LHCb StarterKit 2024: Practice session *

Analysing your data

StarterKit Organizers

November 2024

1 Goals of the session*

We will now put to use some of the knowledge learned in this workshop. You (should) have the data you're interested in, in some ROOT ntuples. What can you do with it?

We will go through some basic steps that you would take to study a Monte Carlo sample of your signal channel of interest. This includes applying some rudimentary cuts to it, and fitting the resulting mass distribution. Something

2 Setup of the Session

You should have worked this morning on producing a ntuple for the decays $D^+_{(s)} \to \phi(\to \mu^+\mu^-)\pi^+$, with a simple DaVinci script. As running DaVinci can take a long time to get a reasonable amount of signal events in your ntuple, for this second part of the Session you are provided with a ROOT file that contains the TTree with the kinematic, PID, and MC-truth variables, but with much higher statistics. The ntuple you need can be found at the following path:

`/afs/cern.ch/work/j/jcambonb/public/ntuples_files/Ds2mumupi_MC_MagDown_2024_tuple`

We can now do some fits to the data in our ROOT file. To do that, you have many options at your disposal. You can use `RooFit` or `minuit` like we saw in the statistics lesson. You can also use `zfit`, in which case you will need to set up an environment. For that you can use the 'lb-conda' one as we saw in the python lessons

You can write a python script or a jupyter notebook if you want to add comments and explanation for the steps that you'll take throughout the session. We will not mark you, so this is just for your own future reference!

Use the lessons and materials provided during the week as you go along! This Session is not meant to be test of your knowledge, but just to make you more familiar with the tools we've seen. Helpers will be around to assist with issues you may encounter.

3 Exploring the data

First, you can use `uproot` to read and load in the data contained in the ntuple. Sort through the ROOT file to find the location of your TTree. Once you're able to load it, you can start by loading in the $D^+_{(s)}$ mass branch, which is called `D_M`. Try and plot it and see what it looks like! You should see a peak over a continuous background.

*Found a mistake in this document? Please contact lhcb-starterkit-organisers@cern.ch

Later, you can start loading in more branches. Here's a list of the ones that will be useful for today:

`D_M`, `D_TRUEID`, `mup_PID_MU`, `mum_PID_MU`, `mup_TRUEID`, `mum_TRUEID`, `pi_PID_K`, `pi_PID_MU`, `pi_TRUEID`

You can have a look at them by plotting them and seeing what their distributions look like. We can note a few things about these branches:

- The ntuple contains both $D^+_{(s)} \to \phi(\to \mu^+\mu^-)\pi^+$ and the charge-conjugate process $D^-_{(s)} \to \phi(\to \mu^-\mu^+)\pi^-$. The names of the branches of the children $\mu^\pm$ and π^+ are only meant to reflect what the process is, but it doesn't mean that those branches *only* contains muons and pions of a given charge.
- The `TRUEID` variables return an identification number assigned at MC level to any particle candidate in the ntuple. This is useful because simulated data that passes through the nominal LHCb reconstruction may produce candidates which do not correspond to the original signal generated by Gauss (as reconstruction is not 100% efficient). The `TRUEID` for D_s^+, ϕ, $\mu^\pm$ and π^+ are 431, 333, $\pm$13, 211, respectively. More information on the conventions for this numbering scheme can be found here.
- The `PID` variables can be interpreted as a likelihood that a particle candidate is of a given species. For example, `pip_PIDmu` returns the likelihood that the π^+ candidate is a muon; `mum_PIDmu` returns the likelihood that the μ^+ candidate is actually a muon; and so on. Note that the number returned is not a probability, i.e a number from 0 to 1, but as a rule of thumb the chance of a particle being of a given species is higher the higher the value of the corresponding `PID`.

We can use these to apply cuts on our data! For example, when using simulation, a common selection is what is known as *truth-matching*. This means requiring that the `TRUEID` of the candidates match what we expect. Because we have both D_s^+ and D_s^- decay candidates, we have to require that the *absolute value* of the `TRUEID` matches the known values.

Apply the truth-matching cut on your data and look at how the shape of the distribution changes. You should see that most of the background gets removed, and only the peak remains.

Alternatively, you can play with the `PID` variables and apply different cuts to see how the distribution is affected.

3.1 Fitting your data

We're ready now to start doing some fits! We want to fit the mass distribution of the D_s^+ candidates. First, start by setting up your observable which will be the proxy for the variable `D_M`.

As you have seen by plotting the data, we can start by modelling the peak as a simple gaussian. Set up the parameters for the mean and width, as well as a yield parameter. Try doing an extended unbinned fit with the gaussian to the non-truth matched data, and print out the result. Is it a good fit?

Next, you can plot the data and the fitted model together to see how well the gaussian models the data. You can also try to plot the 1D likelihood function for the mean of the gaussian.

Advanced: you can also try to plot what are known as *pulls*. You can think of pulls as a very naive measure of the goodness-of-fit. If you plot your data as a histogram, then the pull relative to one of its bins is defined as such:

$$pull = \frac{N_{bin} - y_{bin}}{\sqrt{N_{bin}}} \quad y_{bin} = \int_{bin} pdf(m)dm \tag{1}$$

This is the difference between the number of entries in a given bin N_{bin} with the integral of the extended pdf over that bin, normalised by the poissonian error $\sqrt{N_{bin}}$. Generally, in a good fit, the distribution of

pulls for all bin should be centered in zero and be symmetric. They should be randomly scattered and no systematic trend should be visible.

3.2 Improving the model

Since we're not modelling the background at the moment, we shouldn't expect to see a good fit when plotting it against our data. So, we can add to our model a second pdf, which represents the background. This can be modelled by an exponential function. Remember to set up the yield for this as well and sum the resulting extended pdf with the one that we made for the gaussian peak.

Try to fit this updated model to the data, and plot it. Does it look better?

Advanced: Plot the pulls for this as well.

4 Advanced: MVA trainining

This year we had a specific class regarding the use of machine learning algorithms for signal to background optimization at high energy physics analysis. Since the $D_s^+ \to \phi(\to \mu^+\mu^-)\pi^+$ are quite rare, we expect that the data will be dominated by background. So, the use of MVA algorithms can be really useful. For that, we have produced also a data ntuple with some events where we can play a bit :). It can be found here:

`/afs/cern.ch/work/j/jcambonb/public/ntuples_files/D2pimumu_bkg_sample_MagDown_2024_tuple.root`

- Prepare the training samples. For that, define a sample of signal using your simulation and a sample of background using simulation or data. Data ntuple we share with you is a pure background distribution, so you can use it as background sample. Compare the features for both samples and choose which ones are good at discriminating them.
- Define your MVA algorithm following. *Hint*: You can check the class regarding BDTs and choose one of its python implementations
- Do a train and test split of your signal and background samples. Train your algorithm with the training sample and test it with the test one. Get the ROC curve
- **Really advanced**: Check the analysis essentials page `https://hsf-training.github.io/analysis-essentials/advanced-python/30Classification.html` and see how can the training can be improved. Try the implementations of k-folding for dealing with over-training and also check how to tune the hyperparameters of your model