Using the minimum chi2 estimation to determine the distance between two distributions

The cornerstone of almost all fitting is the Chi-squared method, which is based on the statistics of the Chi-squared function as defined:

$$\chi^2 = \sum_i^{N_d} \left[\frac{N_i(t_i) - f_i(t_i; a_M)}{\sigma_i} \right]^2 \qquad (1)$$

where the N_i(t_i) are the individual measurements (for example, the number of counts in one of the time bins), and the predicted value of the model is f_i (t_i ; a_M) where the a_M are the M parameters which are set to some reasonable trial value. The standard error of each measurement is the sigma_i in the denominator. There are a total of N_d measurements.

The statistical properties of the Chi-squared distribution are well-known, and the probability of the model's correctness can be extracted once this function is calculated. If the model has M free parameters, they can be varied over their allowed ranges until the most probable set of their values (given by the lowest Chi-squared value) is found. One can treat the M free parameters as coordinates in an M-dimensional space. The value of Chi-squared at each point in this coordinate space then becomes a measure of the correctness of that set of parameter values to the measured data. By finding the place in the M-space where Chi-squared is lowest, we have found the place where the parameters and model most closely match the measured data. This will ideally occur at a global minimum (eg., the deepest valley) in this M-dimensional space. There are many methods for finding the minimum of these M-parameter spaces. A "brute force" approach is to systematically vary our position in the M-space, and to then calculate the value of Chi-squared at each location that we visit. We will look for the lowest value, and also use some physical intuition to ensure that we did not just find some "local" minimum, rather than a global one.

Goal of these exercises

The aim of these exercises is to build a chi2 distribution and find its minimum (in the mono-dimensional case). This minimum will represent the distance between the mean of two Gaussian-distributed histograms.

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

`/home/mocchiut/pamela/data/pamsimu2013.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` .

In these exercises, consider only the energy range [2-20] GeV. We want to create the chi2 variable using the observables qtt1≡qtr/qtot and qtt2≡qtrack/qtot. We will use only events for which `pID==1`.

Exercise 1

Write an an executable compiled program which reads the input file

`/home/mocchiut/pamela/data/pamsimu2013.root`

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

- qtt1 (≡qtr/qtot)
- qtt2 (≡qtrack/qtot)

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

1. the event lays in the energy range 2 – 20 GeV (hint: pay attention to the sign of "energy", use "fabs"!).
2. the variable "pID" is equal to 1 for each event (`pc->ID == 1`)

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 45K, 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_170315.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 with:

the event distribution histogram (`TH1D`) of `qtt1,` in red, and of `qtt2,` in green, fitted with Gaussian functions. (Hints: fill two different histograms - range [0,1] - and draw them on the same pad using the option "`same`") .

Save in two variables "mean1" (qtt1) and "mean2" (qtt2) the mean values of the Gaussian fits and print them on the STDOUT.

Exercise 3

We want to print on the STDOUT “mean1-mean2” and the same distance determined using the chi2 minimization method. So, update the script of exercise 2 in order to:

1. Create a new TCanvas and a new TH1D histogram (let’s call it “qttshift”).
2. Create three float arrays of 10 elements each (`xchi2[10],chi2[10],echi2[10]`).
3. Start a loop over `int i` from 0 to 10 (ten excluded). For each iteration:
    a. create a float “d” defined as `i*0.05`;
    b. loop over all the entries of the file created in ex. 1 (i.e.: you will have a nested loop); inside this loop fill “qttshift” with the variable qtt2 incremented by d (hint: `qttshift->Fill(qtt2+d)`);
    c. fit the histogram qttshift with a Gaussian function and create the float variables `temp_chi2` defined as `((mean_fit – mean1)/sigma_fit )`^{2} and `temp_echi2` defined as `chi2[i] * sigma_fit/mean_fit` ;
    d. save d in the element i of `xchi2`, save `temp_chi2` in the element i of `chi2` and save `temp_echi2` in the element i of `echi2`;

    (Hints: remember to clean the TH1 at each iteration [TH1D::Reset()]).
4. Create a TGraphError using xchi2, chi2 and echi2 arrays (use NULL for errors on the x-axis).
5. Draw the TGraphError (in the new canvas created in point 1 of this exercise) and fit it with a “pol2” function.
6. Calculate (analytically) the x-value of the minimum of the pol2 fitted function and print it on the STDOUT together with the value “mean1-mean2”.

Preparing the output

- create a directory and put inside this directory ALL the files you want me to correct and look at.
- 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 2015/03/20, 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.