Building a chi2 variable for particle identification

Many times in experimental particle physics it is necessary to properly combine the information brought by a certain set of observables.

In this example we will take an observable and we will look how it distributes for background and signal events, noticing that the signal is almost disappearing over the background.

By using the additional information coming from the knowledge of how signal-like events are distributed we will create a chi2 observable out of the same variable.

We will look then at the resulting chi2 distribution of signal and background events.

Goal of these exercises

The aim of these exercises is to build a chi2 distribution out an observable (given a data sample) and look at its capability in selecting the signal while rejecting the background.

Basic informations: you will work on a set of events coming out from a PAMELA experiment simulation. The given file

`/home/mocchiut/pamela/data/pamelasimu.root`

contains the TTree pamcalotree, storing data with the PamCalo class, header file:

`/home/mocchiut/pamela/PamCalo/inc/PamCalo.h`

so library:

`/home/mocchiut/pamela/PamCalo/lib/Linux/libPamCalo.so` .

The ROOT file contains about 5.700.000 events: protons, electrons and positrons mixed together in an energy range from 10 to 300 GeV.

Positrons to electrons ratio in this file is about 0.1 . Electrons to protons ratio in this file is about 0.007 .

In these exercises, consider only the energy range [18-20] GeV. We want to create the chi2 variable using the observable a≡nstrip/qtot. Signal-like events are represented by electrons (`pID==1`), background events are represented by protons (`pID==0`), signal events by positrons (`pID==2`).

Exercise 1

Write an an <u>executable compiled program</u> which reads the input file

`/home/mocchiut/pamela/data/pamelasimu.root`

and gives as output a new ROOT file containing a TTree with three variables (a TBranch for each one):

- pID
- a (≡nstrip/qtot)

Save into the new file events which satisfy the following conditions:

1. the event lays in the energy range 18 – 20 GeV (hint: pay attention to the sign of “energy”, use “fabs”!).
2. the variable “noint” is less than 3 for each event (`pc->noint<3`)

Hints:

- to compile, remember to add also the compilation flags:

```
-I/home/mocchiut/pamela/PamCalo/inc
-L/home/mocchiut/pamela/PamCalo/lib/Linux/
-lPamCalo
```

- to run, remember to export LD_LIBRARY_PATH:

```
export LD_LIBRARY_PATH=/home/mocchiut/pamela/PamCalo/lib/Linux/:$LD_LIBRARY_PATH
```

- the output file should have a size of about 142K, if you have quota problem you can write the output on the linux temporary directory “/tmp/”.

Exercise 2

Write a ROOT-CINT script which reads the output file of exercise 2 (should be similar to this one: `/home/mocchiut/scripts/EM_output_190913.root` use this file if you are not able to complete or run exercise 1) and gives as output on the screen and on the disk (pdf format) a TCanvas divided into two pads (one columns, two rows – hint: `TCanvas::Divide`) with:

1. In the top panel the event distribution histogram (`TH1D`) of `a` for the signal events (`pID==2`) in red, the background events (`pID==0`) in green and the sum of background and signal events (`pID==2 || pID==0`) in red. (Hints: fill three different histograms - range [0,1] - and draw them on the same pad using the option "`same`"; set log scale for Y axis) ;
2. In the bottom panel the event distribution histogram (`TH1D`) for `a` for the signal-like events (`pID==1`) fitted with a Gaussian function G1(N1, $\mu 1$,$\sigma 1$);

Exercise 3

Update the script of exercise 2 in order to:

1. Calculate a chi2 variable for each event as: `chi2=`$((a-\mu 1)/\sigma 1)^2$ where $\mu 1$ and $\sigma 1$ come from the fit of exercise 2 (hints: should be about $\mu 1=0.118$ and $\sigma 1=0.011$);

3. draw a new TCanvas which contains the event distribution histogram (`TH1D`) of `chi2` for the signal events (`pID==2`) in red, the background events (`pID==0`) in green and the sum of background and signal events (`pID==2 || pID==0`) in red. (Hints: fill three different histograms - range [0,20] - and draw them on the same pad using the option "`same`") ;

Preparing the output

- create a directory named with the following format:

  YourInitials_C++2012

  (for example in my case it would be: EM_C++2012)

  put inside this directory ALL the files you want me to correct and look at.
- ALL files names format (but Makefile, if any) must be like:

  YourInitials_something.extension

  (for example in my case I would create files: EM_main.cpp, EM_myscript.C, EM_OutputHistogram1.pdf, etc. etc.)
- create a README text file (named like EM_README.txt), inside the file write:
  - **your name and surname**
  - a list of the files you are submitting
  - **in details** how to compile and run the programs
  - any other comment and answer to question(s) rised in the exercise description
- create a compressed tarfile containing the directory:

```
bash> ls
EM_C++2012
bash> tar zcf EM_C++2012.tar.gz EM_C++2012/
```

- copy the tarzipped file on the USB key I will circulate

Timing and rules

- You have four hours time to do your work.
- You can search the web, look at manuals, look at any note you wrote during the course, etc.
- We will discuss what you have written at the oral examination on 2013/09/20, until that (if needed) you can change and improve your programs. In that case prepare an electronic version we can look at during the oral examination, we will compare it to the one handed in today and we will discuss any change and/or correction.