README:
The Upsilon Polarization Analysis Framework

Valentin Knünz
HEPHY - Institute for High Energy Physics, Vienna

November 1, 2011

Contents

1 Overview

This README will explain how to manage the scripts for following steps:

- Preparation of the input TTrees for the polarization fit
- Running and plotting the polarization fits to real data
- Running and plotting ToyMC fits

1.1 Directory Structure

The base directory will be referred to as *basedir*. The basedir contains three folders:

- *interface* - Standard files commonVar.h and rootIncludes.inc, defining the common values of the analysis (e.g. bins)
- *latex* - All latex files for the production of summary files of the produced plots
- *macros* - This is the folder containing the relevant code. All source code files and scripts used for the Preparation of the TTrees are in this folder. All source code files and scripts used for the fits (data and ToyMC) are in the subfolder *polFit*

1.2 General Comments

The framework should work *out of the box*, no need to change any directories (other than the input data TTrees). There are five scripts, that steer everything (if no further options have to be implemented, that is all you have to look at, no need to look at the source code). The five scripts are executed by sh script.sh and are introduced here:

- *macros/PrepareTTree.sh* - Preparation of input TTrees for the data polarization fit
- *macros/polFit/runDataFits.sh* - Steering real data fits
- *macros/polFit/PlotDataFits.sh* - Steering plotting of real data fits
- *macros/polFit/runToyMC.sh* - Steering ToyMC tests
- *macros/polFit/runDataFits.sh* - Steering plotting of ToyMC tests

2 Preparation of Input TTrees for Polarization Fit

2.1 The Preparation Script

macros/PrepareTTree.sh steers all the steps discussed below. Execute it by *sh PrepareTTree.sh*. The individual steps can be activated/deactivated by setting the flags *execute_runXXX* = 1 or 0. The input Trees can be defined by *inputTreeX*. The set of cuts are defined by *FidCuts* (see macros/polFit/effsAndCuts.h for the definitions of the values), the fraction of the left mass sideband with respect to the right sideband for the evaluation of the background angular distribution is defined by *FracLSB*. The mass region about the pole mass in which the events are projected are defined by *nSigma*. (Multiple values can be chosen, then the script loops over the values)

Output Structure:

$macros/DataFiles/SetOfCuts\{FidCuts\}/AllStates\_\{nSigma\}Sigma\_FracLSB\{FracLSB\}Percent/$

A file for each of the three states and each kinematic bin is saved in this directory (and for each cut, nSigma and FracLSB in a separate directory).

$macros/DataFiles/SetOfCuts\{FidCuts\}/PDF$: Here, the mass plot pdf and background $\cos\vartheta/\varphi$ plot pdf are saved.

2.2 The Individual Steps (Details)

The preparation of the input TTrees for the polarization fit is subdivided in several macros, written by Hermine. The structure of the macros was altered, to allow for easy scripting. All steps are steered by one script (**macros/PrepareTTree.sh**). The individual necessary steps are (from Hermine's README.txt):

1.) **runData.cc** is the steering macro for PolData.C/h which loops over the TTree produced from the JPsiAnalyzerPAT.cc macro, applies a series of muon quality and kinematical cuts, and stores the TLorentzVector of the two muons of the selected events, as well as a series of mass histograms in an output file. The default name of the root output file is:

$macros/DataFiles/SetOfCuts\{FidCuts\}/tmpFiles/selEvents\_data\_Ups.root$

2.) **runMassFit..cc** is the steering macro for upsilon_2StepFit.C which loops over all pT / $|y|$ bins and performs a fit to the invariant mass region in the upsilon mass region. The fit is a 2-step fit in which the BG shape and normalization is fixed from the L and R mass sideband windows which is then imposed in the fit to the signal region, defined in between the L and R sidebands. The signal consists of 3 Crystal ball functions with common tail parameters (alpha and n) where only the mass and width of the 1S are left free and the corresponding mass and widths of the 2S and 3S are fixed by the respective mass ratios as given in the PDG2010 tables.

The CB parameters (alpha and n) are fixed from the pT integrated bin, for each rapidity separately, and are imposed on the pT differential bins.

The fitted TF1 objects for the signal (3 CB functions) as well as the BG function are stored in output files called

$macros/DataFiles/SetOfCuts\{FidCuts\}/tmpFiles/data\_Ups\_rap1\_pT1.root$

This macro also saves the graphical representation of the fit in corresponding pdf files.

In order to visualize the series of pdf files created in the previous step latex/massFits.tex produces a summary pdf file.

3.) **runCopyTreeEntries.cc** is the steering macro for CopyTreeEntries.C. This macro loops again over the same pT and y bins and adds to the previously created output file (data_Ups_rap1_pT1.root) the TLorentzVectors of the 2 leptons, specific to that pT and y bin. It furthermore, calculates the width of the L and R sideband windows and projects corresponding events into the L and R (cosTheta,phi) 2D histos.

The macro **PlotCosThetaPhiBG.cc** plots the individual (cosTheta, phi) distributions of the L and R mass sidebands. The individual figures can then be assembled into one pdf file using the file latex/cosThetaPhi_BG.tex

4.) **runTrimEventContent.cc** is the steering macro for TrimEventContent.C. This macro loops over the events in a given pT and y bin and stores in the "data" TTree only those events which are selected within a certain nSigma window around the signal pole mass. It also adds the Left and Right Sideband mass windows in a given fraction and stores one output BG histogram, as a function of (cosTheta, phi). Furthermore, it also projects the (cosTheta, phi) distribution in a 2D histogram of the signal events; the fraction of BG in the nSigma window around the signal is stored in a 1D histogram. Inputs to the macro are the
*) fraction with which the L BG should be added to the right one
*) nSigma within which the events should be projected
*) which state is being considered: //[0]... 1S, [1]... 2S, [2]... 3S The final input files for the polarization fit are saved to

$macros/DataFiles/SetOfCuts\{FidCuts\}/$
$AllStates\_\{nSigma\}Sigma\_FracLSB\{FracLSB\}Percent/data\_\{nState\}SUps\_rap1\_pT1.root$

5.) **runMeanPt.cc** again loops over the files previously produced, calculating the mean pT, number of signal events and background fraction. It produces a text file containing arrays of these quantities, which are used for plotting and as input for the ToyMC studies (one needs to copy these arrays to ToyMC.h, if one wants to change the settings of the toys). The text file can be found on

$macros/DataFiles/SetOfCuts\{FidCuts\}/$
$AllStates\_\{nSigma\}Sigma\_FracLSB\{FracLSB\}Percent/meanPt.txt$

3 Data Fits

3.1 The runDataFits.sh Script

This script steers the polarization fits to real data, as prepared by the previous step. It is important that the same fiducial cuts are applied in the fitting code, as in the preparation code. This is ensured, if the cuts in macros/polFit/effsAndCuts.h are not changed inbetween preparation and fitting. Following settings can be altered:
*) fracL (in percent, needed to find the correct data set)
*) nSigma (needed in 2 decimal accuracy (x.yz), needed to find the correct data set)
*) nState (which state to fit, all three can be fit in a loop)
*) JobID (The results and plots are saved in macros/polFit/{JobID}, Please define nSigma and fracL yourself in the JobID, if needed)
*) rapBinMin, rapBinMax, ptBinMin, ptBinMax (which bins to fit)
*) nEff (which efficiency definion to use for the single lepton efficiencies, see macros/polFit/effsAndCuts.h for the definitions. If binned efficiencies are used, the corresponding root files must be copied to macros/polFit/EffFiles/, the file names have to be updated in macros/polFit/effsAndCuts.h. nEff $>$ 100 corresponds to binned efficiency definitions (These considerations are valid for all variables of kind *nEff*, also in the ToyMC script.)
*) nDileptonEff (which efficiency definion to use, for efficiencies depending on dimuon variables p_T and y. See macros/polFit/effsAndCuts.h for the definitions. nEff $>$ 200 corresponds to binned efficiency definitions)
*) FidCuts (needed to find the correct data set and used to define the cuts applied in polFit.C)
*) nSample (defines the number of iterations in polFit.C. Testing: 6000 is enough, for the final results, 100000 should be used)
*) datadir can be defined different optionally. As it is now, it finds the correct dataset via the previously mentioned values.
*) nFits defines the number of fits (as background subtraction is based on random process, the fit should be repeated nFits times). The result root files of the individual fits are merged, the merged file is used to extract the mean and error from the parameter probability distributions.
*) nSkipGen can be used, if after nFits fits additional fits are to be made. Then nSkipGen can be set to nFits (from before). This can not be done in parallel!
*) storagedir, should be set to a directory, where no issues concerning storage is expected. N fits of each bin can result in a large amount of data.

3.2 The PlotDataFits.sh Script

This script plots the results of a certain fit of state {nState} with JobID {JobID}. The summary pdf (plots and numerical values) files are then saved in macros/polFit/FiguresData/{JobID}. Define following variables:
*) JobID (which JobID's to plot, can be several in a loop)
*) nState (which state to plot, all three can be plotted in a loop)
*) ptBinMin, ptBinMax (which bins to plot)
*) storagedir, has to be set to the storagedir defined in runDataFits.sh

4 ToyMC Fits

4.1 The runToyMC.sh Script

This script steers all generation, reconstruction fitting and the storage of the result files. This script can be run in parallel for any scenario, bins, efficiencies, fiducial cuts. The results are stored in storagedir/JobID/... Following variables need to be defined:
gen..... boolean, if true, polGen.C is executed
rec..... boolean, if true, polRec.C is executed
fit..... boolean, if true, polFit.C is executed
plot..... boolean, if true, polPlot.C is executed
JobID..... Name to be specified for a certain test
rapBinMin..... test will be conducted from this rapidity bin...
rapBinMax..... ...to this rapidity bin
ptBinMin..... test will be conducted from this pT bin...
ptBinMax.....to this p_T bin
polScenSig..... polarization scenario Signal (see ToyMC.h)
polScenBkg..... polarization scenario Background (see ToyMC.h)
frameSig..... natural polarization frame signal (1...CS, 2...HX, 3...PX)
frameBkg..... natural polarization frame background (1...CS, 2...HX, 3...PX)
nGenerations..... number of pseudo samples to be generated and fit
nEff..... defines the efficiency to be used for the fitting (see macros/polFit/effsAndCuts.h)
nDileptonEff..... defines the additional dimuon efficiency to be used for the fitting (see macros/polFit/effsAndCuts.h)
FidCuts..... defines, which set of cuts will be used (see macros/polFit/effsAndCuts.h)
nSample..... numer of iterations in the algorithm (see polFit.C, 2000 burn-in iterations are included in this number)
ConstEvents..... integer, see below
UseConstEv..... boolean, if true it generates ConstEvents events, if false, it uses the number of events stored in ToyMC.h
storagedir..... can be set optionally (if there is not enough storage space in the code directory) to the directory where all datasets, results and figures will be stored
nEffRec..... defines the efficiency to be used for the generation of the pseudo set (see macros/polFit/effsAndCuts.h), which is irrelevant if UseDifferingEff is set false
nDileptonEffRec..... defines the dimuon efficiency to be used for the generation of the pseudo set (see macros/polFit/effsAndCuts.h), which is irrelevant if UseDifferingEff is set false
UseDifferingEff.... boolean, if true it generates the pseudo sample according to the efficiency definition nEffRec, if false, it uses the same efficiency for the generation as well as for the fitting macro, both with nEff.

4.2 The PlotToyMC.sh Script

When all fits, bins, scenarios are finished, or if one wants to get a snapshot of the results in between, run this script. It plots the results and further produces two summary pdfs, one pdf for each bin showing all parameter and pull distributions and one pdf containing numerical results, saved in the directory macros/polFit/FiguresToyMC/{JobID}
storagedir..... same as above
JobID..... same as above
ptBinMin..... can be same as above, this just defines the minpT of the plots
ptBinMax..... can be same as above, this just defines the maxpT of the plots
frameSig..... same as above
polScenSig..... same as above
frameBkg..... same as above
polScenBkg..... same as above
nGenerations..... same as above

additionalName..... can be set optionally, if one wants to add a certain extension to the pdf summary files

4.3 The Source Code

macros/polFit/ToyMC.h This file contains all relevant information that is needed for the tests. Polarization scenario definitions, Number of events estimated from data, background fractions estimated from data, Color and Marker settings.

macros/polFit/polGen.C Macro needed to generate pseudo sample
Output:
-genData.root, containing the pseudo sample on generator level
-GenResults.root, containing information about the actual injected polarization in all frames, needed for plotting the results

macros/polFit/polRec.C Macro needed to alter pseudo sample according to defined efficiencies and fiducial cuts
Input:
-genData.root
Output:
-data.root, containing the eventually used sample
-efficiency.root, containing 2D efficiency histograms (muon pT vs. $|\eta|$)

macros/polFit/polFit.C This macro conducts the fit
Input:
-data.root
Output:
-results.root, containing the posterior distributions of all parameters, and a TTree containing the numerical results (mean and rms of the posterior distributions)

macros/polFit/polPlot.C This macro can plot the individually generated data sets vs. the fit result.
Input:
-results.root
Output:
-several plots

macros/polFit/polRapPtPlot.cc This is the source file for the executable that plots the real data fit results and ToyMC results and produces latex tables.

macros/polFit/polGenRecFitPlot.cc This is the source file for the executable that generates, reconstructs and fits the data sets (real data and Toys)

macros/polFit/Makefile This file steers the compilation of polRapPtPlot.cc and polGenRecFitPlot.cc.