87944 - STATISTICAL DATA ANALYSIS FOR NUCLEAR AND SUBNUCLEAR PHYSICS

Module 3 : Laboratory of Stat. Data Analysis for Nucl. and SubNucl. Physics
teacher: G. Sirri

Hands-on 2: RooFit Workspace, Working with Likelihood, Fitting Errors,

To be submitted in one week

DOCUMENTATION:

- slides shown during the lecture, available on VIRTUALE:
- RooFit website: https://root.cern/manual/roofit/
- RooFit Manual (PDF A4 format)
- RooFit Quick Start Guide (PDF A4 format)
- RooFit tutorials: https://root.cern/doc/master/group__tutorial__roofit.html

Download

roofit_empty.cpp	an empty macro
roofit_empty.ipynb	an empty notebook

[0] WARM UP

Download and run the tutorials **rf502**, **rf503** (RooWorkspace), **rf511** (factory) and **rf601** (Minuit)
https://root.cern/doc/master/group__tutorial__roofit.html

[1] Hands-on: RooWorkspace, Factory, Composite Model; Working with Likelihoods

Exercise 11 - Composite Model; Working with Likelihoods

Note: RooMinuit is replaced by RooMinimizer starting from ROOT v6.30

Using the RooFit factory, create a model with the following components:

- Observable: A variable named “x” defined in the range [−20,20].
- Model: The sum of two Gaussians: f·gaus1(..)+(1−f)·gaus2(..)
    - Both Gaussians share the same mean, **“mean”**, fixed at 0 (constant).
    - The standard deviation (σ) of the first Gaussian, **“s1”**, is 3.K, where K is the last digit of your matriculation number. This is a constant.
    - The standard deviation (σ) of the second Gaussian, **“s2”**, is defined within the interval [3,6] and has an initial value of 4.
    - The coefficient **“f”** has an initial value of 0.5 and is not constant.

Note: This model has strong correlations.

Tasks:

1. **Generate a Dataset** with 1000 events.
2. **Save the Workspace** (model + data) to a file.
3. **Minimize the Likelihood**
   Pass the likelihood directly to a RooMinuit object and minimize it (as shown in the slides).
    - Construct an unbinned (negative log) likelihood for the model with respect to the generated data using the method RooAbsPdf::createNLL(RooAbsData&). The object returned by this function is of type RooAbsReal*.

- Create a MINUIT interface (a RooMinuit object) to this likelihood function.
- Enable verbose mode to display the MINUIT steps (use the RooMinuit::setVerbose(kTRUE)).
- Call MIGRAD to minimize the likelihood.
- Display the values of the parameters **f**, **mean**, **s1**, and **s2** using RooRealVar::Print().
  Note: the parameter values (and their errors) now reflect the results of MIGRAD.

4. **HESSE Error Calculation**
   - Disable verbose mode.
   - Call HESSE to compute errors using the second derivative of the
   - Display the values of the parameters **f**, **mean**, **s1**, and **s2** using RooRealVar::Print().
     Note: the parameter errors now reflect the results of HESSE.
5. **MINOS Error Calculation for "s2"**
   - Call MINOS for the parameter **"s2"**.
   - Display the values of the parameters **f**, **mean**, **s1**, and **s2** using RooRealVar::Print().
     Note: Observe that the errors for **"s2"** are now asymmetric. [C]
6. **Save Results**
   Save the fit results:
   - Take a snapshot of the fit result using the method RooMinuit::save().
     The object returned by this function is of type RooFitResult*.
   - Print the result in verbose mode using Print("v").
     Note: The snapshot contains the initial and final parameter values, the correlation matrix, the EDM value, the FCN value, the last MINUIT status code, and the number of times MINUIT encountered computational problems (e.g., null probability during likelihood evaluation).
   - Visualize the correlation matrix from the fit result
     gStyle->SetPalette(1) ;
     fit_results->correlationHist()->Draw("colz") ;
7. **Contour Plot**
   - Create a contour plot for **"f"** vs **"s2"** at 68%, 95.45%, 99.73% confidence levels using the method `RooMinuit::contour(var1, var2, n1, n2, n3)`.
     The object returned by this function is a plot frame of type RooPlot*.

     NOTE: Contours can be drawn using the arguments n1, n2, 3 to request the desired coverage in units of σ = 1, 2, 3 (see RooMinimizer::contour()).
     The minimizer automatically adjusts these values based on:
     - Coverage: Values are taken from the column (M = 2) of Table 40.2 in the PDG Review:
     https://pdg.lbl.gov/2024/web/viewer.html?file=../reviews/rpp2024-rev-statistics.pdf
     - Statistical scale: This considers whether the function to be minimized is a Chi² or a Negative Log-Likelihood (see RooMinimizer:SetErrorDef())
   - Draw the plot frame (using the standard Draw() method) and save it to a file.

Hands-on activities continue on the next page…

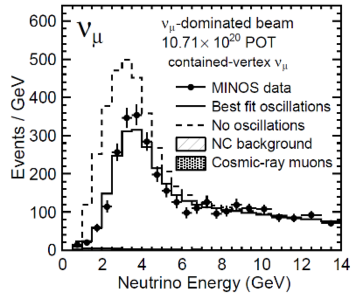
[3] Hands-on: Composite Model; Working with Likelihoods (ADVANCED)

NOTE THAT the word “MINOS” refers to the name of a neutrino experiment

Note: RooMinuit is replaced by RooMinimizer starting from ROOT v6.30

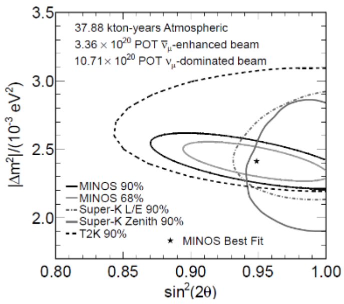
MINOS
composite model; working with likelihoods

Inspired by Figure 1 and Figure 2 of: “Measurement of Neutrino and Antineutrino Oscillations Using Beam and Atmospheric Data in MINOS” arXiv:1304.6335v3 [hep-ex] 10 Jul 2013
https://arxiv.org/abs/1304.6335



Neutrino oscillation provides direct evidence that neutrinos have non-zero mass and represents the only phenomenon observed to date with an origin beyond the Standard Model of particle interactions.

The MINOS Experiment was a long-baseline neutrino experiment designed to observe the phenomena of neutrino oscillations, an effect which is related to neutrino mass. The observed energy distributions of muon neutrino charged-current interactions in the MINOS Far Detector depends on the resulting neutrino oscillation probability, i.e. on mixing angles and on the differences between the squared neutrino masses.



The MINOS experiment performed precision measurements of oscillations via ν_μ disappearance. These oscillations are well described by an effective two-flavor model with flavor and mass eigenstates related by a single mixing angle θ

In this approximation, the ν_μ survival probability is given by

$$P(\nu_\mu \rightarrow \nu_\mu) = 1 - \sin^2 2\theta \ \sin^2(1.267\ \Delta m^2\ L\ /E\ ) \qquad \text{[Eq. 1]}$$

where L (km) is the distance traveled by the neutrino, E (GeV) is its energy, and Δm^2 (eV²) is the mass splitting.

Download:

minos_2013_data.dat	unbinned dataset of a neutrino oscillation experiment
minos_2013_mc.dat	unbinned dataset of simulated neutrino oscillation experiment

Define your observable, which is the reconstructed neutrino energy. This variable ranges from 0.5 to 14 GeV

Load the **unbinned** dataset with events observed by the MINOS experiments

*Hint: RooDataSet data = *RooDataSet::read("minos_2013_data.dat", energy, "v");*

Then, build a model to describe such spectrum using the following instructions:

- The energy distribution for non-oscillated neutrinos is simulated by a Monte Carlo. The resulting unbinned dataset is stored in the file *minos_2013_mc.dat*.
  Load this file into a RooDataSet object “mc_noosc”, as done for the observed data.
- Then, use this Monte Carlo dataset to create an histogram-based function which represents the shape of the energy distribution for non-oscillated neutrinos
  Hint: an histogram-based function. is represented in RooFit by the RooHistFunc class.
  Use the method binnedClone() to create a RooDataHist from the RooDataSet;
  ```
  RooDataSet* dd = (RooDataSet*) mc_noosc.reduce(RooArgSet(e)) ;
  RooDataHist* dh_mc_noosc = dd->binnedClone();
  ```
- *then create a RooHistFunc object from the RooDataHist.*
  ```
  RooHistFunc func_noosc { "func_mc_noosc", "No oscillation", e, *dh_mc_noosc, 2 };
  ```
- The shape of the energy distribution for oscillated neutrinos is obtained by multiplying this function by the oscillation probability shown in [Eq. 1]. The oscillation probability depends on mixing $\sin^2 2\theta$, mass splitting Δm^2, neutrino energy (observable) and distance $L = 730$ km (constant).
  - Create the variables **mixing** (i.e. $\sin^2 2\theta$ as a whole) and **dm2** (i.e. Δm^2).

- Create a `RooFormulaVar` to evaluate the oscillation probability expressed by the [Eq. 1]
- Create the final p.d.f of the energy distribution of oscillated neutrino using a RooGenericPdf

  Hint:
  ```
  model = RooGenericPdf{ "model", "model", "@0*@1",
                         RooArgSet(prob_osc, func_noosc) };
  ```
- Don't care about normalization. RooFit will adjust it for you.

Now you can fit the model to the data. Plot data and model. Save the plot as minos_data.png.

PART 2:

You shall minimize the likelihood explicity by hand using a RooMinuit object (look to the slides

- Construct a function object representing the negative log likelihood of the model with respect to the data.
  Hint: use *RooAbsPdf::createNLL(RooAbsData&), the returned object is a RooAbsReal**
- Create a MINUIT interface object (RooMinuit)
- Activate verbose logging of MINUIT parameter space stepping
  Hint: RooMinut::setVerbose(kTRUE)
- Call MIGRAD to minimize the likelihood
- Print values (use: RooRealVar::Print()) of all parameters (dm2, mixing), that reflect values (and error estimates) that are back propagated from MINUIT as evaluated by MIGRAD [A]
- Disable verbose logging
- Run HESSE to calculate errors from the second derivative at maximum
- Print values of all parameters (dm2, mixing), that reflect values (and error estimates) that are back propagated from MINUIT as evaluated by HESSE [B]
- Run MINOS on dm2 parameter only
- Print value of dm2 parameter, that reflect value (and error estimates) that are back propagated from MINUIT as evaluated by MINOS(the algorithm) [C].
  Please note that the error estimates is not symmetric.

Saving results.

- Save a snapshot of the fit result. This object contains the initial fit parameters, the final fit parameters, the complete correlation matrix, the EDM, the minimized FCN , the last MINUIT status code and the number of times the RooFit function object has indicated evaluation problems (e.g. zero probabilities during likelihood evaluation).
  Hint: use *RooMinuit::save()* ; the returned value is a *RooFitResult* type. Then, call* Print("v") . [D]

Contour plot.

- Make contour plot of dm2 vs mixing at 1,2,3 sigma
  Hint: use *RooMinuit::contour(var1, var2, n1, n2, n3); the returned value is a RooPlot* frame;*
  plot the frame using the method Draw(..) as usual

  NOTE: Contours can be drawn using the arguments n1, n2, 3 to request the desired coverage in units of σ = 1, 2, 3 (see [RooMinimizer::contour()]()).
  The minimizer automatically adjusts these values based on:
  - Coverage: Values are taken from the column (M = 2) of Table 40.2 in the PDG Review:
  https://pdg.lbl.gov/2024/web/viewer.html?file=../reviews/rpp2024-rev-statistics.pdf
  - Statistical scale: This considers whether the function to be minimized is a Chi² or a Negative Log-Likelihood (see [RooMinimizer:SetErrorDef()]())

- Zoom (by hand) the plot axes to better visualize the countour.
  Save the plot as minos_likelihood.png

(submit source code, minos_data.png, minos_likelihood.png and a text file with the fit results obtained in [A], [B], [C], and [D])