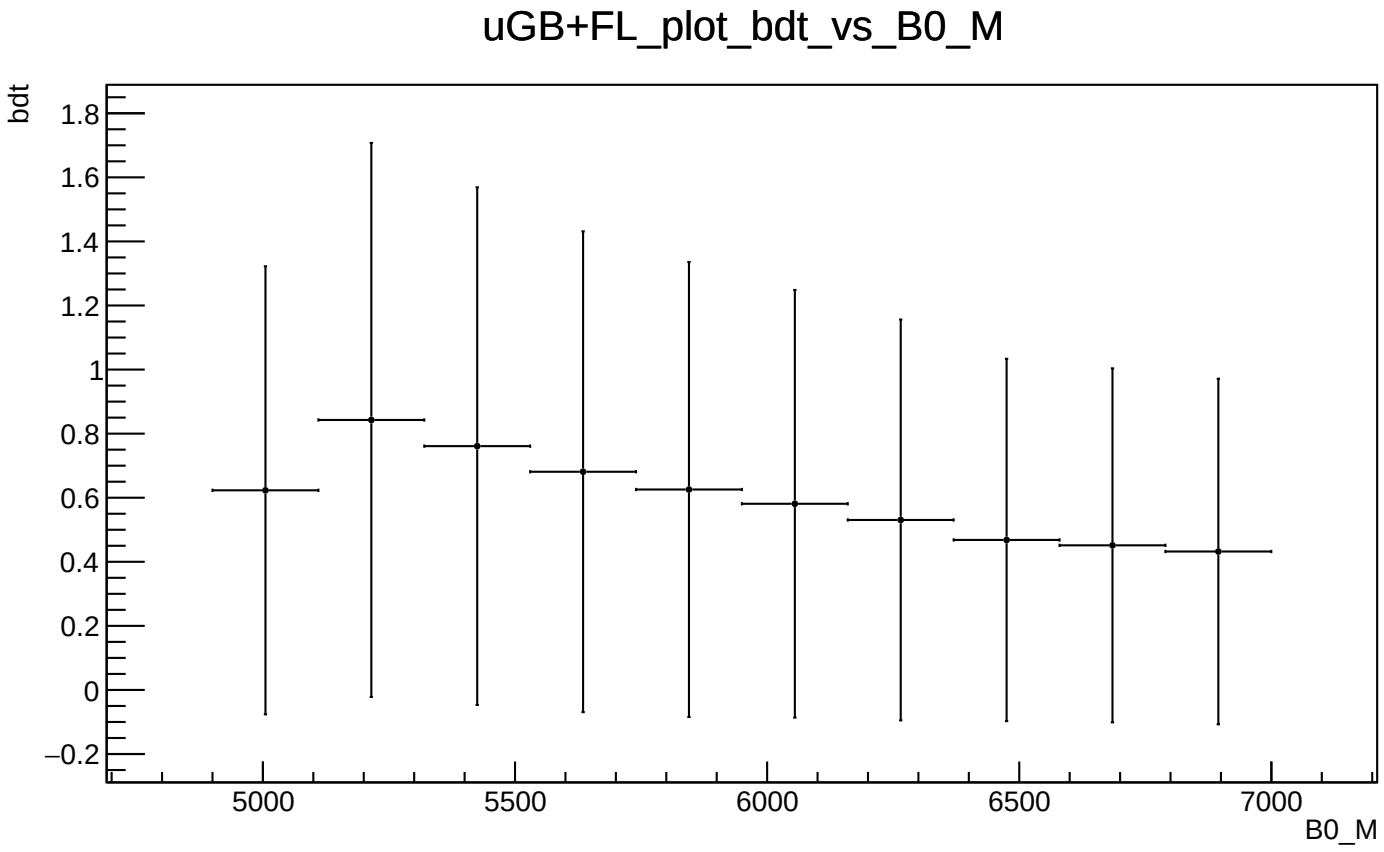

sk_bdt_plot_bdt_vs_B0_ThetaK
bdt
1
0.8
0.6
0.4
0.2
0
−1
−0.5
0
0.5
1
cos B0_ThetaK

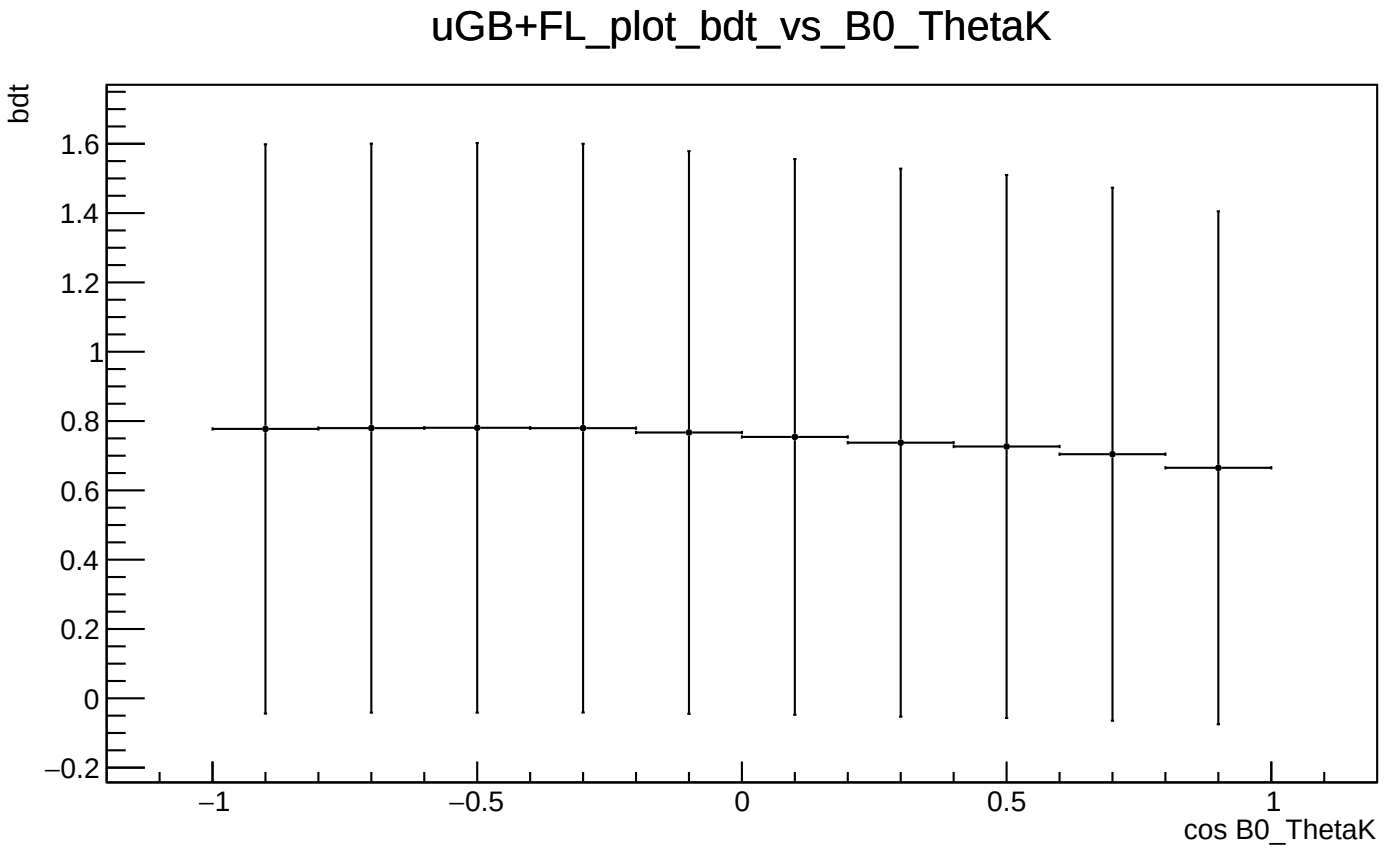

sk_bdt_plot_bdt_vs_B0_ThetaL
bdt
1
0.8
0.6
0.4
0.2
0
−1
−0.5
0
0.5
1
cos B0_ThetaL

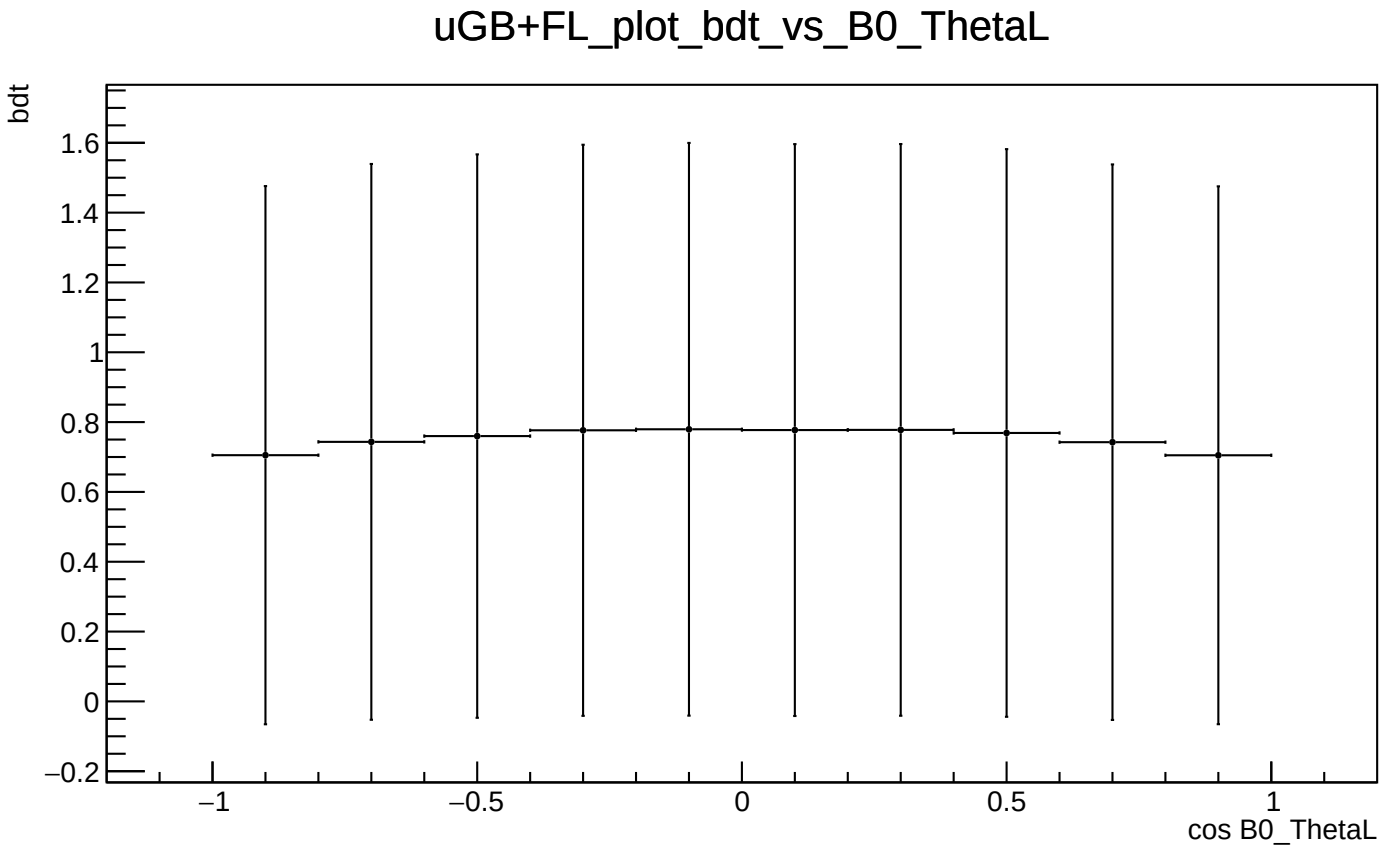

uBoost_plot_bdt_vs_B0_M
bdt
1.2
1
0.8
0.6
0.4
0.2
0
5000
5500
6000
6500
7000
B0_M

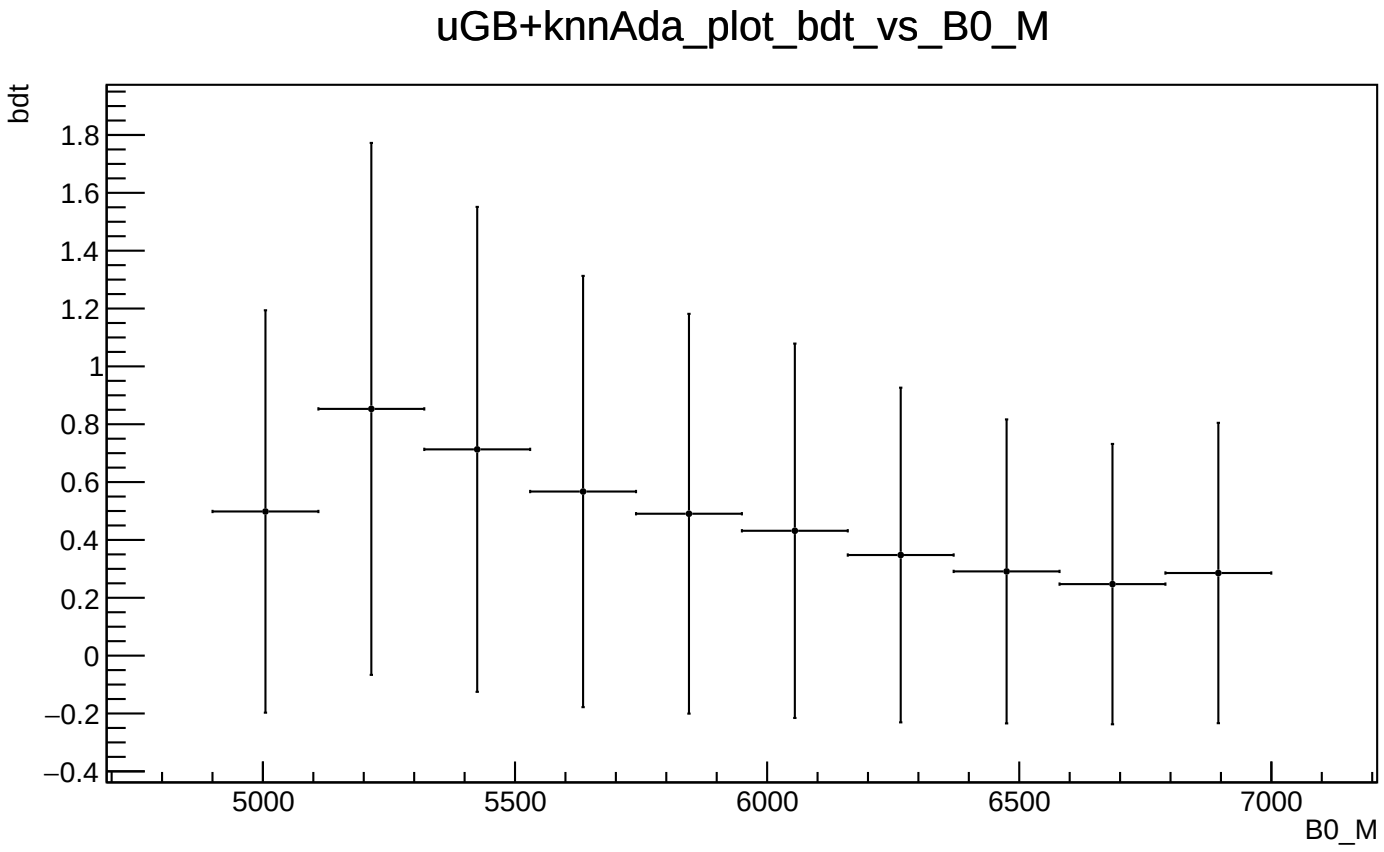

uBoost_plot_bdt_vs_B0_ThetaK
bdt
1.2
1
0.8
0.6
0.4
0.2
0
−1
−0.5
0
0.5
1
cos B0_ThetaK

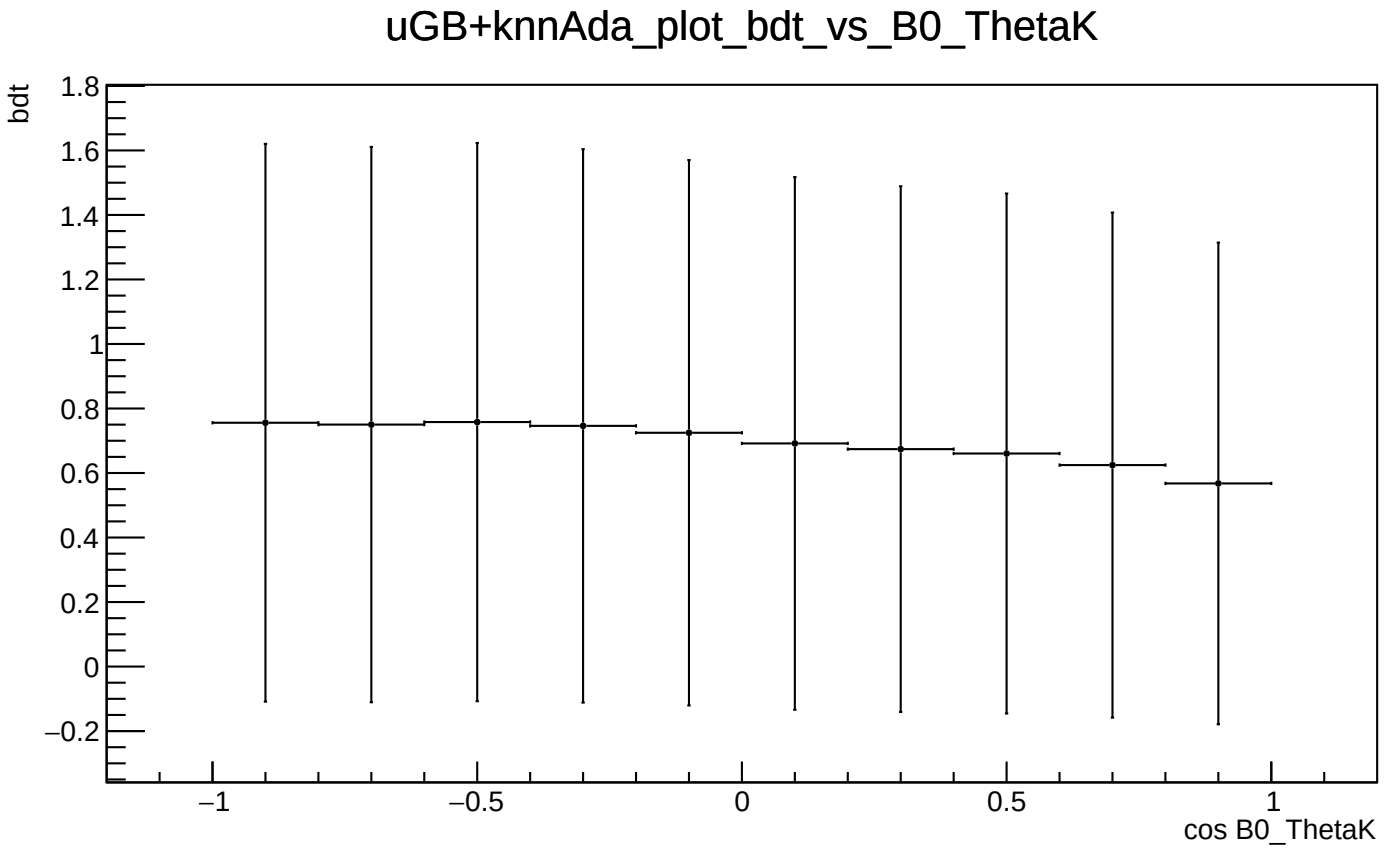

uBoost_plot_bdt_vs_B0_ThetaL
bdt
1.2
1
0.8
0.6
0.4
0.2
0
−1
−0.5
0
0.5
1
cos B0_ThetaL

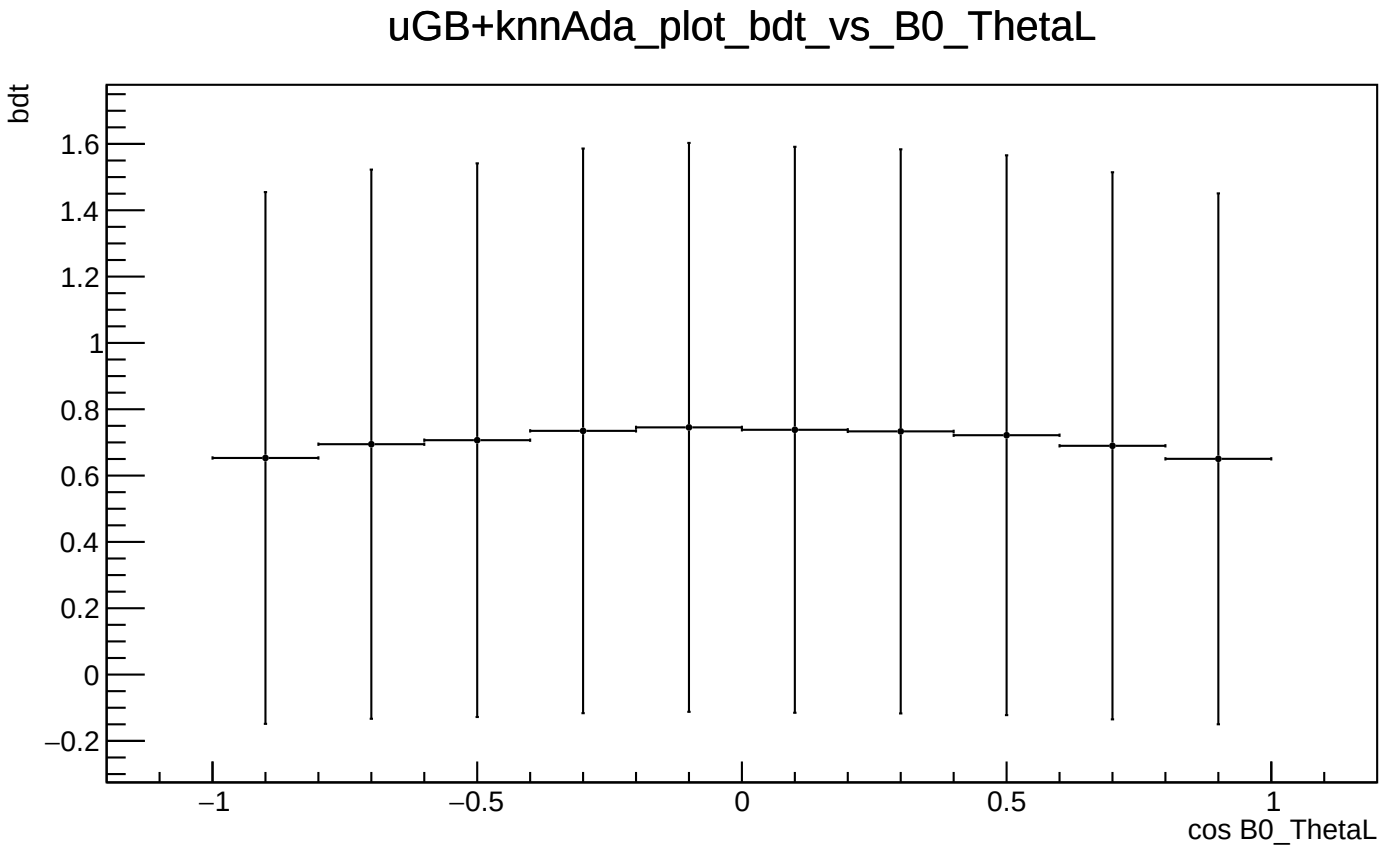

uGB+FL_plot_bdt_vs_B0_M
bdt
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
−0.2
5000
5500
6000
6500
7000
B0_M

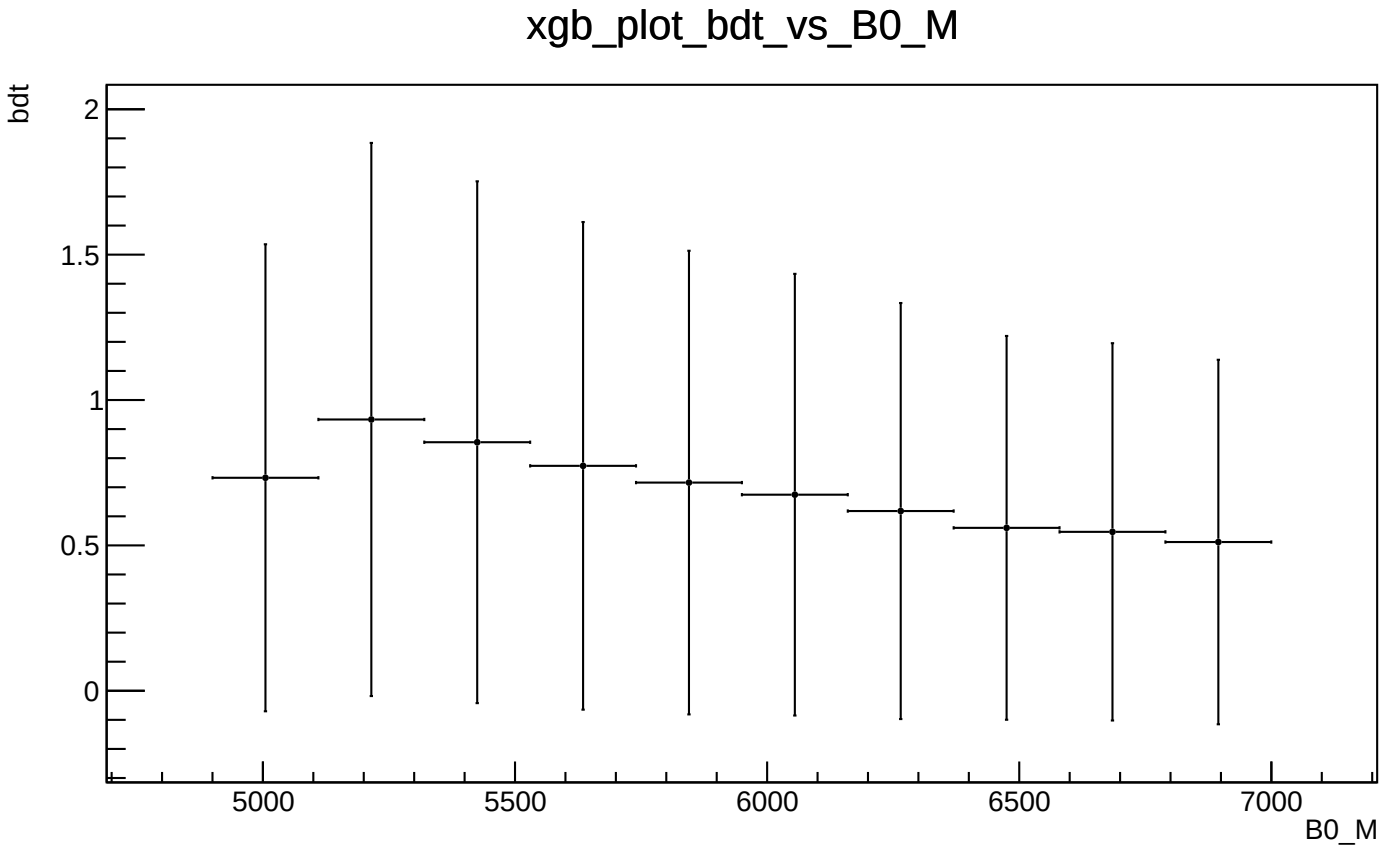

uGB+FL_plot_bdt_vs_B0_ThetaK
bdt
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
−0.2
−1
−0.5
0
0.5
1
cos B0_ThetaK

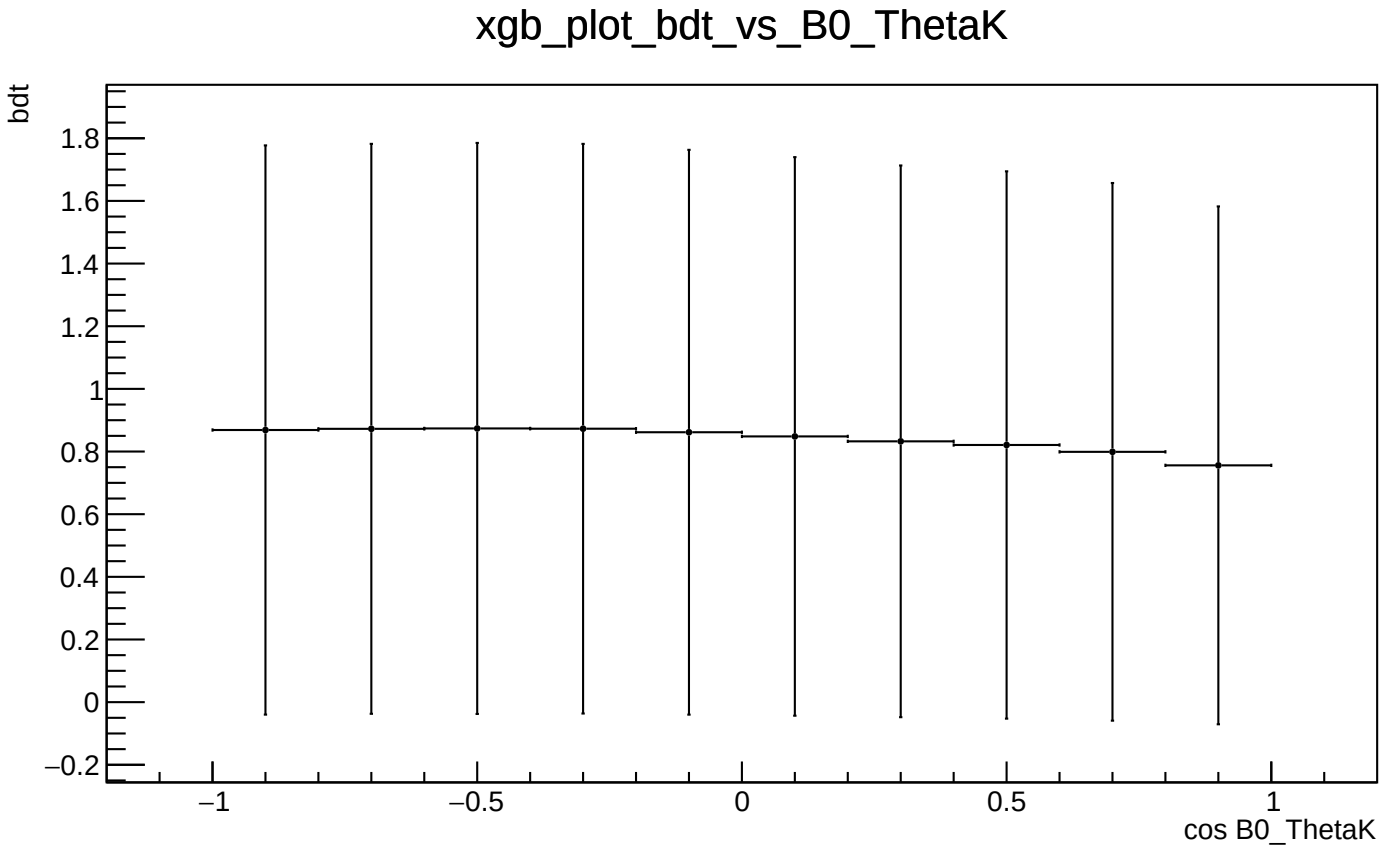

uGB+FL_plot_bdt_vs_B0_ThetaL
bdt
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
−0.2
−1
−0.5
0
0.5
1
cos B0_ThetaL

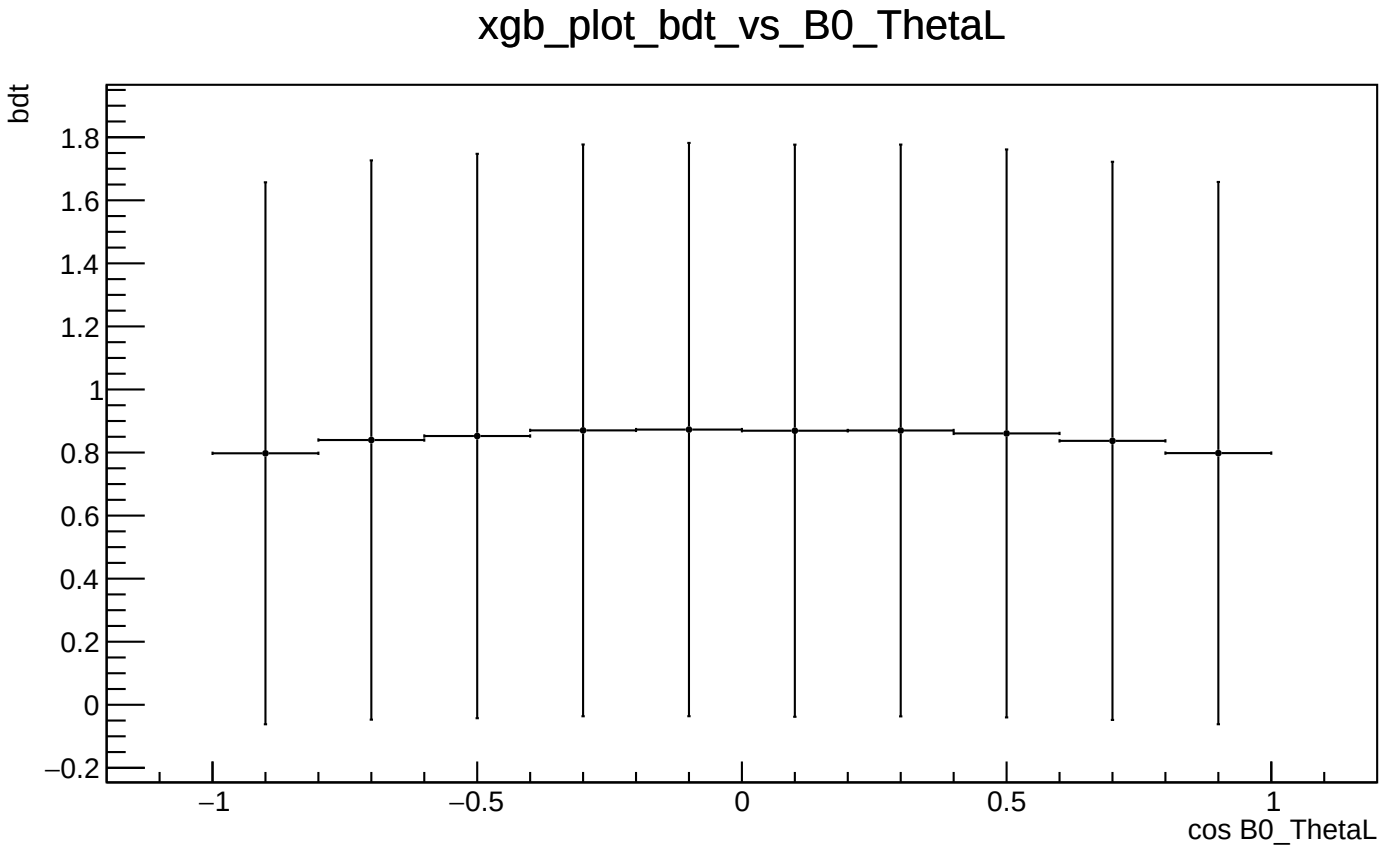

uGB+knnAda_plot_bdt_vs_B0_M
bdt
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
5000
5500
6000
6500
7000
B0_M


uGB+knnAda_plot_bdt_vs_B0_ThetaK
bdt
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
−0.2
−1
−0.5
0
0.5
1
cos B0_ThetaK

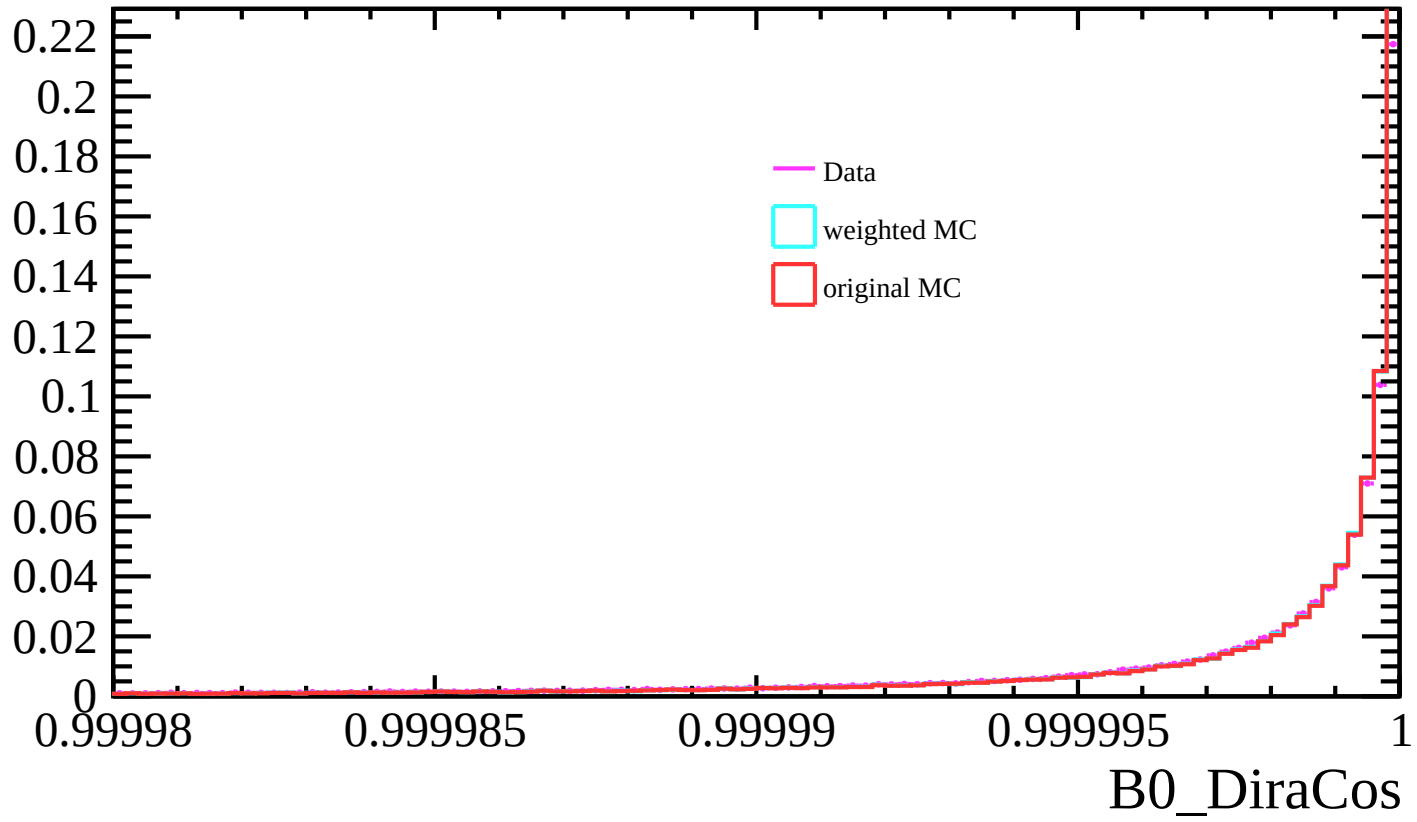

uGB+knnAda_plot_bdt_vs_B0_ThetaL
bdt
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
−0.2
−1
−0.5
0
0.5
1
cos B0_ThetaL

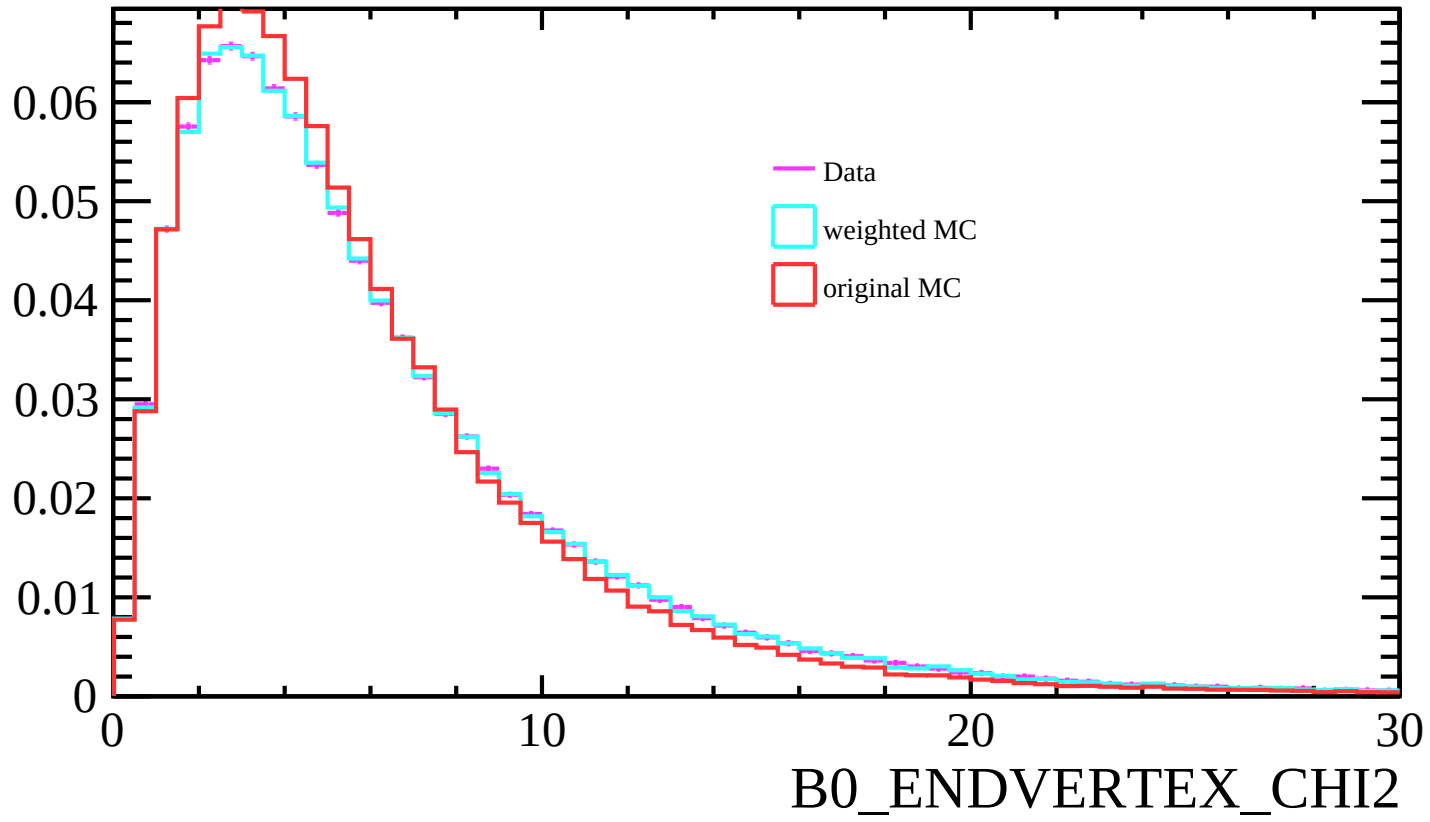

xgb_plot_bdt_vs_B0_M
bdt
2
1.5
1
0.5
0
5000
5500
6000
6500
7000
B0_M

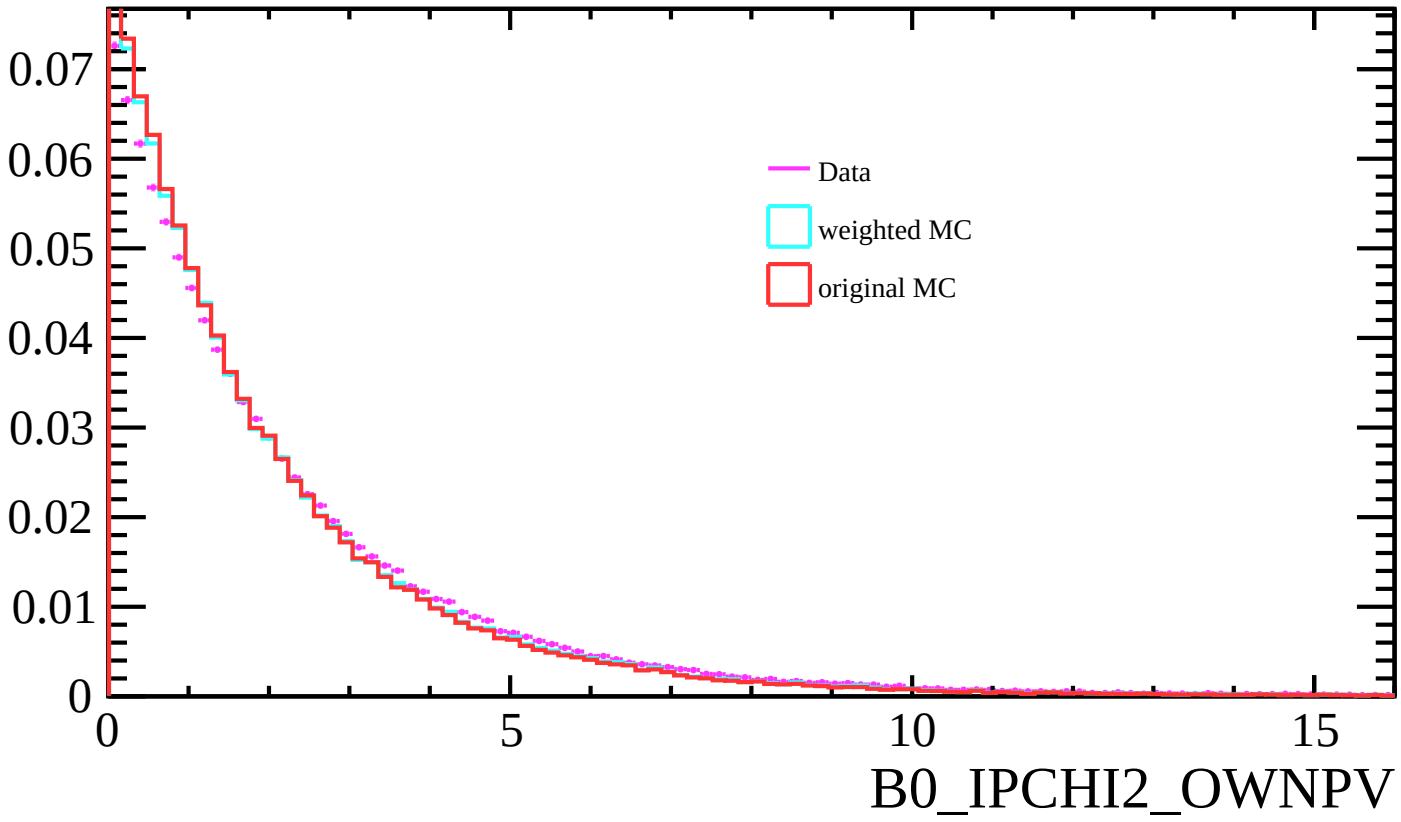

xgb_plot_bdt_vs_B0_ThetaK
bdt
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
−0.2
−1
−0.5
0
0.5
1
cos B0_ThetaK

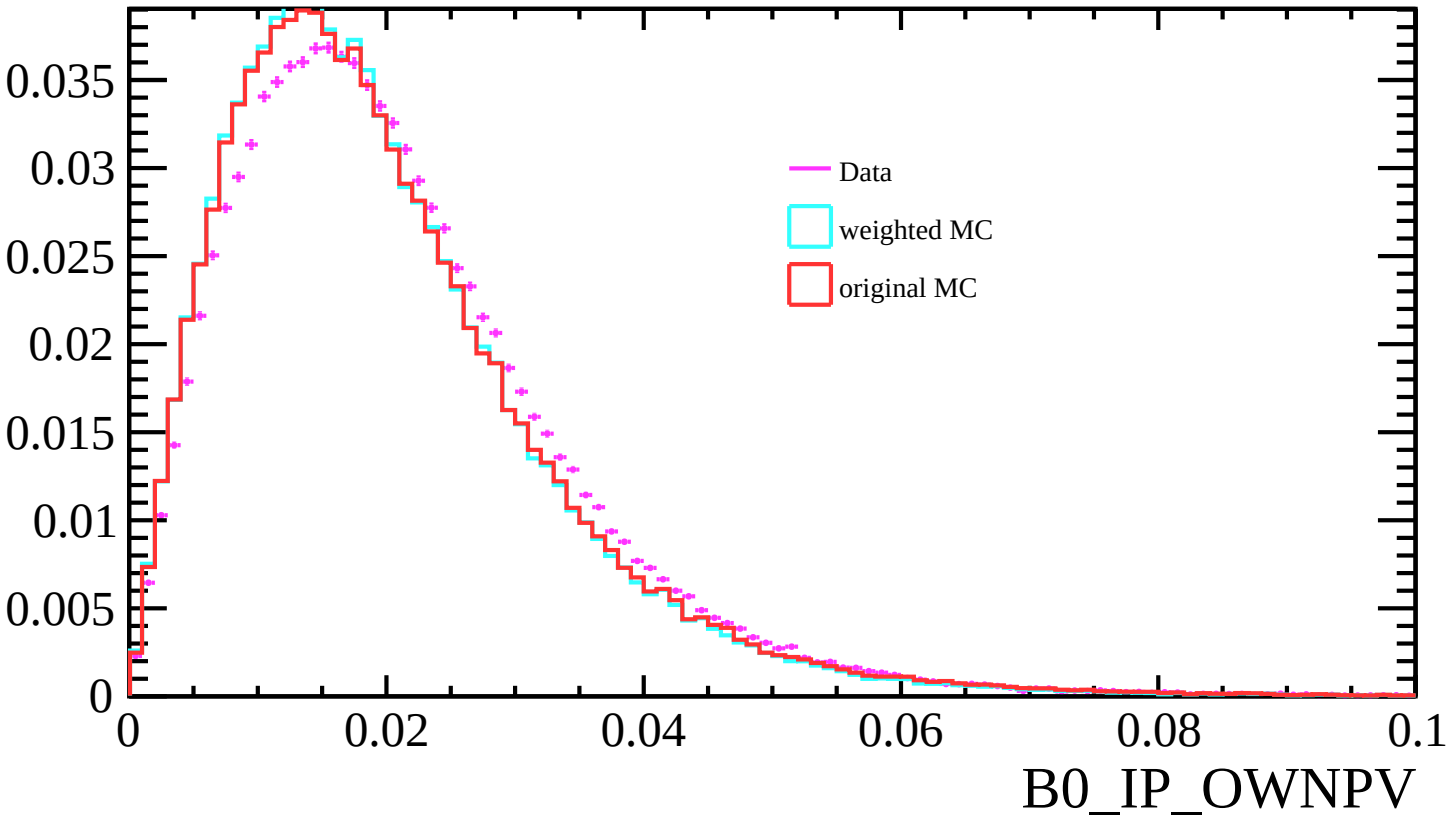

xgb_plot_bdt_vs_B0_ThetaL
bdt
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
−0.2
−1
−0.5
0
0.5
1
cos B0_ThetaL

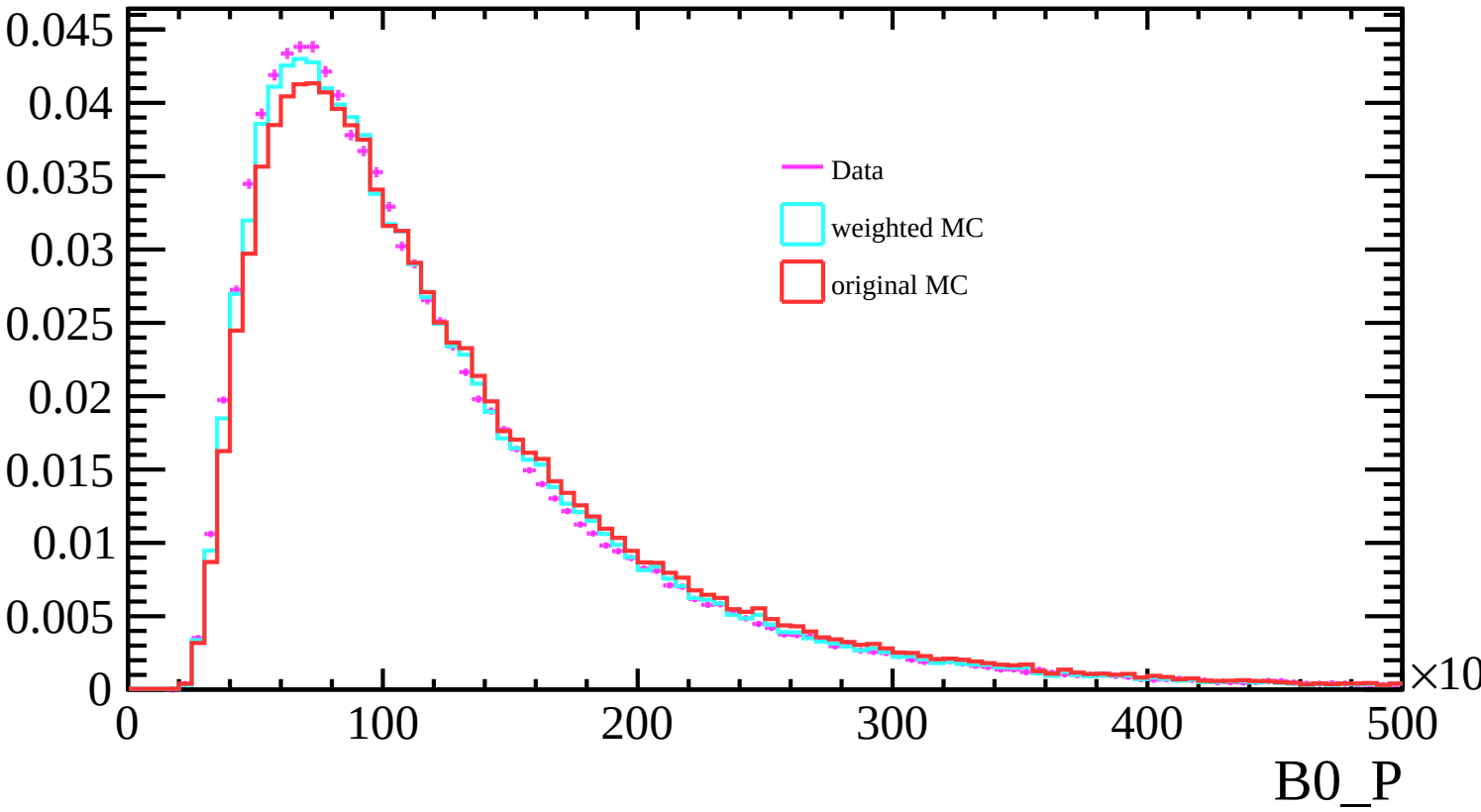
C. Reweighting plots

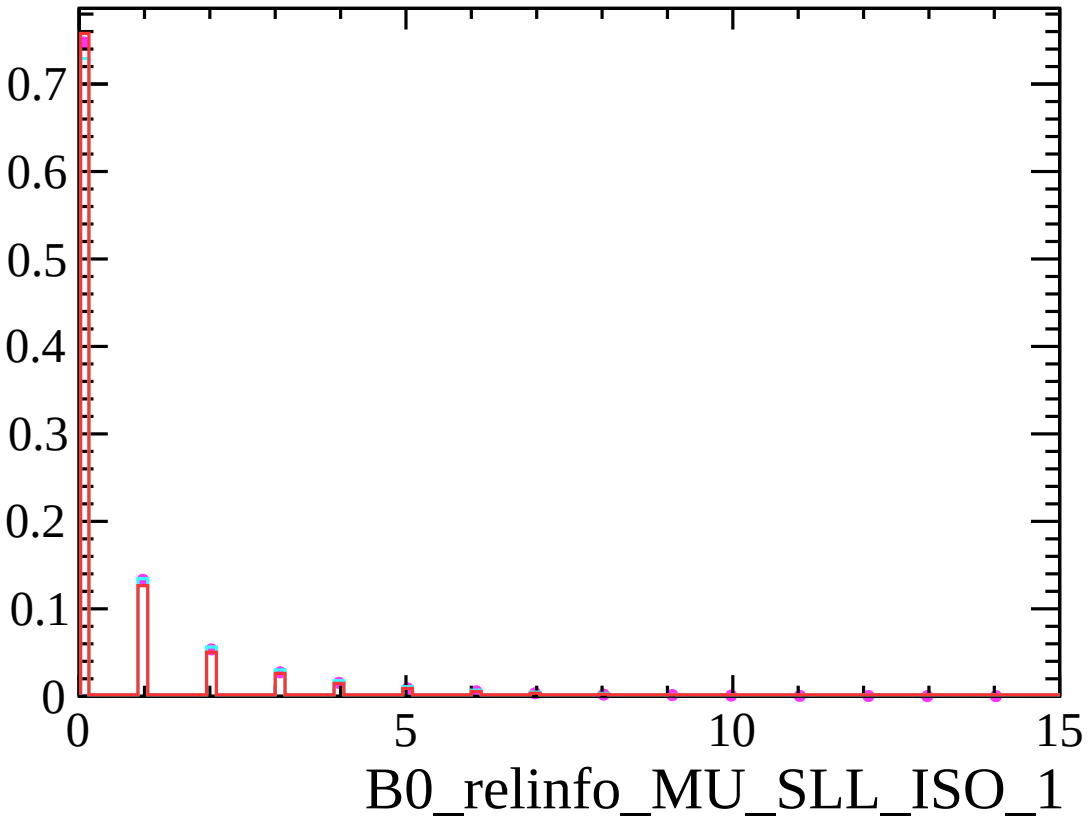

0.22
0.2
0.18
0.16
0.14
0.12
0.1
0.08
0.06
0.04
0.02
0
0.99998
0.999985
0.99999
0.999995
1
B0_DiraCos
Data
weighted MC
original MC

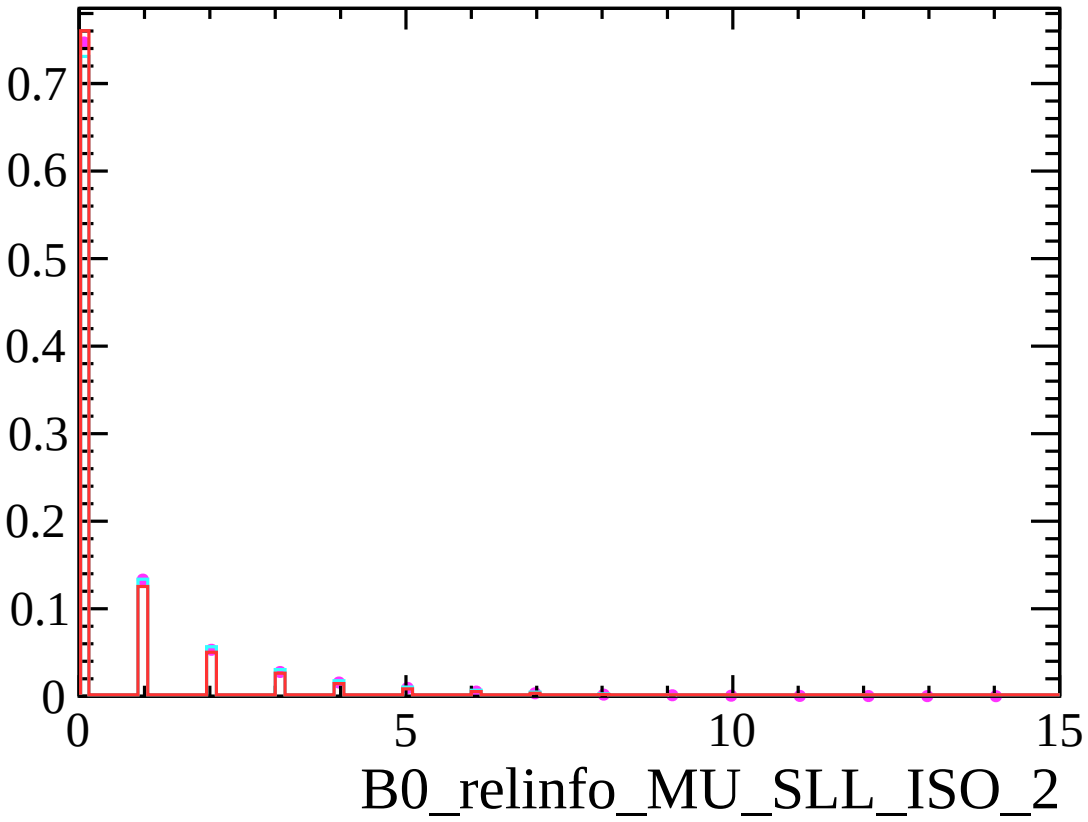

0.06
0.05
0.04
0.03
0.02
0.01
0
0
10
20
30
Data
weighted MC
original MC
B0_ENDVERTEX_CHI2

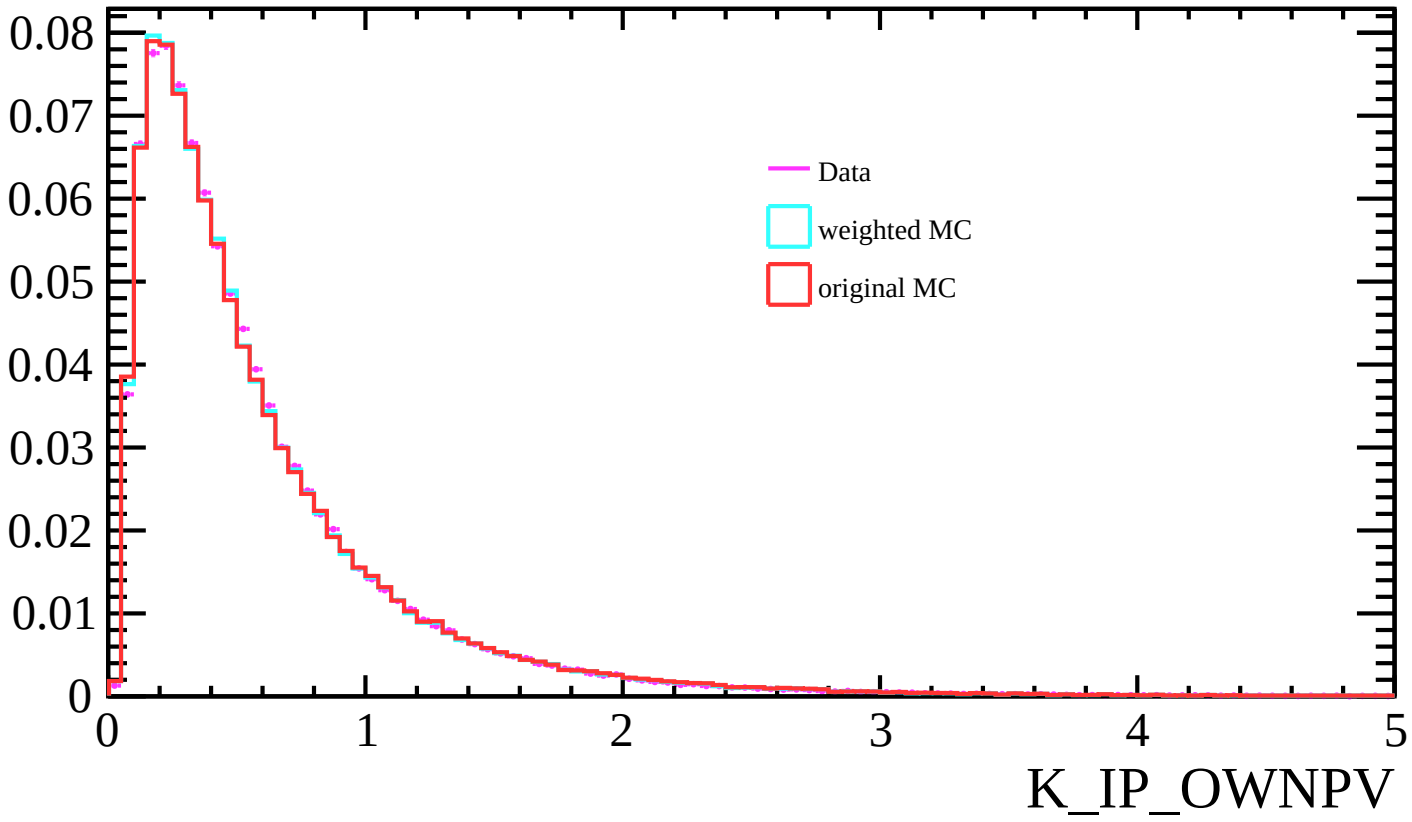

0.07
0.06
0.05
0.04
0.03
0.02
0.01
0
0
5
10
15
B0_IPCHI2_OWNPV
Data
weighted MC
original MC

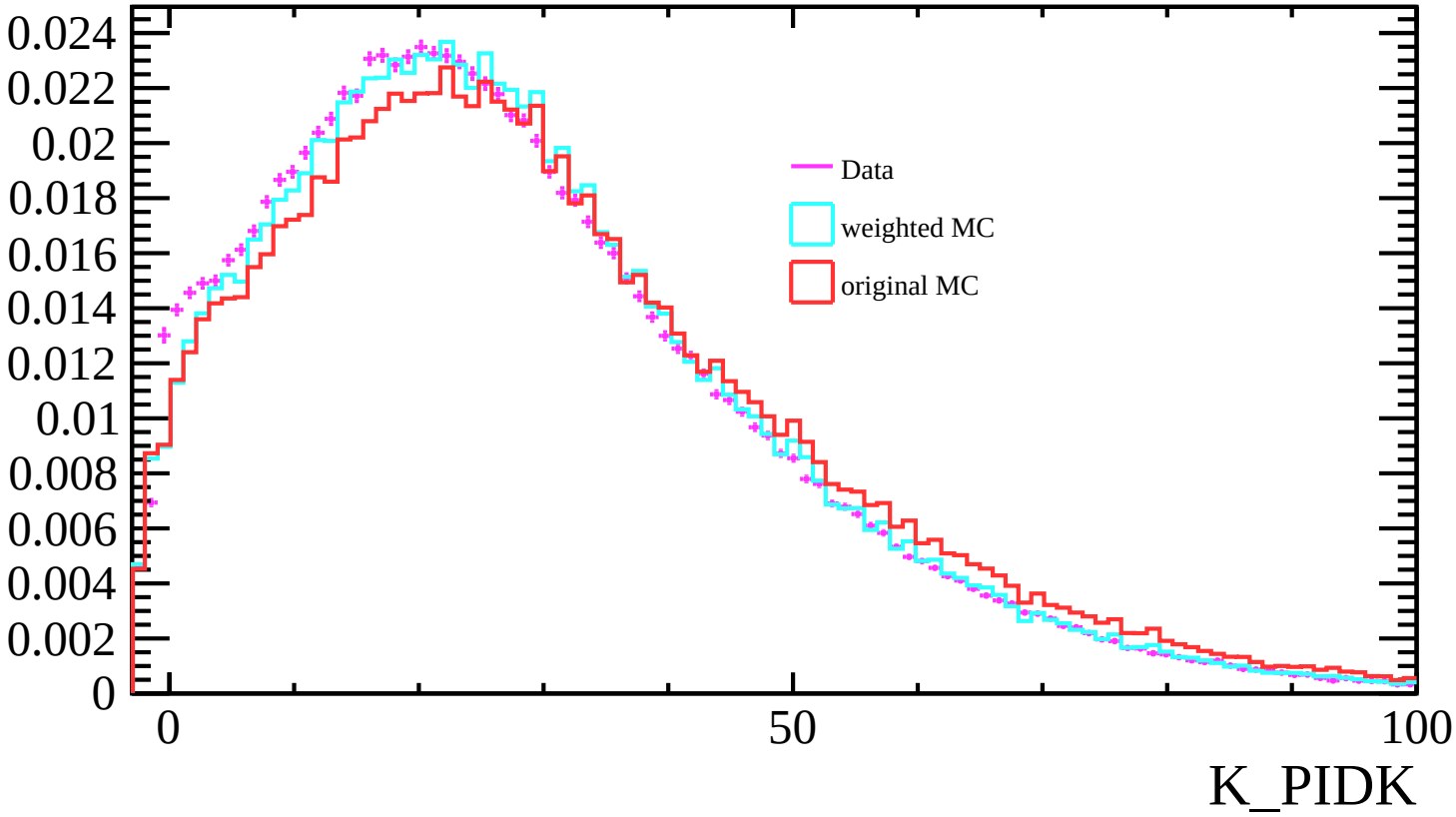

0.035
0.03
0.025
0.02
0.015
0.01
0.005
0
0
0.02
0.04
0.06
0.08
0.1
B0_IP_OWNPV
Data
weighted MC
original MC

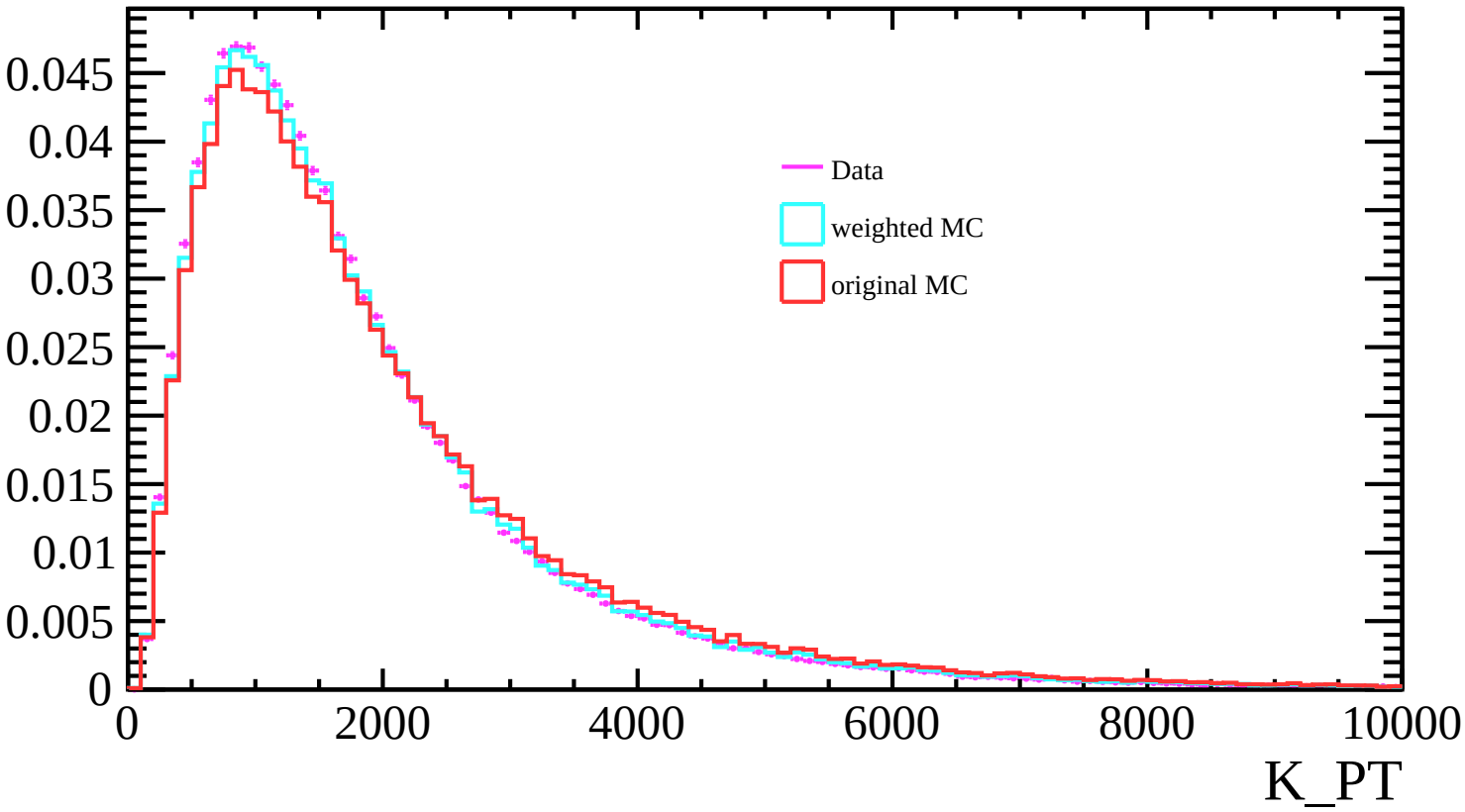

0.045
0.04
0.035
0.03
0.025
0.02
0.015
0.01
0.005
0
0
100
200
300
400
500
×10
B0_P
Data
weighted MC
original MC

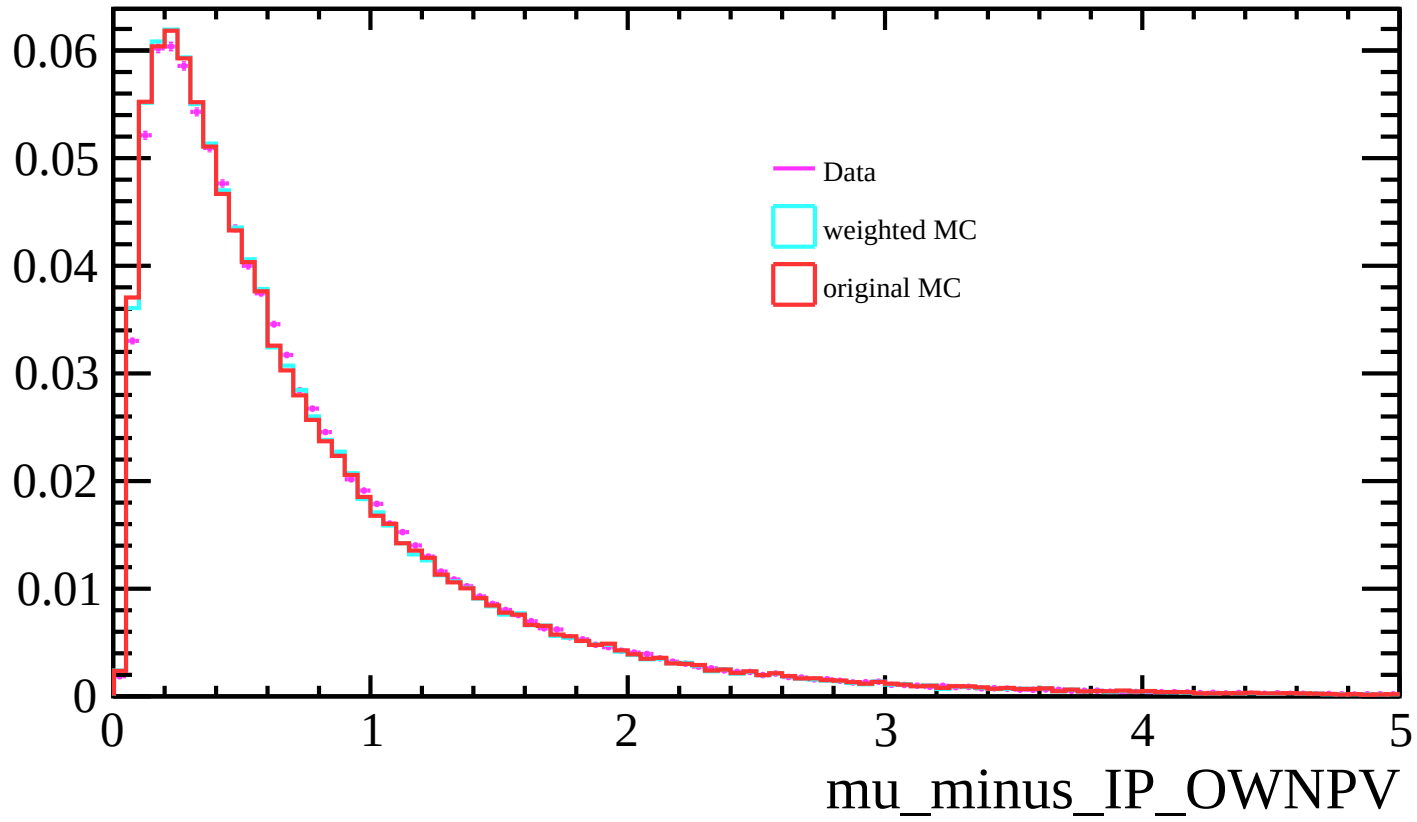

0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
5
10
15
B0_relinfo_MU_SLL_ISO_1

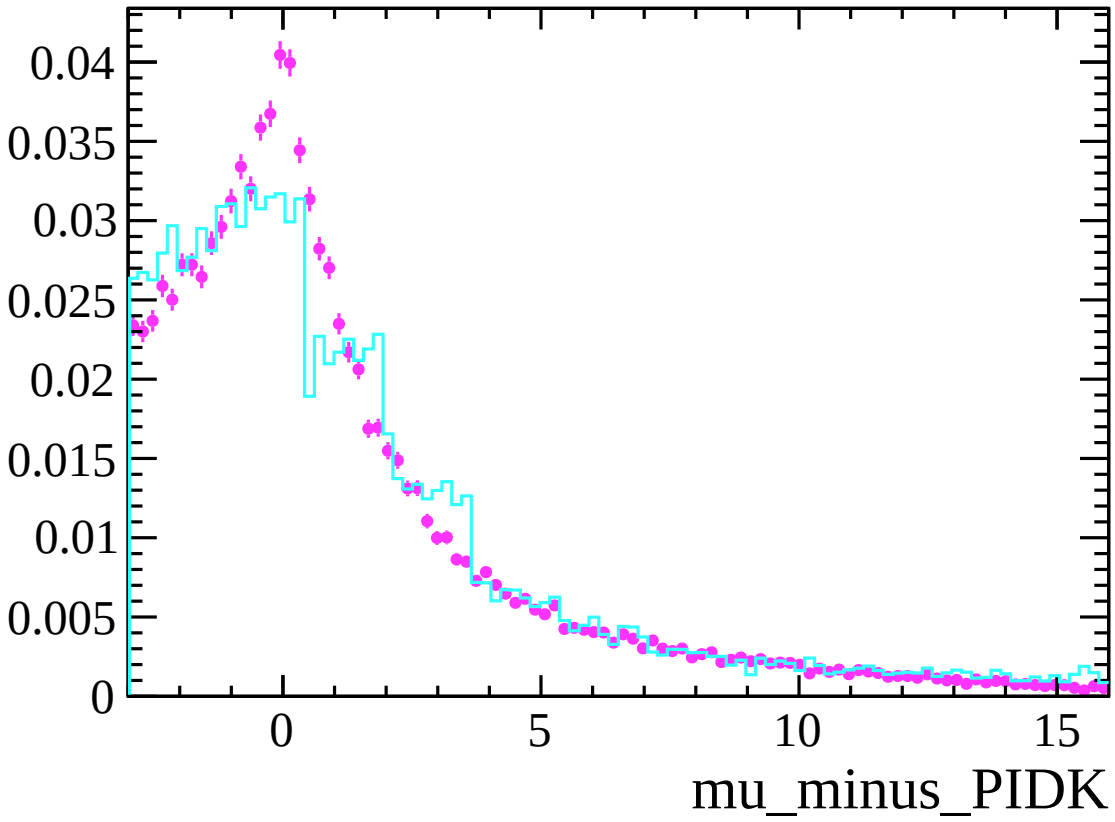

0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
5
10
15
B0_relinfo_MU_SLL_ISO_2

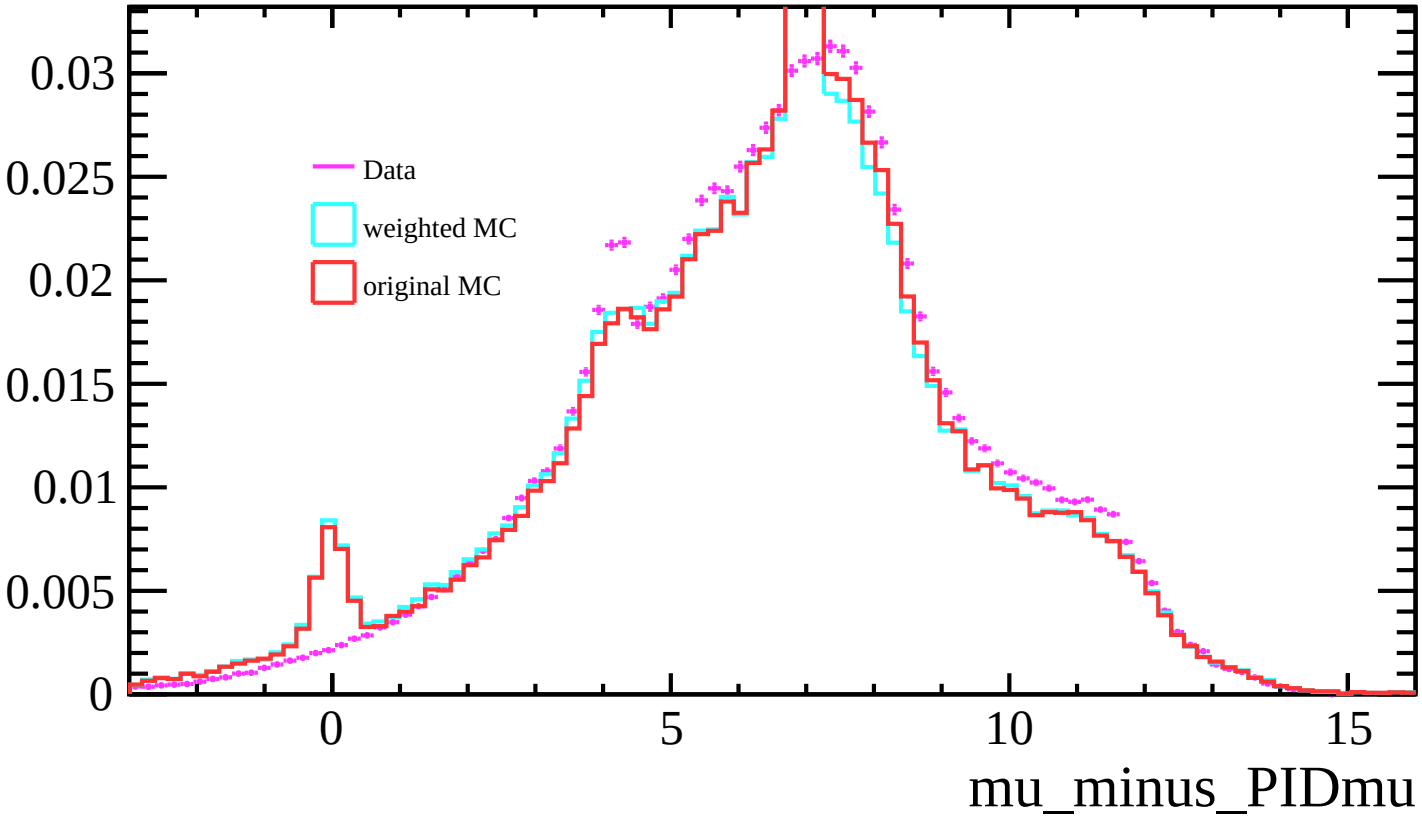

0.08
0.07
0.06
0.05
0.04
0.03
0.02
0.01
0
0
1
2
3
4
5
K_IP_OWNPV
Data
weighted MC
original MC

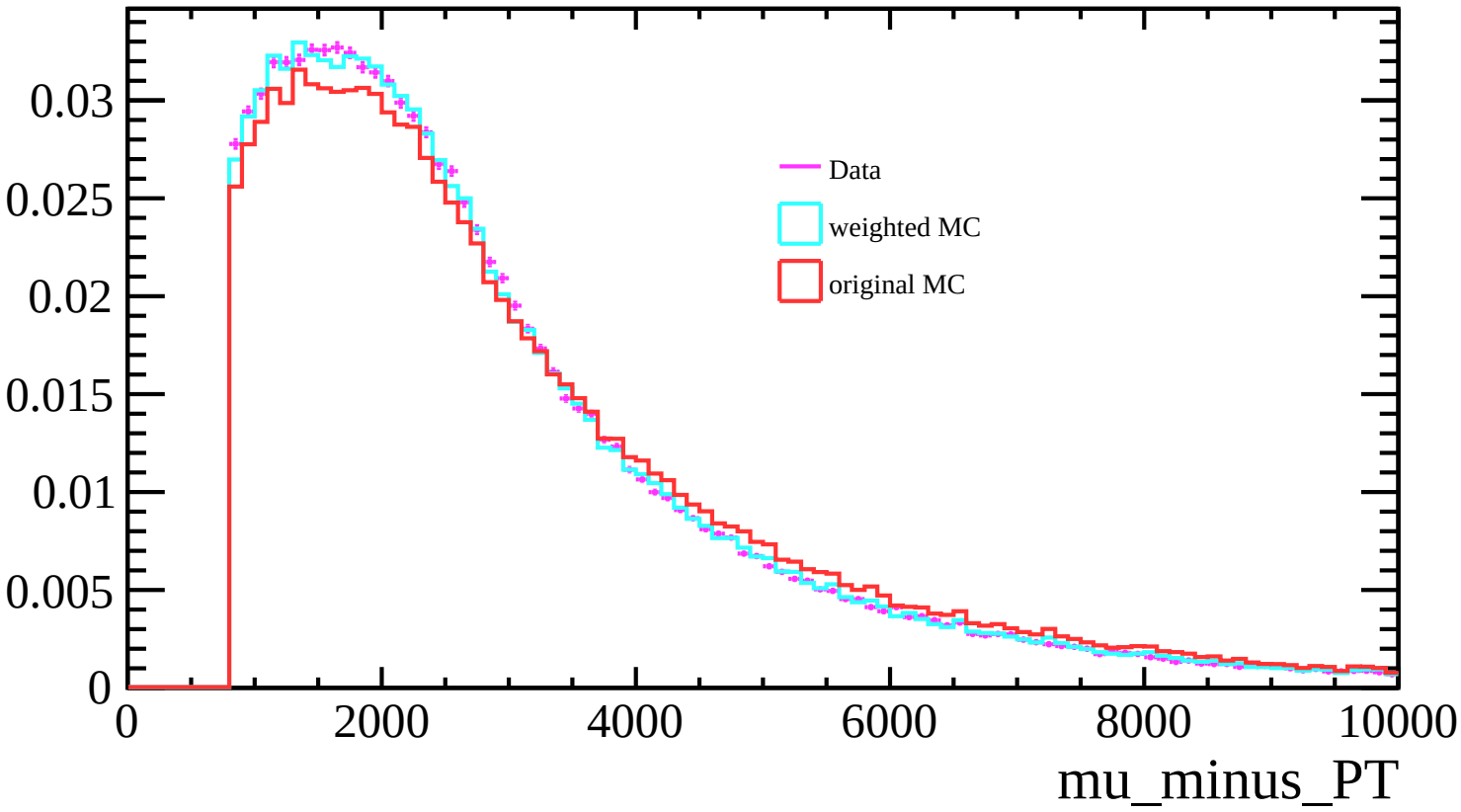

0.024
0.022
0.02
0.018
0.016
0.014
0.012
0.01
0.008
0.006
0.004
0.002
0
0
50
100
K_PIDK
Data
weighted MC
original MC

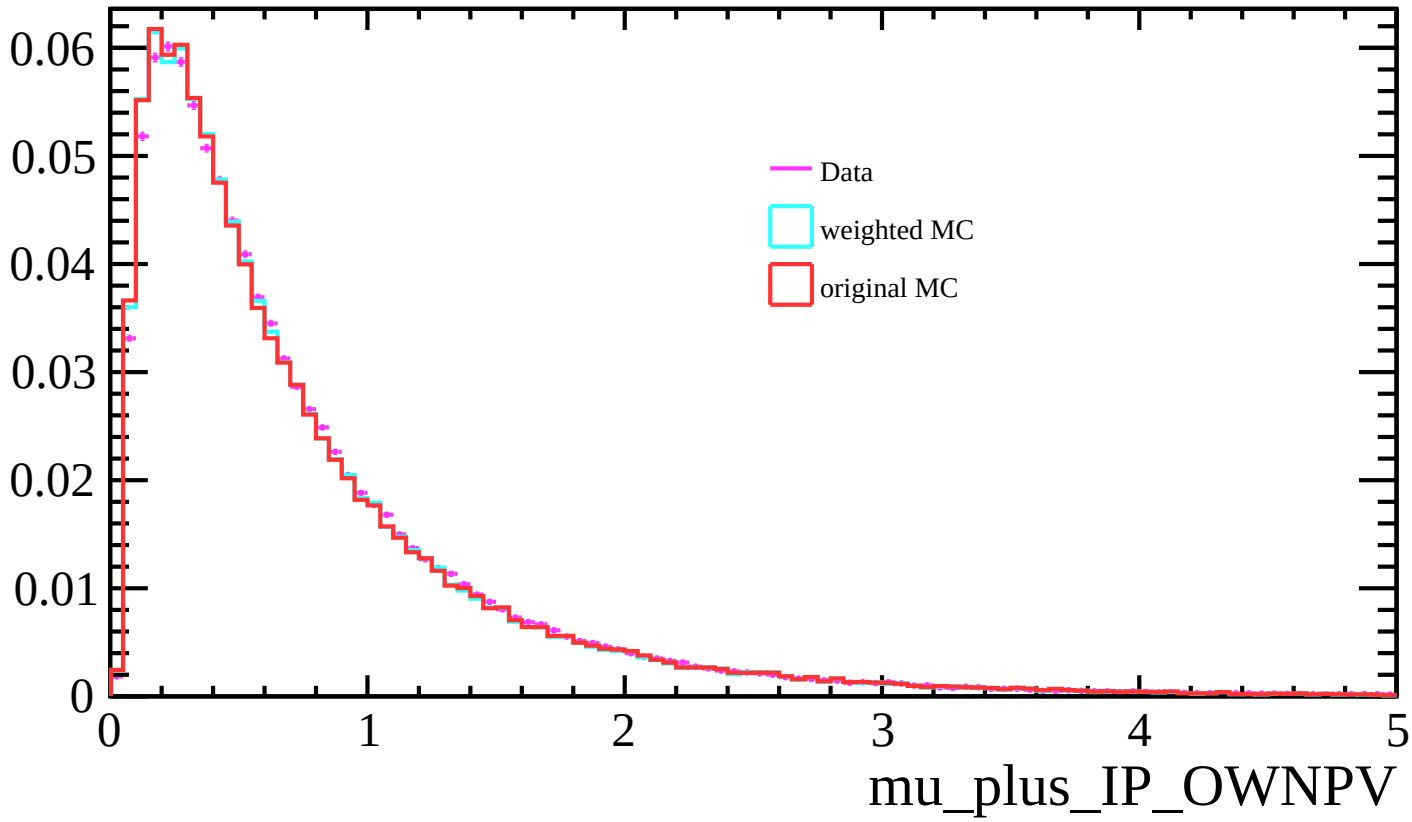

0.045
0.04
0.035
0.03
0.025
0.02
0.015
0.01
0.005
0
0
2000
4000
6000
8000
10000
K_PT
Data
weighted MC
original MC

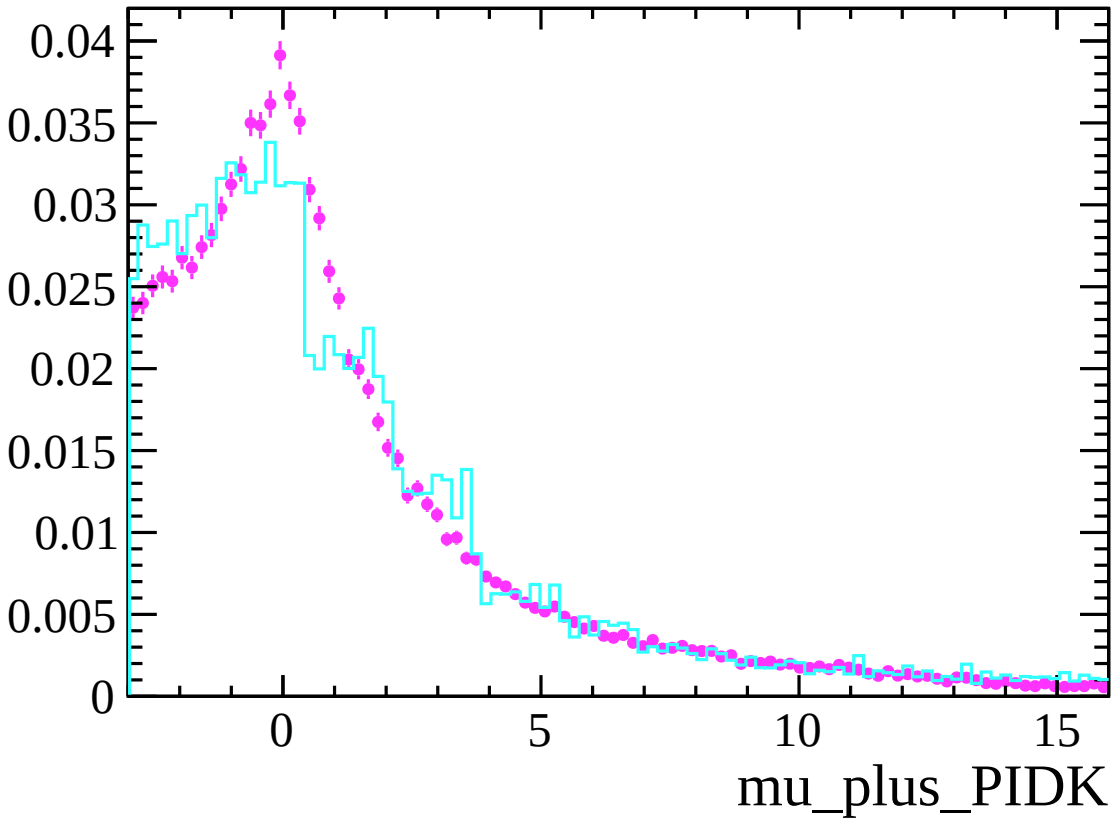

0.06
0.05
0.04
0.03
0.02
0.01
0
0
1
2
3
4
5
Data
weighted MC
original MC
mu_minus_IP_OWNPV

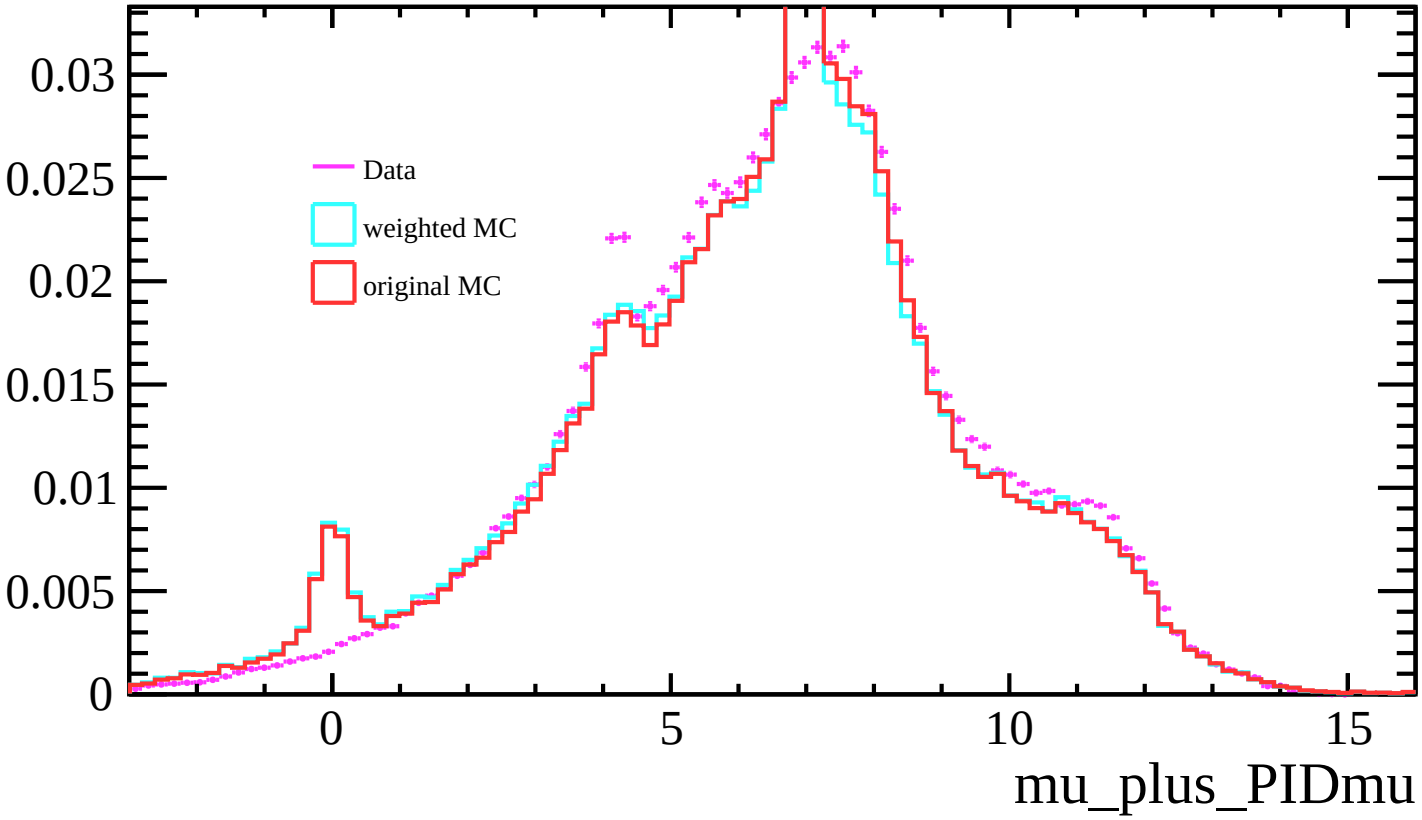

0.04
0.035
0.03
0.025
0.02
0.015
0.01
0.005
0
0
5
10
15
mu_minus_PIDK

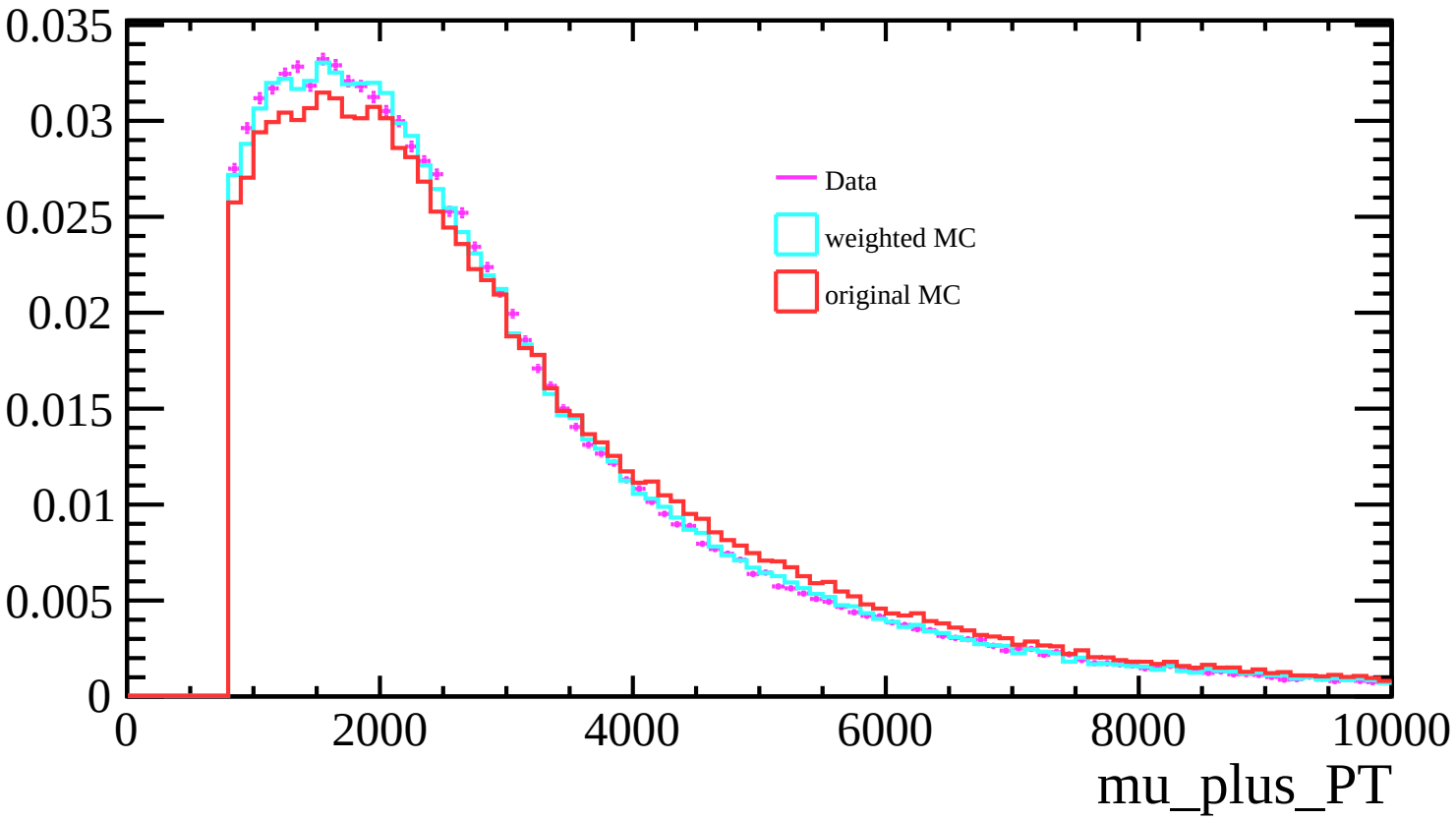

0.03
0.025
0.02
0.015
0.01
0.005
0
Data
weighted MC
original MC
0
5
10
15
mu_minus_PIDmu

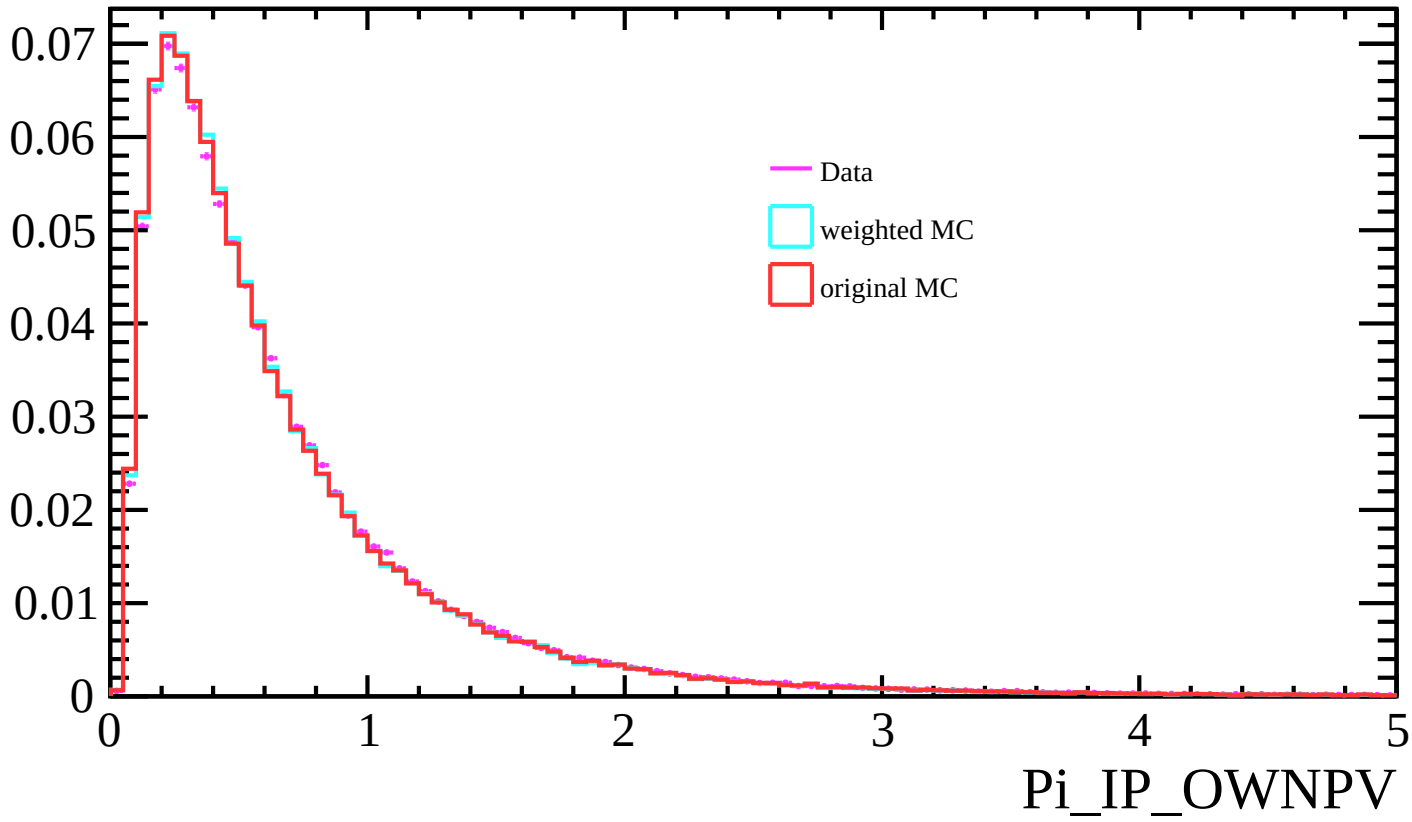

0.03
0.025
0.02
0.015
0.01
0.005
0
0
2000
4000
6000
8000
10000
mu_minus_PT
Data
weighted MC
original MC

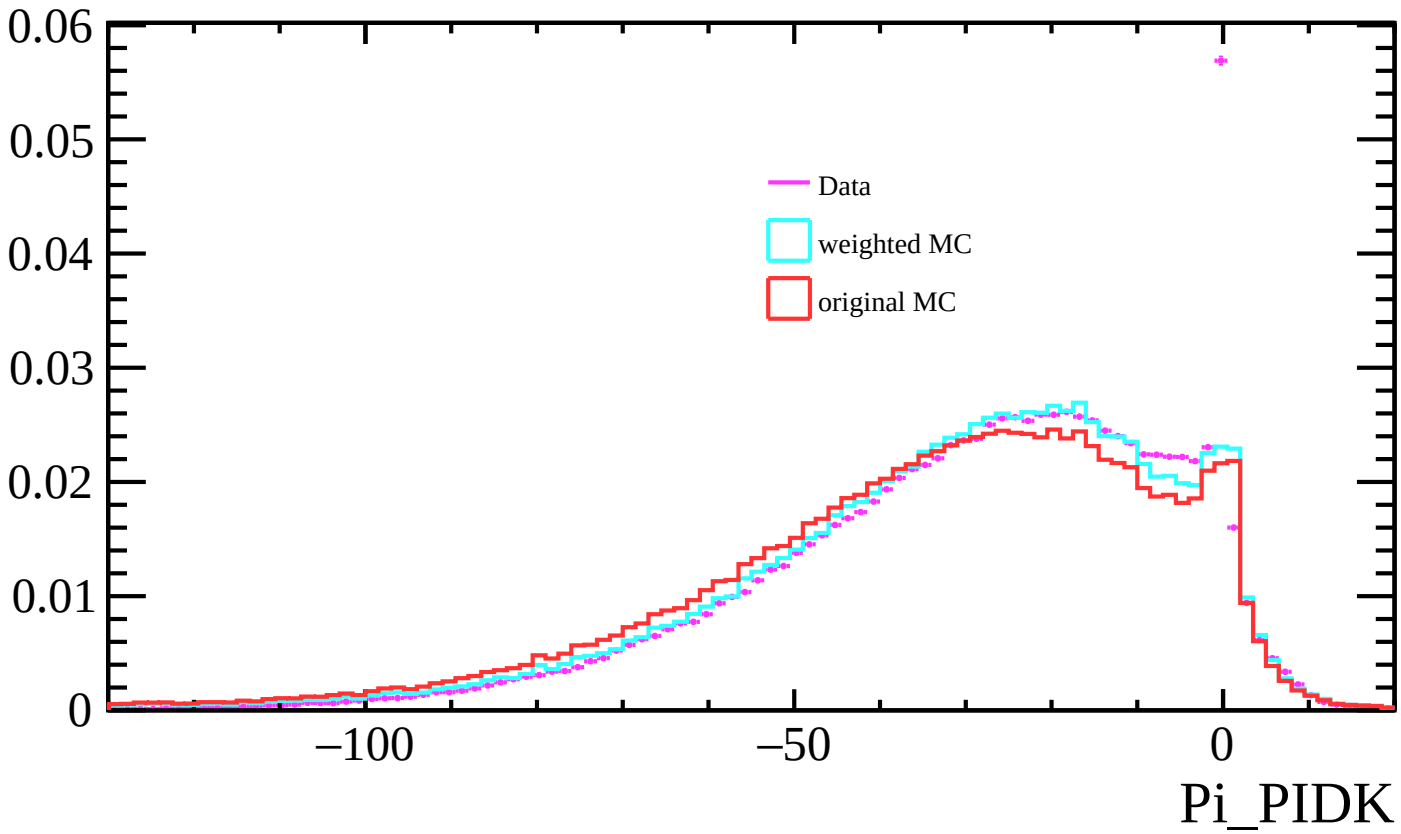

0.06
0.05
0.04
0.03
0.02
0.01
0
0
1
2
3
4
5
Data
weighted MC
original MC
mu_plus_IP_OWNPV

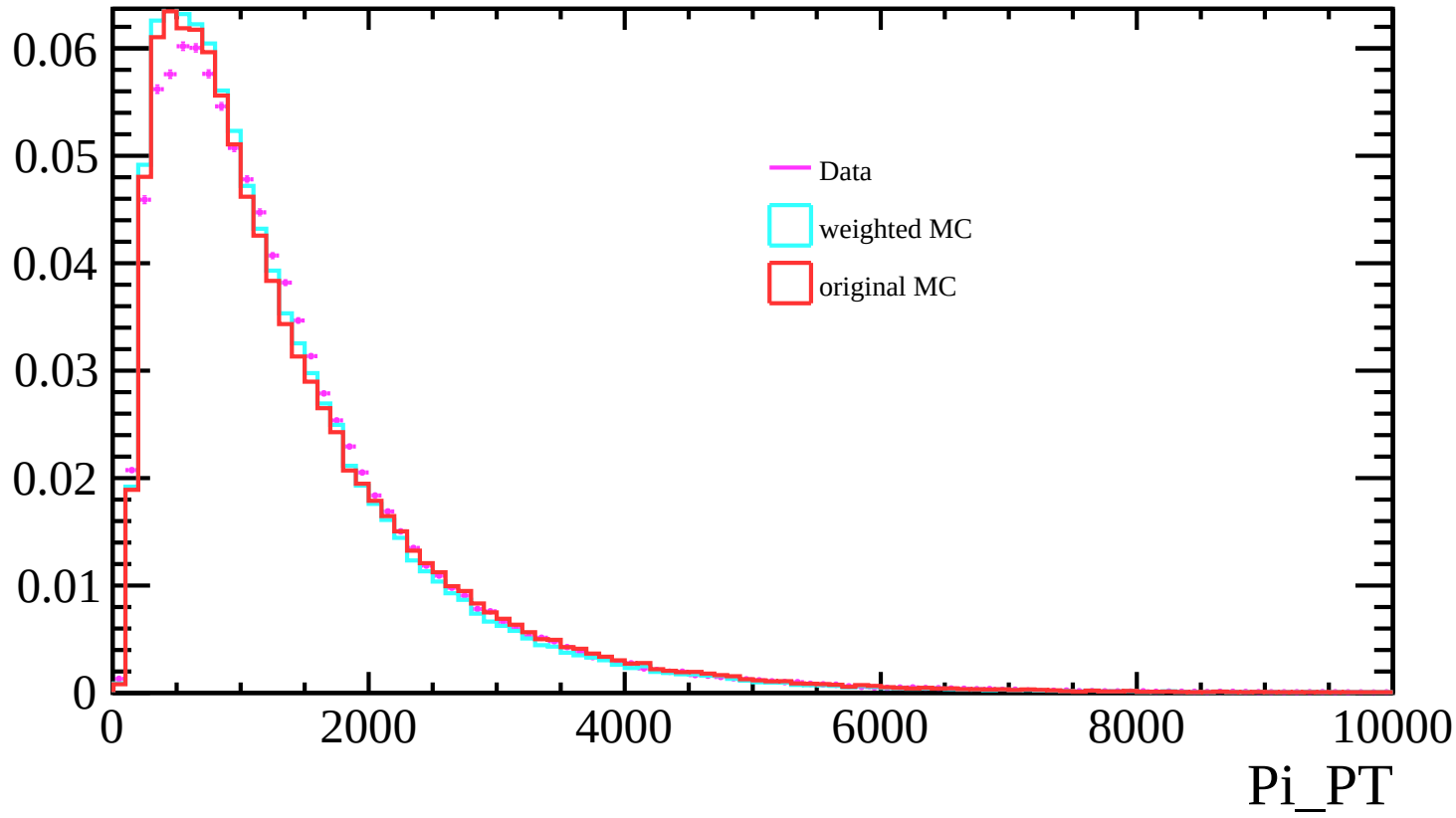

0.04
0.035
0.03
0.025
0.02
0.015
0.01
0.005
0
0
5
10
15
mu_plus_PIDK


0.03
0.025
0.02
0.015
0.01
0.005
0
Data
weighted MC
original MC
0
5
10
15
mu_plus_PIDmu


0.035
0.03
0.025
0.02
0.015
0.01
0.005
0
0
2000
4000
6000
8000
10000
mu_plus_PT
Data
weighted MC
original MC


0.07
0.06
0.05
0.04
0.03
0.02
0.01
0
0
1
2
3
4
5
Pi_IP_OWNPV
Data
weighted MC
original MC


0.06
0.05
0.04
0.03
0.02
0.01
0
−100
−50
0
Pi_PIDK
Data
weighted MC
original MC


0.06
0.05
0.04
0.03
0.02
0.01
0
0
2000
4000
6000
8000
10000
Pi_PT
Data
weighted MC
original MC