The Widget

Jake Bennett

April 24, 2018

The Widget is a tool for performing and testing the dE/dx reconstruction and calibration procedure at BESIII and Belle II. Its modular design is intended to be useful for various tests as well as for obtaining the actual calibration constants for a data sample.

1 Introduction

While the primary purpose of the Widget is to enable testing and development of calibration procedures without the need to reprocess data samples, it is also useful to actually perform the dE/dx calibration tasks for data samples. Therefore, the structure of the input samples must be similar to that used for the BESIII calibration, which uses the *TupleWrite* package in BOSS. Belle II samples are prepared by using a basf2 module called *HitLevelInfoWriter* to convert mDST files to ROOT files that contain the hit information. For example, the input Widget files have branches containing things like the momentum, $\cos\theta$, and number of ADC counts for each track.

The structure of the Widget includes three primary classes and several ancillary classes. The primary structures include the HadronCalibration, WidgetPrep, and WidgetGenerator classes. Ancillary clases include objects like WidgetParameterization, and HadronSaturation. These are described in detail below. These classes are called from the WidgetInterface class, which defines low level parameters like the range of $\beta\gamma$ and $\cos\theta$.

Example executables are given in the WidgetExe directory. The *HadronCalibration* executable performs the full hadron calibration procedure, with the option of only running one or more individual steps based on command line input. The input to the Widget may be supplied in a ROOT file. If no input file is given, the events are generated according to a Gaussian distribution. In either case, the dE/dx values may be modified to introduce the hadron saturation effect. The paths to input files and various Widget parameters are specified via a configuration file.

The *HadronCalibration* examples generates or reads in a set of hadron samples, performs fits to the dE/dx truncated means in bins of $\beta\gamma$ and $\cos\theta$ for each particle type, and fits the resulting means as a function of $\cos\theta$ in bins of $\beta\gamma$. The results of the fits to the truncated means as a function of $\cos\theta$ are the five hadron saturation parameters. These values are then returned to the WidgetPrep class and the dE/dx values are adjusted according to the fitted values. Then the fits to the means are repeated and the resulting spectrum (which should be flat as a function of $\cos\theta$) is plotted.

The *run* directory contains a shell script (runWidget.sh) that sets environment variables, compiles the Widget and runs the executables based on user input. Users must first set the *$ROOTSYS* environment variable to point to their installation of ROOT (e.g. /usr/local/root).

2 Primary Classes

2.1 WidgetGenerator

The WidgetGenerator class is responsible for generating event samples and/or modifying the dE/dx values of a given sample. The input ROOT file should contain branches for the truncated mean, momentum, and angle for a given track. The output of the WidgetGenerator class is a ROOT file containing this same

information. If desired, the dE/dx measurements are replaced with a distribution generated according to the inverse of the hadron saturation parameterization for a certain set of constants.

The WidgetGenerator class also contains methods to recreate the dE/dx reconstruction. That is, it takes the hit level information and constructs truncated mean distributions. The input ROOT file is generated using the *HitLevelInfoWriter* module in the Belle II analysis framework (basf2) and the *TupleWrite* package in BOSS for BESIII.

2.2 WidgetPrep

The output of the WidgetGenerator class (or some prepared data samples) may be passed into the WidgetPrep class, which is responsible for fitting the dE/dx measurements in bins of $\beta\gamma$ and $\cos\theta$. The output of the WidgetPrep class is a ROOT file containing the means and errors for these fits. Plots are also produced for the fits (*dedx_uncorrected.ps*) and for the mean distribution as a function of $\cos\theta$ in bins of $\beta\gamma$ (*costh_uncorrected.ps*).

The method that performs the fits as a function of $\cos\theta$ takes a boolean argument that dictates whether to use the corrected or uncorrected values. A HadronSaturation object is passed by reference to obtain the parameters of the correction. If the correction is applied, the resulting plots are saved as *dedx_uncorrected.ps* and *costh_uncorrected.ps*.

2.3 HadronCalibration

In addition to the five parameters that define the hadron saturation correction, the Widget can also provide the parameters used in the $\beta\gamma$ parameterization. The fit to the $\beta\gamma$ curve, as well as the fits to σ versus $nHit$ and σ versus $\sin\theta$, is perfomed by the HadronCalibration class. This takes as input the (corrected) results of the WidgetPrep class. The output of this class are plots and a text file containing the hadron calibration parameters (including the saturation correction parameters). In the current form, this is the file that is sent to Zhu Kai at IHEP.

2.4 ElectronCalibration

This class is designed to perform the electron calibration, including run gains, wire gains, the 2D correction, the cosine correction and the 1D residual cleanup. Some of these methods already exist, others have yet to be implemented.

3 Ancillary Classes

3.1 WidgetParameterization

The WidgetParameterization class contains the hadron calibration parameterization (not including those for the hadron saturation). The parameterizations themselves are contained in smaller classes called WidgetCurve and WidgetSigma, which are called by the WidgetParameterization class to determine the predicted mean and resolution for a given $\beta\gamma$ (or dE/dx, $nHit$, $\sin\theta$).

3.2 HadronSaturation

The HadronSaturation class contains the saturation parameterization. The output of the WidgetPrep class is taken as input. The means are fitted with the hadron saturation parameterization as a function of $\cos\theta$ and the resulting parameters are stored in a HadronSaturation object. This object, containing the saturation parameters, may then be passed into the WidgetPrep object that performs the fits to the truncated means in order to apply the hadron saturation correction.

3.3 ElectronSaturation

This class is similar to the *HadronSaturation* class, except it is intended for use in the cosine correction for electrons.