README:
The Upsilon Polarization Analysis Framework

Valentin Knünz
HEPHY - Institute for High Energy Physics, Vienna

January 29, 2013

Contents

1 Overview

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

- Preparation of the input files for the polarization fit
- Running and plotting the polarization fits of 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*

The storage directory will be referred to as *storagedir*. **Important:** The storagedir has to be defined by creating a file basedir/macros/polFit/storagedir, containing the path, where you wish to save all the result files (which can be large).

1.2 General Comments

The framework should work *out of the box*, no need to change any directories (other than the input data TTrees and defining the storagedir). There are seven main 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 files for the data polarization fit
- *macros/polFit/runDataFits.sh* - Steering real data fits (and external MC, background polarization)
- *macros/polFit/PlotDataFits.sh* - Steering extraction of results from PPDs of real data fits
- *macros/polFit/runToyMC.sh* - Steering ToyMC tests
- *macros/polFit/PlotToyMC.sh* - Steering extraction of results from PPDs of ToyMC tests
- *macros/polFit/PlotResults.sh* - Steering the plotting of all simple λ plots
- *macros/polFit/PlotCentrals.sh* - Steering the plotting of the multi-panel λ plots and the production of the SupplementalMaterial.txt file

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). You have to define a JobID, which identifies the specific dataset.

Output Structure:
macros/DataFiles/SetOfCuts{FidCuts}_JobID/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). Further, in this directory some tex files are saved, containing tables of the estimated number of signal events, background fraction, mean p_T, ...

macros/DataFiles/SetOfCuts{FidCuts}_JobID/PDF: Here, the mass plot pdf and background cosϑ/φ plot pdf are saved.

Some additional interesting plots (dataset characteristics, performance plots, pedagogical plots) are saved to *macros/DataFiles/SetOfCuts{FidCuts}_JobID/Figures*.

2.2 The Individual Steps (Details)

The preparation of the input files 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:

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, trigger matching, 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}_JobID/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}_JobID/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

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

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

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

2.3 Additional features

With PrepareTTree.sh, other features can be used, differing from the “default” preparation of the data set as described above:

1.) **Preparing data for external MC tests:** For this feature, set *UpsMC=1* and *UpsMCstate =n* for a Υ(nS) MC sample. The code will then prepare the inputs for the polarization fits as follows:

- The background fraction will be set to 0.001 - so effectively no event will be subtracted in the polarization fit.
- The background histograms are filled with the distributions of the MC sample itself, to avoid any problems in the polarization code.
- Please note that the mass window of the external MC preparation is hard-coded in TrimEventContent.C, as we want to have numerically the exact same window as in data.

2.) **Preparing data for measurement of the background polariztion:** For this feature, set *ProjectLSBdata =1* or *ProjectRSBdata =1* or *CombineSignalPeaks =1* . In the first case, the data of the LSB will be projected, in the second case the data in the RSB, in the third case, the data in the full signal region (combining the Υ(nS) mass windows). The third option only makes sense, if you change the lifetime cuts in interface/commonVar.h, to be able to measure the background enriched high lifetime region. The code will then prepare the inputs for the polarization fits as follows:

- The background fraction will be set to 0.001 - so effectively no event will be subtracted in the polarization fit.
- The background histograms are filled with the distributions of the background sample itself, to avoid any problems in the polarization code.

3.) **Additional settings:**

- *RequestTrigger:* If set to 1, trigger-matching is active (as defined in mactos/PolData.C)
- *DoCPUconsumingPlots:* If set to 0, you can avoid the production of CPU consuming plots (for debugging purposes)

- *adjustOverlapBorders:* If set to 1, the code will avoid that the Υ(nS) mass windows overlap, and therfore avoids the doubled use of events. This is only important, if nSigma is chosen to be large.

3 Data fits

3.1 The runDataFits.sh Script

This script steers the polarization fits to real data (and external MC), 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 in between preparation and fitting. The result files containing the information of the PPD are saved in the directory storagedir/Data/JobID. Additionally, output Figures for each state and bin are saved to storagedir/Data/JobID/Figures (projections of the PPD, background subtraction performance plots, plots of the distributions of cosϑ, φ and $\tilde{\varphi}$, ...)

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............... defines the efficiency to be used for the fitting (see macros/polFit/effsAndCuts.h)
*) UseMCeff...............boolean, if true, the MC efficiency in the file nEff is used
*) nDileptonEff............... defines the online dilepton efficiency to be used for the fitting (see macros/polFit/effsAndCuts.h)
*) UseMCDileptoneff.... boolean, if true, the MC efficiency in the file nDileptonEff is used
*) nRhoFactor............... defines the ρ maps to be used for the fitting (see macros/polFit/effsAndCuts.h)
*) 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: 20000 is enough, for the final results, 100000 should be used, 10000 burn-in iterations are included in this number)
*) DataID (This should be identical to _JobID with JobID the one from PrepareTTree.sh)
*) 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. nFit=50 is the default for the central data results. nFit=10 is good enough for systematic tests, nFit=1 is enough if you work with external MC.
*) 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.

3.2 The PlotDataFits.sh Script

This script analyzes the results of a certain fit of state {nState} with JobID {JobID}. From the 1D PPD of the λ parameters, the most probable value (MPV) and the statistical uncertainties (positive and negative) corresponding to the nSigma confidence level are evaluated, and saved as function of pT and rapidity in TGraph objects in the directory storagedir/Data/JobID (The TGraphs are the input for the plotting step, as described below). Further, 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)
*) nSigma (defines the CL of the uncertainties) - this can be run over a loop (e.g. "for nSigma in 3 2 1;do")

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/ToyMC/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 pT 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
FidCuts............ defines, which set of cuts will be used (see macros/polFit/effsAndCuts.h)
nSample............ number 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
nEff............... defines the efficiency to be used for the fitting (see macros/polFit/effsAndCuts.h)
UseMCeff..............boolean, if true, the MC efficiency in the file nEff is used
nDileptonEff............... defines the online dilepton efficiency to be used for the fitting (see macros/polFit/effsAndCuts.h)
UseMCDileptoneff.... boolean, if true, the MC efficiency in the file nDileptonEff is used
nRhoFactor............... defines the ρ maps to be used for the fitting (see macros/polFit/effsAndCuts.h)
UseDifferingEff.... boolean, if true it generates/reconstructs the pseudo sample according to the efficiency definitions as in nEffRec, UseMCReceff, nDileptonEffRec, UseMCDileptonReceff, nRecRhoFactor (same meanings and notations as above). If false, it uses the same efficiency for the generation/reconstruction as well as for the fitting macro, therefore ignoring the five variables mentioned in this point.

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 evaluates the median of the nGenerations toyMC-results and saves a TGraph object in storagedir/ToyMC/JobID (The TGraphs are the input for the plotting step, as described below).
This script 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}
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

5 Plotting

The plotting of the data, toyMC, systematics or any other results (λ vs p_T) is done with the source file PlotFinalResults.cc, which is steered by two scripts:

- *macros/polFit/PlotResults.sh* - Steering the plotting of all simple λ plots
- *macros/polFit/PlotCentrals.sh* - Steering the plotting of the multi-panel λ plots and the production of the SupplementalMaterial.txt file

PlotResults.sh

There are two different kinds of plots, that you can switch on and off with 0 and 1:

- PlotFinalData=1
- PlotSystematics=0

Both kinds of plots are saved in the directories macros/polFit/FinalResults/JobID/nSUps/Figures_ additionalName (you have to define *additionalName* and *JobID* in the options). The corresponding summary pdf files are saved to macros/polFit/FinalResults/JobID/nSUps, containing the *additionalName* in the filename.

The **PlotFinalData** plots take the input files from the storagedir/Data subfolders and are specified with *DefaultID=XXX*.

The produced plots show the central values and uncertainties. If you want to compare TGraphs from data results in this way, use *PlotCompare=1*. Specify how many graphs you want to compare with *nComp=XXX* (default + nComp) and specify which subfolders of storagedir/Data to compare to the default with (*CompareID1=XXX, CompareID2=XXX, ...*).

With this, you can specify the legend entries of the individual graphs: *LegendEntryDefID=XXX, LegendEntryCompID1=XXX, LegendEntryCompID2=XXX,...* and decide if you want to plot the legend at all with *PlotLegend=1*.

In these PlotFinalData plots, the individual systematics (explained below) will be added in quadrature and be added as uncertainty to the plot. If you do not want that, you must specify *nSystematics=0*.

Now, concerning the **"systematics" plots** (*PlotSystematics=1*): There are two different kinds of systematic plots. The systematic overview plots (e.g. AN-2012-140, Fig 35) and Figures like AN-2012-140, Fig 24, showing the systematics as lines connecting the points of the TGraphs. Fot the latter plot, use *PlotAsymm=1*, for the overview type use *PlotAsymm=0*

Then, it will use a number of nSystematics that you specified with *nSystematics=3* (as example). From the Systematics folder polFit/Systematics/SystIDXBase/SystIDXSpecify (systematics are evaluated and saved with the help of additional scripts, see Section 7), it will take TGraphs from this folder and plot the graphs. I have commented (in the PlotResults.sh file in CVS) all the settings for all λ vs. p_T plots of the AN-2012-140. For example, if you want to plot the "FrameworkI" systematics, you can use these settings:

```
SystID1Base=TheGreatRun_FrameworkI
SystID1Specify=FrameworkI_3S
SystID1Title=Upsilon3S

SystID2Base=TheGreatRun_FrameworkI
SystID2Specify=FrameworkI_2S
SystID2Title=Upsilon2S

SystID3Base=TheGreatRun_FrameworkI
SystID3Specify=FrameworkI_1S
SystID3Title=Upsilon1S
```

PlotFinalResults.cc then takes the corresponding three TGraphs and plots it with legend entries *SystIDXTitle*. So in order to make these systematics plots, in basedir/Systematics/SystIDXBase/SystIDXSpecify these TGraphs have to exist. Either you copy the TGraphs from the storagedir/XXX (in case of systematics evaluated with toyMC, where the toyMC-bias is equivalent to one of the systematics) directory, or use one of the scripts listed in Section 7 (e.g. difference of two data results). The plotting macro only compares TGraphs from one state (e.g. 1SUps). So, if you want to compare 1S and 2S toys, you should rename the 2S-TGraph to look like a 1S TGraph make a new SystIDXSpecify.

There are a few other bools that are not important (If you need more information, ask for help)

PlotCentrals.sh

PlotCentrals.sh works essentially the same as PlotResults.sh - The only difference is that the settings are fixed to those needed to produce the multi-panel λ plots and the supplemental material file basedir/macros/polFit/FinalResults/JobID/SupplementalMaterial.txt.

The specific settings that are needed to produce the publication-quality multi-panel plots are:
PlotBrazilian=1
DrawLatexStuff=1
DrawPreliminary=0 (unless the plots are preliminary, e.g. for the approval)
MultiPanelPlots=1
MPCentralsWithTotalSystID=XXX
PlotAlteredPPDResults=1
This script needs as input a DefaultID and a MPCentralsWithTotalSystID. The folder storagedir/Data/DefaultID must contain the TGraphs coresponding to the 1, 2 and 3 sigma total uncertainty (as obtained after including the systematics in the PPD, see Section 7). The folder basedir/macros/polFit/Systematics/TotalSyst/ MPCentralsWithTotalSystID must contain the central results, with the 1 sigma statistical uncertainty.

6 The “polFit” 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 the evaluation of the toyMC-result and toyMC-bias

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 evaluates the real data fit results and ToyMC results from the individual PPDs, 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/PlotFinalResults.cc This file plots the TGraphs as produced by polRapPtPlot.cc

macros/polFit/Makefile This file steers the compilation of polRapPtPlot.cc, PlotFinalResults.cc and polGenRecFitPlot.cc (among others).

7 Additional useful scripts and macros

This section introduces additional useful scripts and macros that are available in the framework, mostly to handle the systematics. These scripts and macros are all located in basedir/macros/polFit. Only basic information about the purpose of the scripts is provided:

- **AlterPPD.sh:** Adds a smearing to the PPD, corresponding to the systematics and the assumed probability distribution, that you specify
- **AverageSyst.sh:** Adds a number of systematics quadratically, or calculates the mean of a number of systematics (changes in the source code needed)
- **ChangeTGraph.sh:** Changes a TGraph very flexibly - you have to change the source code according to your needs
- **EvalSyst.sh:** Calculates the difference between two TGraphs, which is useful to define systematics
- **MergeDataFiles.sh:** Merges the result files of the nFit data fits
- **BinExamplePlots.C:** Produces some important plots corresponding to the results of data fits with specified JobID (background subtraction control plots, nFit individual results scatterplot, Metropolis-Hastigs-efficiency, ...)
- **ScaleMCtruth.C:** Marco used to parametrize the T&P efficiencies
- **Chi2FitBGratio.C:** See code for more information
- **FitBGratio.C:** See code for more information
- **FitRho.C:** See code for more information

8 Pitfalls

- If you use a new efficiency file, make sure that it is read correctly. The formats of the efficiency files are changed many times. To cope with that, the code was made very flexible, but this can lead to problems, in case the format changes again.

- If you use a ρ correction, make sure that the binning in pT and rapidity of the maps is correct. This is hardcoded in effsAndCuts.h, as there is no information about it in the file.