Topical Lectures Statistics – Exercises Set 3

Wouter Verkerke (Dec 2020)

General Instructions

Input files

Input files for exercises can be found in three places

1. At nikhef (stbc-i5.nikhef.nl) in directory `~verkerke/stats2020/`
2. At CERN (lxplus7.cern.ch) in directory `~verkerke/public/stats2020`
3. On the web at `http://www.nikhef.nl/~verkerke/stats2020`

Running ROOT

All exercises are based on ROOT. You are recommended to use version 6.14.04, in which all prepared material has been tested. To pick up a pre-installed ROOT version please execute the following setup script

```
source /cvmfs/sft.cern.ch/lcg/app/releases/ROOT/6.14.04/x86_64-centos7-gcc48-opt/root/bin/thisroot.sh
```

This release will work both at Nikhef and at CERN

Where to work

If you have an account at Nikhef, please work on stbc-i1.nikhef.nl or stbc-i2.nikhef.nl only (these run CentOS7 – required for above ROOT version)

If you have an account at CERN, please work on lxplus7.cern.ch, this will select an CentOS7 node (required for above ROOT version)

You can also work directly on your laptop if you have ROOT installed yourself

Exercise 14 – Template sum models

In this exercise we will explore binned shape models that use simulation histograms as shape functions, these are then added together to form a model that predicts a variable amount of signal on top of background

Copy file `ex14_build_binned.C`. This macro performs the following steps

- Generate template histograms for signal and backgrounds on the fly, so that this exercise does not need an input file (Normally these histograms would be produced from full simulation and would be read in from a file)
- Constructs an 'amplitude sum' model that adds histograms together into a probability density model, where the signal strength is multiplied with a scale factor μ.
- Generates a binned dataset from the template model with μ=1.5
- Fits template model to the binned dataset (thus performing a binned likelihood fit)

The main difference between a sum of pdfs (RooFit operator `SUM`) and a sum of amplitudes (RooFit operator `ASUM`) is that the latter retains the information on the event count in the histogram. (In other words in `ASUM` the normalization to a unit probability happens *after* the summation, whereas in `SUM` it happens to each pdf *before* the summation).

It is also important to note that the event count that is associated with a histogram is equal to the *integral* over that histogram. If the bin width is not 1, then this integrals is *not equal* to the number of entries that was used to fill the histogram, and we should correct interpretation of the yield parameters with the bin width.

Furthermore, if the simulation templates correspond to a luminosity that is different from that of the data, an additional correction factor is needed for that in the interpretation of yield parameters.

- Study the macro, and try to find the places where correction factors for the bin with (variable `binw`) and the luminosity ration (variable `L`) are introduced.
- Run the macro through your favorite RooStats limit calculators

In the initial configuration the Luminosity of the 'simulation data' is identical to that of the data.

- Change the macro so that the luminosity of the simulation is increased by a factory 10 (look for the lines '`wsim.pdf()->generateBinned()`'.
- Run the fit again and observe that the fitted signal strength μ now is reported a factor 10 too low. Fix this by adjusting the luminosity ratio parameter `L` in the fit model.

Exercise 15 – Template morphing models

In this macro we explore the concept of template morphing to take a set of three simulation signal histograms, sampled at masses of 123, 125 and 127 GeV, and construct a morphing interpolation model out of these templates that predicts the signal distribution for any mass, as function of a newly introduced morphing parameter α.

Copy `ex15_build_binned_morphing.C`. This macro performs the following steps:

- Generate template histograms for signal (at 3 mass points) and background
- Construct a morphing interpolation signal model from the three mass templates
- Construct a signal+background model
- Multiply the summed model with a subsidiary measurement for the morphing parameter alpha (which is a unit Gaussian)

After the morphing signal model construction, the macro largely follows the code of `ex14_build_binned.C.`

Questions & explorations

- Run the macro and study its output. Is the NP alpha that expresses the uncertainty on the signal mass overconstrained?
- How does the fit behave if you (strongly) increase or decrease the template statistics?
- Study the effect of turning off the interpolation feature that zeros event yields that come out negative in the extrapolation. You can best see this if you create a 2D ROOT histogram sampling the morphing model sig in `mgg` vs `alpha` (`sig->createHistogram("mgg,alpha")`)

Optional extra steps

- Modify the macro so that the signal width is a free parameter.
- Modify the macro to have both width and mean as free parameters.
  Note that with multiple interpolation directions and parameters the syntax of the interpolation class becomes

```
PiecewiseInterpolation::sig(nom,{loA,loB},{hiA,hiB},{alphaA,alphaB}) ;
```

Exercise 16 – Building a combined workspace

In this exercise we explore how to build likelihood-level ('perfect') combination out of previously performed measurements for which the likelihood was saved in a RooFit workspace.

The inputs we will combine for the demonstration combination is the Poisson counting experiment of exercise 12 and the template morphing model of the exercise 15. To prepare for this exercise you must first do the following

- Run `ex12_build_PoissonGaussGlobs.C` and rename its workspace output file `model.root` to `model_ex12.root`
- Run `ex15_build_binned_morphing.C` and rename its workspace output file `model.root` to `model_ex15.root`

Open each of the renamed output files in a clean ROOT session, and inspect the workspace contents ('`w->Print("t")`') . The "`t`" option (for 'tree') will organize components of each likelihood expression in the form of a dependency tree so it is easier to understand their structure.

Now copy `ex16_combined.C` and inspect it's contents. This macro performs the following steps

- It constructs a joint likelihood of two previously constructed models by reading their workspace
- To be able to join workspaces and datasets, a discrete observable (like a C++ enum) must be introduced to label the parts of the pdf and the dataset that belong to each channel.
- After a fit to the joint model, the joint data and models are written to a new joint workspace file

Questions & explorations

- The default strategy of the workspace commands (as specified in the macro) is to rename all component functions of in an imported pdf with a suffix so that there are no naming collisions when multiple functions are imported. By default parameters are not renamed, so that any parameter that occurs in more than one model becomes a common parameter of those models.

  In the macro this default behavior is explicitly overridden, initially. The signal strength parameters of each likelihood are also renamed so that the combined model has two strength parameters `mu_ex12` and `mu_ex15`. Since the models of ex12 and ex15 share no other parameters the likelihood are completely orthogonal, so a minimization of the combined likelihood should give the same results as the minimization of the individual likelihoods. Run both the joint fit and the individual fits (macros bellowing to ex12 and ex15) and confirm that this is the case (within numeric precision, which is about 4% of the fitted uncertainty of each parameter)

- Modify the macro so that the mu variables of the input workspaces are joined as a common parameter. To this simple remove the "`RenameVariable()`" command options from the import commands. Run the combined fit and observe how the fitted μ is now the weighted average (to good approximation) of the μ values of the individual fits.

- Now copy input file `model_ex12_highstats.root` to `model_ex12.root`. This file is identical to the standard output of ex12 except that signal and background counts have been increased by a factor 10 so that the uncertainty on μ from this standalone model is now comparable to that of ex15. Run the combination again and try to interpret the combined μ value.

- *Write a ModelConfig file for the joint model and store it in the workspace, then run the RooStats PLR calculator on the joint model with a single* μ *parameter*

Exercise 17 – Fourier Convolution models

In ths final macro we return to unbinned likelihood models and explore convolution models based on discrete FFT transformation.

Copy file `ex16_convolution.C`. This macro performs the following steps

- Construct a Landau physics model and a Gaussian resolution model, then construct a FFT-based convolution of these two
- Generate 1000 events and fit the model to this data
- Plot the fitted convolution model on the data, and also make separate plots of the fitted shapes of the physics truth model (the Landau) and the resolution model

Questions and explorations

- Run the macro. Change number of events to different counts (e.g. 100k events) and observe how little CPU time is needed to perform the convolution calculations.

- Change samples size back to 1000 and hen change the value of the mean of the Landau from 30 to 80. Rerun the macro. Do you see any signs of cyclical spillover?

- Now uncomment the line that sets the overflow buffer fraction to zero and rerun. Do you see signs of cyclical spillover?

- Try to calculate some other convolutions numerically

  For example a convolution of a Gaussian with Gaussian should be another Gaussian where the width/mean is the sum of the two input Gaussians width/mean (note that you may need to fix one of the Gaussians width parameters as only the sum of the width can be constrained by the fit).

  Next try a convolution of an exponential distribution with Gaussian and see how well the fit can resolve the Gaussian smearing versus the exponential slope of the convoluted distribution. To successfully fit this convolution you should increase the spillover buffer size of the FFT convolution to 1.0 (as the exponential takes on large values close to the boundaries).