Data Analysis Tutorials

Introduction

The following exercises will take you through the various steps of analysing data in nuclear physics. The tutorial will focus on the use of c++ ROOT, however, for individual steps you are welcome to use pen & paper, excel, radware, python etc, though ROOT use is encouraged.

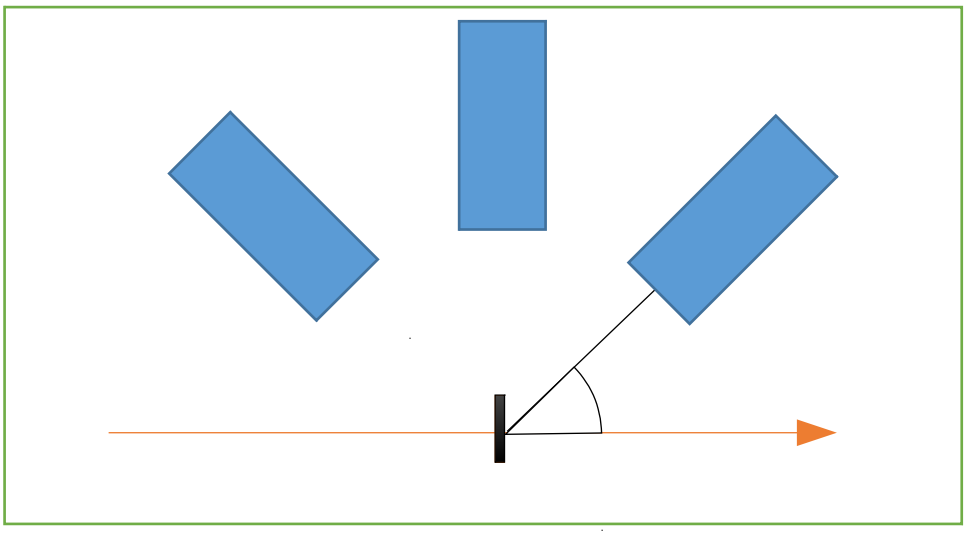
The majority of tasks in ROOT can be accomplished with the Graphical User Interface supplemented by entering commands on the command line. Or by executing a series of commands in a macro script .C file. Using the [tab] autocomplete functionality on the ROOT command line is an effective way of checking available functions and required inputs. The online ROOT class documentation is another invaluable source of information:
https://root.cern.ch/doc/master/classTH1.html
https://root.cern.ch/doc/master/classTF1.html

Do utilise the instructors and each other.

To get the most from these exercises, don't read too far ahead.

The Experiment

Data are taken with an array consisting of 3 gamma-ray detectors positioned around a target position on an accelerator beam line. The 3 detectors are positioned, with respect to the target and beam, at angles of 45,90,135 degrees.



Data are stored in root "trees", effectively long lists of individual events; an event being a single moment in time, in which the data acquisition computer detected an input and recorded all data from detectors during a short time window.

A series of .root data files are provided from the experiment:

Run1 – A source of ^{60}Co at the target position
Run2 – A source of ^{152}Eu at the target position
Run3 – A source of ^{133}Ba at the target position
Run4 – An A=36 beam with an energy of 250 MeV impinging on a thin A=40

target.
Run5 – A decaying source feeding the nucleus observed in Run4

Important note :
Before attempting to open the run files in a root session, run the command ".L Det.h" to load the data format library for the experiment. This is achieved with an include statement in the example scripts.

Exercise 1 – Looking at the Data

Before moving on to complex tasks ensure you can access, view and sort the events. This can be done using the GUI by creating a instance of the TBrowser class on the command line and then navigating to the desired root file. Histograms and graphs in a TFile can be viewed directly in a TBrowser by double clicking.

Data "TTrees" can be interacted with in the TBrowser, or in more detail by right clicking and starting a TTreeViewer (StartViewer).

In the TFile Run1.root you will see a TTree named AnalysisTree. Inside you should see several members of the data class that holds the hit information shown as TTree "leaves". This include detector ID, the event timestamp and the recorded charge.

Double click Hit.Charge to draw the recorded values for all events in the tree into a histogram with some default binning.

A) How many events are there in Run1.root ?

.....

Now, in the ROOT command line, run the following command to simultaneously define and fill a histogram, rather than using the default binning:

AnalysisTree->Draw("Hits.Charge>>h1(BINS,FROM,TO)")

You MUST include the **quote marks**. **BINS** is the number of bins in your new histogram, **FROM** is the lower bound of the X axis and **TO** the upper bound.

h1 is the "*name*" of the histogram that will be created, this is the identifier in the ROOT memory system that is used to access the histogram on the ROOT command line etc. *This is different to the title*.

For more complex sorting using a script is advisable. More advanced users might prefer to use a TSelector, but this is not needed or recommended for beginner/intermediate users. An example sorting script is provided. Load it with the following command (within root)

.L ExampleSort.C

Then execute the sorting function with :

ExampleSort("Run1.root")

Output histograms are saved to a new .root file which can then be viewed in a TBrowser.

B) Create and view a 1D histogram of the charge from the detector with ID=0 in Run1.root with an appropriate range and binning.

You can modify ExampleSort.C OR use the command:
AnalysisTree->Draw("Hits.Charge>>h1(BINS,FROM,TO)","Hits.ID==0")

C) What is the integral of counts between 0 and the highest point of the spectrum?

.....

Use the TH1::Integral() function. TH1::GetXaxis()->FindBin(x) can also be useful as commands require histogram “bin number” argument ***not*** x value.

D) Over what range of time do the events of Run1.root occur?

.....

Use any of the above methods to plot Hits.TimeStamp and view the extent of values covered.

Exercise 2 – Calibrate Detectors

Detectors actually record an induced charge which is proportional to the incoming energy. Due to differences in detectors and amplifiers all channels in an experiment must have the conversion between charge and energy determined from data of known energy.

1. 60Co.csv contains known data of gamma-rays from the decay of 60Co
2. Create and fill and appropriately binned histogram of the charge detected by each detector in Run1
3. Looking at the charge spectra you have created, identify the largest peaks. The charge centroid of these peaks must correspond to the energy of the most intense gamma rays expected from the decay.
4. Determine the centroids of the peak by fitting the peak with a Gaussian. This can be done in 1 of 3 ways:
    a. Using the FitPanel GUI by right clicking the on the histograms on the line of a drawn histogram and selecting “FitPanel”.
    b. Use a script such as ExampleFit.C
        i. .L ExampleFit.C
        ii. ExampleFit(histogram)
    c. On the ROOT command line by executing functions such as those called in ExampleFit.C
5. Calculate initial calibration parameters for the detectors.
6. Modify ExampleSort.C in order to produce a histogram of energy in keV

E) What is the energy resolution of the detector array (combined spectra of all channels)?

……% at …… keV

7. Sort an energy spectrum of Run2.root

F) What is the energy resolution of the 122 keV?
Give 2 reasons for the change from your answer to E.

……

……

……

……

……

……

……

Exercise 2 Extra

Use the ^{152}Eu data from Run2.root to improve your calibration