CLAS-NOTE 2001-014
August 20, 2001

EC Time Calibration Procedure For Photon Runs in CLAS

Matthieu Guillo
University of South Carolina

Dennis Weygand, Jingliang Hu, Carlos Salgado, Mina Nozarm
Thomas Jefferson National Laboratory

Contents

List of Figures

1 Motivation for EC time calibration

1.1 Photon / Neutron separation

Discriminating neutrons from photons is crucial in many channels studied in Hall B. Neutral particles are identified by a hit in the forward or large angle calorimeters with no matching track. Neutrons are dicriminated from photons based on β: the current code considers that neutrals with $\beta < 0.9$ are neutrons and photons otherwise. In order to achieve good separation, good EC timing is essential. Time resolution is related directly to β resolution according to the following formula:

$$\begin{aligned} \beta_0 &= \frac{L}{ct} && (1) \\ \sigma_{\beta_0}(\sigma_t, \sigma_L) &= \frac{\beta_0}{L}(\sigma_L + \beta_0\, c\, \sigma_t) && (2) \end{aligned}$$

- L is the distance from the vertex to the hit in the calorimeter and its value is roughly $5\,m$.
- σ_L is the distance uncertainty due to vertex resolution and hit position in the calorimeter. This value will vary according to the azimuthal angle but can be roughly estimated:
  $\sigma_L \approx 1cm$ for photons (they interact almost immediately in the calorimeter)
  $\sigma_L \approx 20cm$ for neutrons (asssuming they can interact with uniform probability in the thickness of the calorimeter)
- $$\beta_0 = \begin{cases} 1 & \text{for photons} \\ \frac{p}{\sqrt{m_n^2+p^2}} = \beta_0(p) & \text{for neutrons} \end{cases} \qquad (3)$$

  As the momentum increases, β becomes closer to 1, making the photon / neutron separation more difficult (Figure 1).

1.2 Need for a new calibration scheme

A package (*ec_time*) already exists to calibrate the calorimeter. The procedure was developed for electron runs, and uses both electrons and pions for

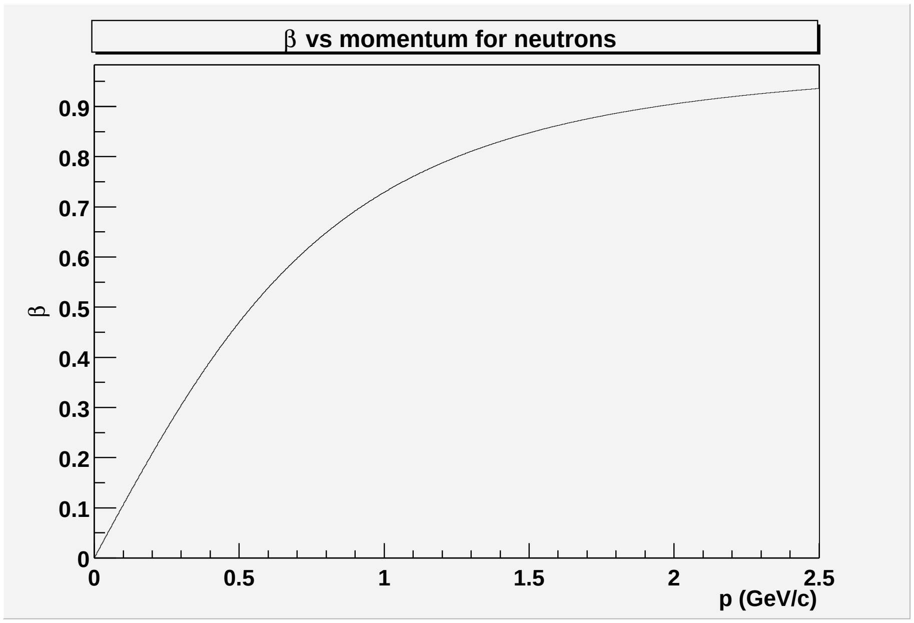


Figure 1: *β vs momentum for neutrons*

the calibration. Electrons are well suited; their velocity is c, and they interact on the surface of the calorimeter creating an electromagnetic shower. Pions, on the other hand, can interact anywhere inside the calorimeter or may just ionize: they will give poorer time resolution. For electron runs, this is not such an issue since every event has at least one electron which can be used for the calibration. Since CLAS toroidal magnetic field bends electrons in the forward direction, pions are used only at the highest azimutal angles. Using this scheme, time resolution varies between $250\,ps$ and $400\,ps$
In photon runs, generally we do not have electrons, and attempts to calibrate the EC using pions produced time resolutions between $500\,ps$ and $1\,ns$. We will show how much this affects photon / neutron separation and indicate the need for another scheme.
Let's assume a simple gaussian β distribution for both photons and neutrons at a given momentum p:

$$f_\gamma(\beta, \sigma_t) \quad = \quad \frac{1}{\sqrt{2\pi}\sigma_{\beta_0}(\sigma_t, \sigma_L)} e^{-\frac{(\beta-1)^2}{2\,\sigma^2_{\beta_0}(\sigma_t, \sigma_L)}} \tag{4}$$

$$f_{neutron}(\beta, p, \sigma_t) = \frac{1}{\sqrt{2\pi}\sigma_{\beta_0}(\sigma_t, \sigma_L)} e^{-\frac{(\beta-\beta_0(p))^2}{2\sigma^2_{\beta_0}(\sigma_t,\sigma_L)}} \quad (5)$$

where $\beta_0(p)$ has been defined in Equation 3 and $\sigma_{\beta_0}(\sigma_t, \sigma_L)$ in Equation 2. For $\beta < 0.9$, neutrals are called neutrons, whereas for $\beta \geq 0.9$, they are called photons. Figure 2 shows the particle misidentification due to the width of the distribution (function of the time resolution σ_t and position resolution σ_L) as well as the position of the neutron peak (function of the neutron momentum).

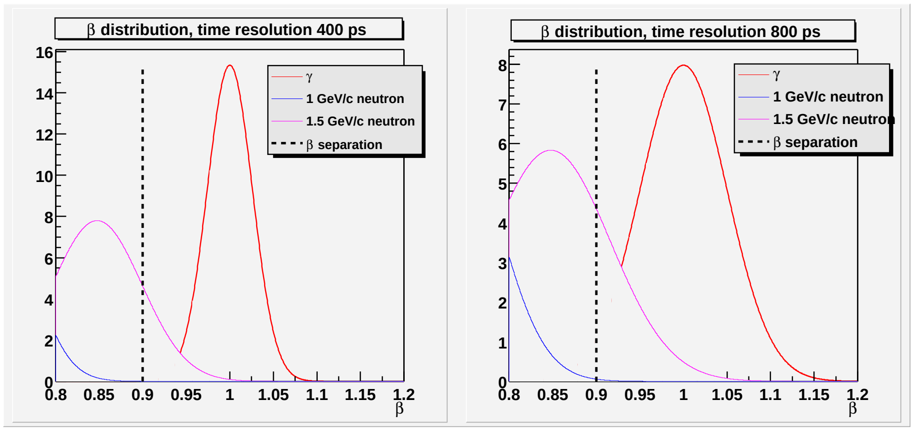


Figure 2: *β distribution for neutrons and photons*

For neutrons with momentum below $1\,GeV/c$, time resolution is not so important as the peak is far away from $\beta = 0.9$, but resolution effects show up clearly at higher momentum. For photons, the effects are less dramatic since σ_L is smaller, but still evident. The percentage of real neutrons (respectively real photons) misidentified is denoted by $m_N(p, \sigma_t)$ $(m_\gamma(\sigma_t))$, and in this simple model can be computed as:

$$m_N(p, \sigma_t) = \int_{0.9}^{\infty} f_{neutron}(\beta, p, \sigma_t)\, d\beta \quad (6)$$

$$m_\gamma(\sigma_t) = \int_{-\infty}^{0.9} f_\gamma(\beta, \sigma_t)\, d\beta \quad (7)$$

Figure 3 shows the results for some values of σ_t and neutron momenta.

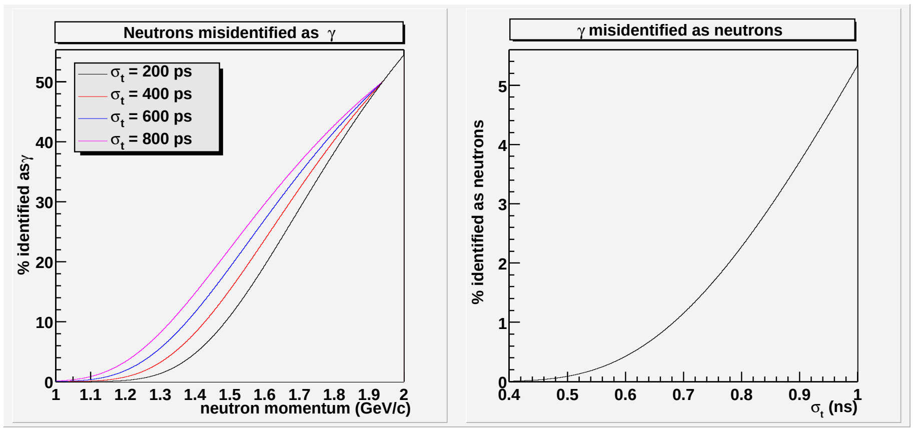


Figure 3: *Neutron and photon misidentification*

Time resolution has a small effect (compare the difference between $\sigma_t = 400ps$ and $\sigma_t = 800ps$) in the leakage of photons into neutrons. However, the effect is not neglectable for the leakage of neutrons into photons when the neutron momentum is between 1.2 and 1.8 GeV/c.
This very simple study takes into account only time resolution. If the time is systematically off, then the β peaks position are displaced from their correct values, and the effects are even worse. It is therefore crucial for the time to be correct and have the smallest uncertainty.

2 Method for calibration

We propose to use photons to calibrate the forward electromagnetic calorimeter (EC) in photoproduction. Photons interacting with the massive structure of the calorimeter will convert quickly into an (e^+, e^-) pair and produce an electromagnetic shower. A package has been written to achieve this goal, mainly in C++ and Perl to glue programs together. All programs have a help accessible through the "-h" flag. The package is available from the CVS repository ($CLAS_PACK/utilities/ecGammaCal); there is a README file that explains how to build and use the programs as well as the prerequisite libraries.

The whole calibration procedure requires 7 steps:

1. filtering events that can be used for calibration (program *filterGammas*).
2. computing the calibration constants (program *computeCalibConsts*).
3. computing the difference between the time expected and the time from the calibration (program *fitTimeDifference*). The difference is fitted using a gaussian.
4. rejecting gammas biasing the fit and giving poor quality calibration constants (program *rejectGammas*).
5. selecting the best calibration constants (program *getCalibConsts*).
6. checking the quality of the calibration constants (ROOT macros).
7. putting the numbers into the map.

Steps 2, 3 and 4 are performed in a cycle, the output of step 4 feeding the input of step 2. During each cycle, we create a new set of calibration constants as well as monitoring histograms. At the end of step 5, final calibration constants are selected from the previous sets and final monitoring histograms are produced. There should be no need for re-cooking the data in order to check the quality of the calibration constants.

2.1 Filtering events for calibration

2.1.1 Identifying photons

Since the EC timing has not been done at that stage, β for neutral particles is incorrect, so is the particle ID for neutrons and photons ie some particles identified as photons are really neutrons and *vice versa*. We need to rely on physics (not on particle id) if we want to use photons for calibration.
The current sheme is to select events where one, and only one, proton has been identified and one or many neutrals have been detected. Since the target is a proton, baryon number conservation forbids the neutral to be a neutron. Particle id for these neutrals is changed, if necessary, so that they are gammas.
Another sheme has been tested and failed. The idea was to use events where one proton and exactly 2 neutrals were detected. The neutrals were treated

as γ's, and their invariant mass computed. The π^0 peak stood up clearly in the mass distribution on top of some small background. This distribution was fitted using a first degree polynomial (for the background) and a gaussian (for the π^0 peak) (Figure 4).

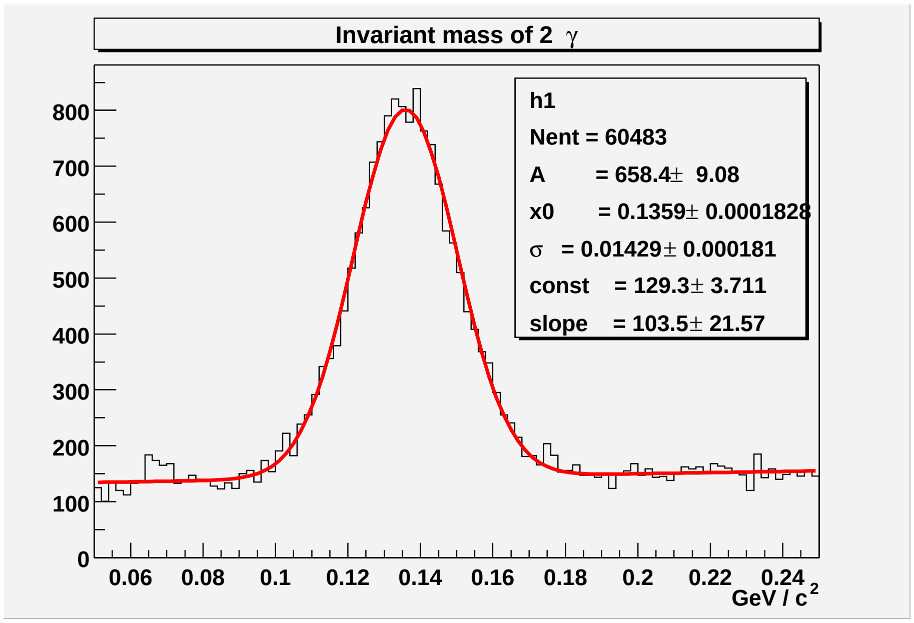


Figure 4: *Mass distribution of 2 γ*

We kept the event if the 2 neutral invariant mass was within 2 σ of the centroid of the π^0 peak: those neutrals were then considered to be true γ's. Unfortunately, no good calibration was obtained using this method. Figure 5 shows a typical hit in the EC and may give a possible explanation why the calibration failed: most of the photons using this selection method had hits in the same sector, leading to some overlapping signal. Those events confuse the calibration since we cannot know which fraction of the ADC is due to the first gamma and which fraction is due to the second. Also the time of the hit is ambiguous.

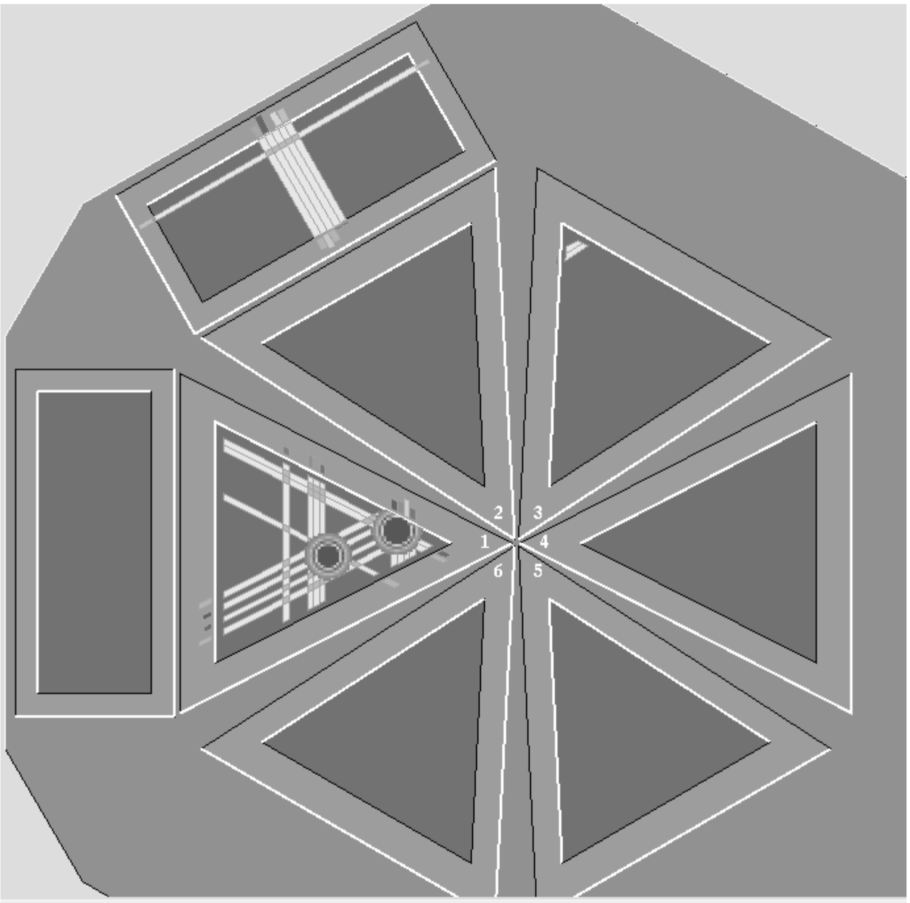


Figure 5: *2 γ's overlapping signals in the EC*

2.1.2 Selecting good events

Before starting, one has to select events that can be used for calibration. This is the purpose of the program *filterGammas*. A good event is one where in the final state, exactly one proton, at least 2 charged tracks (for better vertex reconstruction using the MVRT package) and at least one *good gamma*, after converting all neutrals into gamma, have been detected. A good gamma is defined by:

- all necessary BOS banks are present (PART, TBID, ECHB, ECPC and

MVRT)

- the hit in the EC is in the fiducial region.
- there are no other EC hits in the same sector. This prevents overlapping signal leading to misinterpretation when reading ADC's and TDC's values.
- the energy reconstructed from the energy deposited in the EC must be above some minimum value. This reduces accidentals as well as signal noise. The default value is $0.1\,GeV$.
- the inner layer of the EC must have recorded a hit (but we do not require a hit in the outer layer). This is the expected behavior of gammas interacting in the EC ; neutrons, on the other hand, can interact anywhere in the calorimeter. Requiring a hit in the inner layer gets rid of some neutron leakage into our gammas sample.

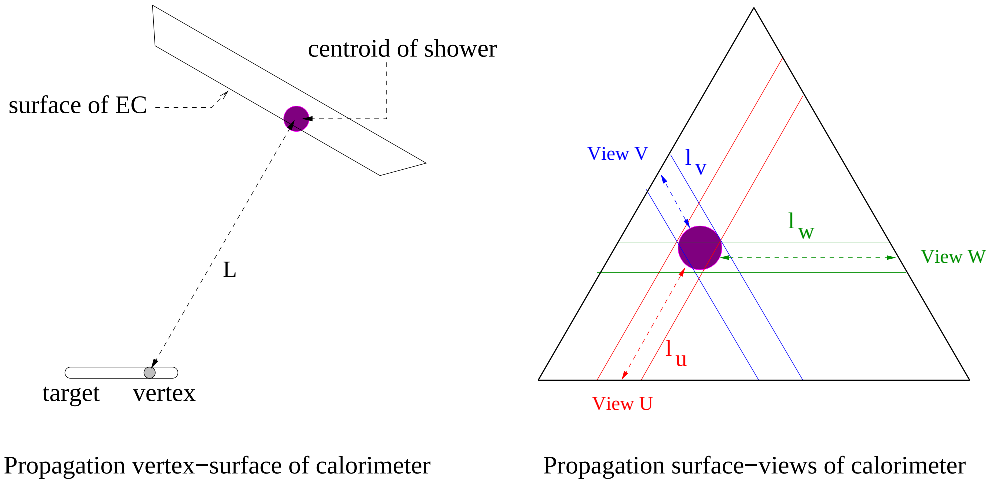
2.2 Computing the calibration constants

2.2.1 $T_{expected}$ versus T_{model}

We can evaluate the time for a photon to generate a signal in the EC PMT's using pure geometry and the speed of light in vacuum as well as in a scintillator. This time will be labelled $T_{expected}$.
The same time can also be evaluated using a semi-empirical model (using a set of 5 parameters denoted a_0 to a_4) based only on ADC, TDC, hit position on the surface of the calorimeter and the vertex time informations. This time will be labelled T_{model}.
For each PMT (indexed by j), we will have a set of $a_{j,k}$ $(k = 0, .., 4)$ and the goal of the calibration is to adjust them so that T_{model} matches as close as possible $T_{expected}$; the program *computeCalibConsts* is in charge of this part.

2.2.2 Expected time: $T_{expected}$

Let's consider a γ interacting in the EC. A shower is created and the light generated in the strips reaches some PMTS and triggers a signal (see Figure 6).



Propagation vertex−surface of calorimeter

Propagation surface−views of calorimeter

Figure 6: *photon's path to reach PMTs*

The time for a photon to propagate from the target to the EC strip and produce light seen by the corresponding PMT for a given view i (where i is either the u, v or w view) is:

$$T_{expected_i} = \frac{L}{c} + \frac{l_i}{v} \tag{8}$$

- $c = 30\, cm/ns$ since the photon travels in a medium with average index close to 1 before reaching the surface of the calorimeter.

- v is the speed of the light in the plastic scintillator and its empirical known value is $v = 18.1\, cm/ns$. Here, this speed is independent of the strip; in a better model, it would be function of the strip length as it is done for in the Time of Flight's calibration procedure.

- L is the distance between the vertex position and the centroid of the electromagnetic shower.

- l_i is the distance from the centroid of the shower to the edge of the view i and is measured along a strip. Its determination involves only geometry.

2.2.3 Model time: T_{model}

Since photons and electrons are expected to interact very similarly in the EC, we will use the same semi-empirical model used for calibrating the EC in electron runs.

$$T_{model_i} = a_0 + a_1 TDC_i + a_2 \frac{1}{\sqrt{ADC_i}} + a_3 l_i^2 + a_4 l_i^3 - T_{vertex} \tag{9}$$

The different terms have the following meaning:

- a_0: constant term. Includes all constant times (delays due to cables length for instance).
- $a_1 TDC_i$: TDC conversion term. Time from the conversion of TDC values into ns.
- $a_2 \frac{1}{\sqrt{ADC_i}}$: time-walk correction term.
- $a_3 l_i^2 + a_4 l_i^3$: light attenuation terms. They are part of the expansion of e^{-l_i/l_0}. The zero rank term is already included in the a_0 term and the first rank term is proportionnal to l_i hence to the TDC and is included in the $a_1 TDC_i$ term. Higher rank terms are neglected.
- T_{vertex} is the time at the vertex and needs to be subtracted since it is the reference time.

2.2.4 Fitting method

To find the best calibration constants $a_{j,k}$, we will use the *least squares method* on every single tube. Let's consider a given tube (indexed by j) where many hits (indexed by i), have been recorded (total number of hits is N_j) and the corresponding $T_{expected_i}$ and T_{model_i} have been computed. The χ^2 is given by:

$$\chi_j^2 = \sum_{i=1}^{N_j} \frac{|T_{expected_i} - T_{model_i}|^2}{\sigma_{j,i}^2} \tag{10}$$

$\sigma_{j,i}$ represents the time resolution due to statistical and systematical errors. At this time, only statistical errors have been considered and we have assumed a normal distribution to evaluate them; then $\sigma_{j,i}^2 = \sigma_j^2 = N_j$. Equation 10 can be rewritten as:

$$N_j\chi_j^2 = \sum_{i=1}^{N_j}[a_{j,0}+a_{j,1}TDC_i+a_{j,2}\frac{1}{\sqrt{ADC_i}}+a_{j,3}l_i^2+a_{j,4}l_i^3-\underbrace{(T_{vertex_i}+\frac{L_i}{c}+\frac{l_i}{v})}_{T_i}]^2 \tag{11}$$

Minimizing the χ_j^2 for the set of $a_{j,i}$ variables requires that each partial derivative with respect to $a_{j,i}$ is equal to 0. This is equivalent to minimize the sum in Equation 11 and can be written in a matrix form since the expression is a linear function of the $a_{j,i}$ parameters.

$$M_j A_j = V_j \tag{12}$$

Where:

$$M_j = \sum_{i=1}^{N_j}\begin{bmatrix} 1 & TDC_i & \frac{1}{\sqrt{ADC_i}} & l_i^2 & l_i^3 \\ TDC_i & TDC_i^2 & \frac{TDC_i}{\sqrt{ADC_i}} & TDC_i l_i^2 & TDC_i l_i^3 \\ \frac{1}{\sqrt{ADC_i}} & \frac{TDC_i}{\sqrt{ADC_i}} & \frac{1}{ADC_i} & \frac{l_i^2}{\sqrt{ADC_i}} & \frac{l_i^3}{\sqrt{ADC_i}} \\ l_i^2 & l_i^2 TDC_i & \frac{l_i^2}{\sqrt{ADC_i}} & l_i^4 & l_i^5 \\ l_i^3 & l_i^3 TDC_i & \frac{l_i^3}{\sqrt{ADC_i}} & l_i^5 & l_i^6 \end{bmatrix} = \sum_{i=1}^{N_j} M_{j,i} \tag{13}$$

$$A_j = \begin{bmatrix} a_{j,0} \\ a_{j,1} \\ a_{j,2} \\ a_{j,3} \\ a_{j,4} \end{bmatrix} \quad and \quad V_j = \sum_{i=1}^{N_j} T_i \begin{bmatrix} 1 \\ TDC_i \\ \frac{1}{\sqrt{ADC_i}} \\ l_i^2 \\ l_i^3 \end{bmatrix} = \sum_{i=1}^{N_j} V_{j,i} \tag{14}$$

For every photon we look to see which tubes j have been hit, compute the corresponding $M_{j,i}$ and $V_{j,i}$ and add them to the M_t and V_t matrices. At the end, the $a_{j,i}$ parameters, which are the elements of the A_j vector, are computed by:

$$A_j = M_j^{-1} V_j \tag{15}$$

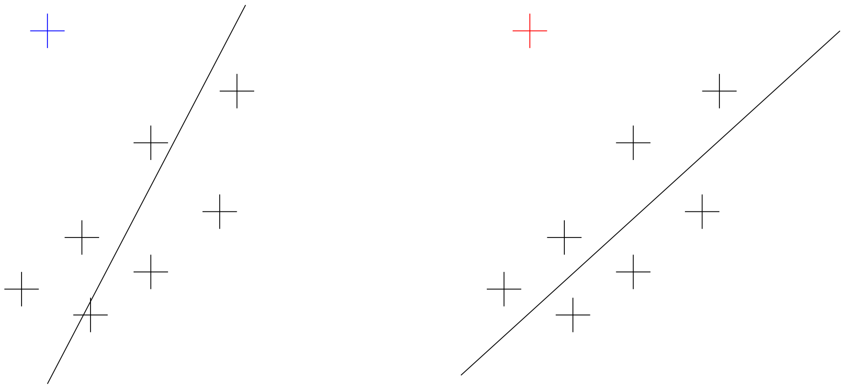
2.3 Fitting $T_{expected} - T_{model}$

Once we have calculated the calibration constants, we can check their quality by looking at the difference between $T_{expected}$ and T_{model}. This is the purpose

of the program *fitTimeDifference*.
$T_{expected} - T_{model}$ histograms distributions are created for each tube, and fitted with a gaussian. We chose ROOT analysis libraries both for the histograms and fitting packages since the libraries can be called seamlessly from our C++ code. The histograms, along with a tree that contains more information (in case of anomalies) are written into a ROOT file. A check of the histograms gives immediate feedback of the quality of the calibration, without the need for recooking the data using the new constants. For a good calibration, we expect the distribution for a given tube to be gaussian, centered at 0 and with width giving the time resolution of this tube.

2.4 Rejecting bad gammas

Unfortunately, the calibration can be biased by a few number of hits for which the time difference is very large. Figure 7 illustrates this point in a trivial example: fitting a line through a cloud of points.



Erroneous fit due to a bad point

Fit after rejecting bad point

Figure 7: *Method for rejecting bad points*

The method for rejecting bad points and getting better calibration constants is the following (this is done on a tube by tube basis):

1. from the histogram we compute the mean and RMS of the distribution.

2. we fit the histogram using a gaussian centered at the mean value and in the range spanning $[mean - RMS, mean + RMS]$ to obtain the centroid position and σ of the gaussian.

3. for every hit we compute the time difference $\Delta T = T_{expected} - T_{model}$ and flag the photon for rejection if the following condition: $centroid - n \times \sigma < \Delta T < centroid + n \times \sigma$ (n is controlled by the user, the default value is $n = 2$) is not respected.

The first 2 steps are actually performed at the fitting step by the program *fitTimeDifference*. The rejection program *rejectGammas* does the last step, using the centroid and σ values written by *fitTimeDifference*. The output of *rejectGammas* is a BOS file almost identical to the initial input of *computeCalibConsts*, but with some photons in the PART bank flagged not to be used the next time. Those are the files that need to be fed back to *computeCalibConsts* in order to have better calibration constants.

2.5 Getting the final calibration constants and checking their quality

The previous procedure can be repeated many times, each cycle bringing a new set of calibration constants. The procedure converges, each cycle resulting in better calibration constants; the last cycle should give the best calibration constants. There is one case where this is not true (and that's why we do not run this loop forever). Since at each cycle we reject photons, it may happen that after a couple of loops, some tubes no longer have enough photons for calibration, the calibration constants are therefore flagged by setting them at some arbitrary large values. The program *getCalibConsts* reads all calibration files, and for every tube selects the constants that were obtained using the minimum number of hits and having not being flagged: these are hopefully the best obtained.

To check the quality of this set of constants, we run *fitTimeDifference* again, but this time on the *whole* data set, including rejected photons. If everything worked properly, the ΔT distibutions should be a gaussian centered at 0 (due to the good photons) on top of some background (due to the rejected photons). The ROOT macro $s$howPasses.C provides the checking by drawing ΔT distibutions for every cycle.

A perl script *calibEC.pl* exists to go through all these cycles of finding the calibration constants, fitting the time difference, rejecting the bad photons and finally getting the best quality calibration constants. In normal use, the user should modify only the first lines to fit his needs. However, for more advanced users, each program can be tuned (throught a set of flags) to achieve better calibration.

2.6 Witting numbers into the Map

The last step is to write the calibrations constants into the map. This is the role of the program *putConstsInMap*. At this time, the program writes the numbers into a *EC_CALIB.map* Map but does not make the changes into the recent Mysql database. This will be part of a future upgrade.

3 Example of calibration: g6b run period

This section shows the results of the calibration procedure for the g6b running period. The goal is to show what can be achieved at this stage, compared to the previous calibration package (ec_time) and can be used as a reference to be checked against.
We used 7 cycles (passes): the reason behind this choice is that the calibration constants didn't seem to change much afterwards. Figure 8 shows the results of the calibration at different passes for all tubes (integrated).

For the first pass, the distribution is not centered at 0, the width is huge, and the overall shape is not gaussian. But already at pass 2 the centroid position, shape and width begin to be acceptable; adding more passes adjusts the shape as well as the width. Notice also that at each new pass the number of photons used decreases, since we reject more and more photons. Figure 9 shows the final result where we used the calibration constants from pass 7 and applied them to all our photons, including the rejected ones.

When comparing to pass 1 from Figure 8, two observations can be made. The first one is that indeed, this method works. The second one is that we can understand where the so-called "bad" photons are from: the "shoulders" at $-4\,ns$ and $-2\,ns$ one can clearly see in the distribution are an indication that bad photons are from a different beam bucket. Being able to distinguish them in the time-difference distribution gives us confidence that the calibration worked well. To obtain the overall (integrated) time resolution, we fit the distribution using the sum of a third degree polynomial and a gaussian. The

overall time resolution is $460ps$ and this includes good tubes as well as bad ones.
Figure 10 shows individual tube time resolution. The left column is the result of the first pass, the right column is after the seventh pass. We clearly see the improvements made by the technique of rejecting bad photons for original good tubes (first row), bad tubes (second row) and even for bad tubes with few statistics (last row).

Figure 11 and Figure 12 show the relative contribution of the different terms in the time model and the values of the calibration constants for every tube.

Finally, one can check the quality of the calibration by recomputing β for neutrals using the new constants. This requires to recooking the data. Figure 13 shows the results sector by sector while Figure 14 compares the old (using pions) versus the new (using gammas) EC calibration method and proves that, at least for photon runs, the new method is much better.

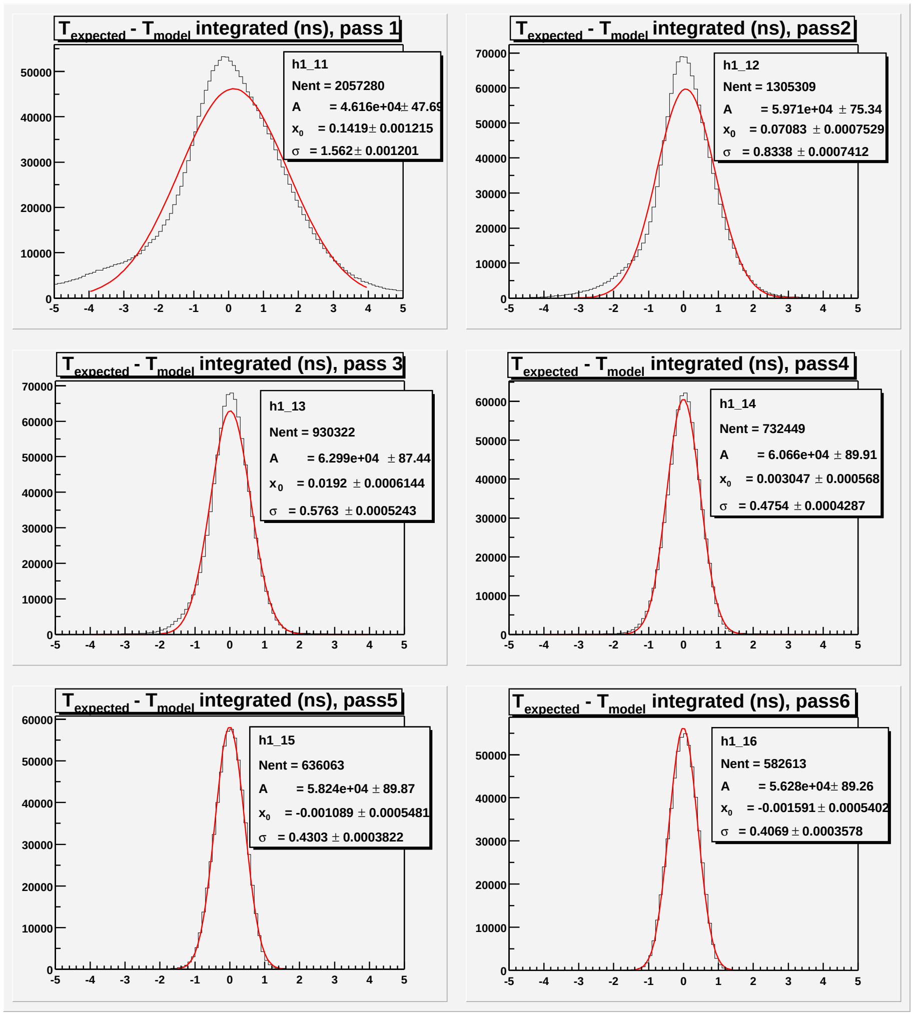


Figure 8: $T_{expected} - T_{model}$ *at different passes for all tubes*

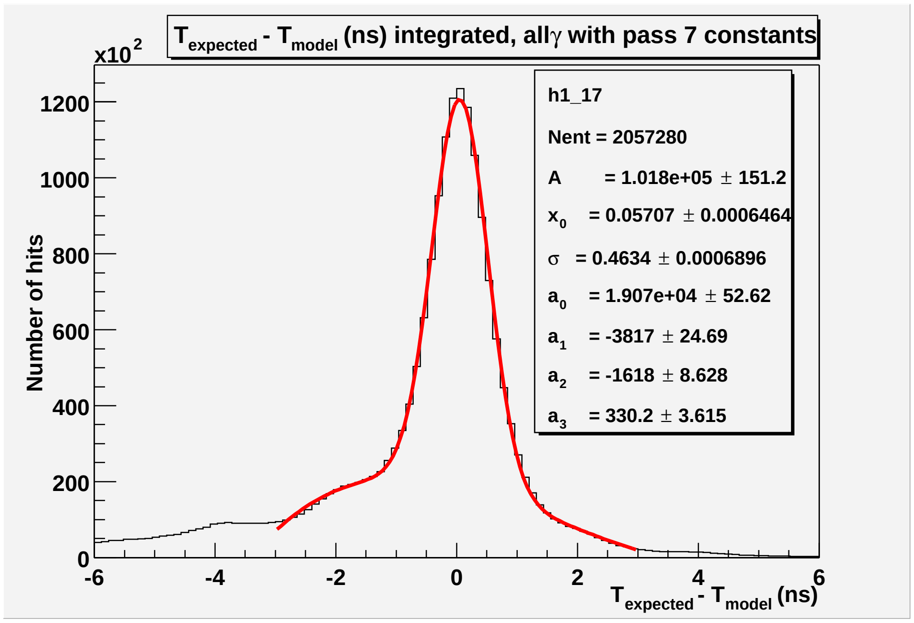


Figure 9: *Calibration results for all tubes*

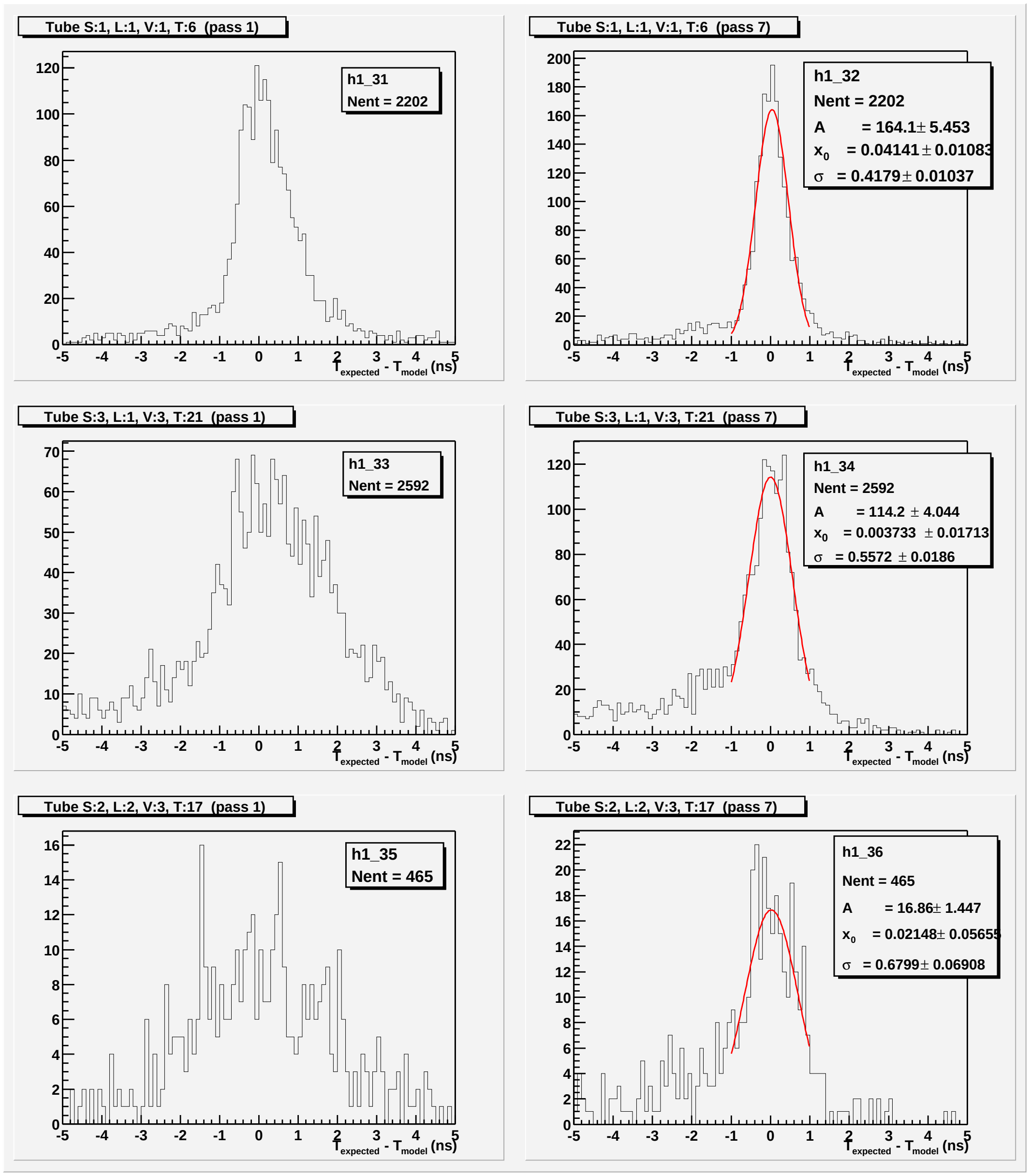


Figure 10: *Calibration improvements for individual tubes*

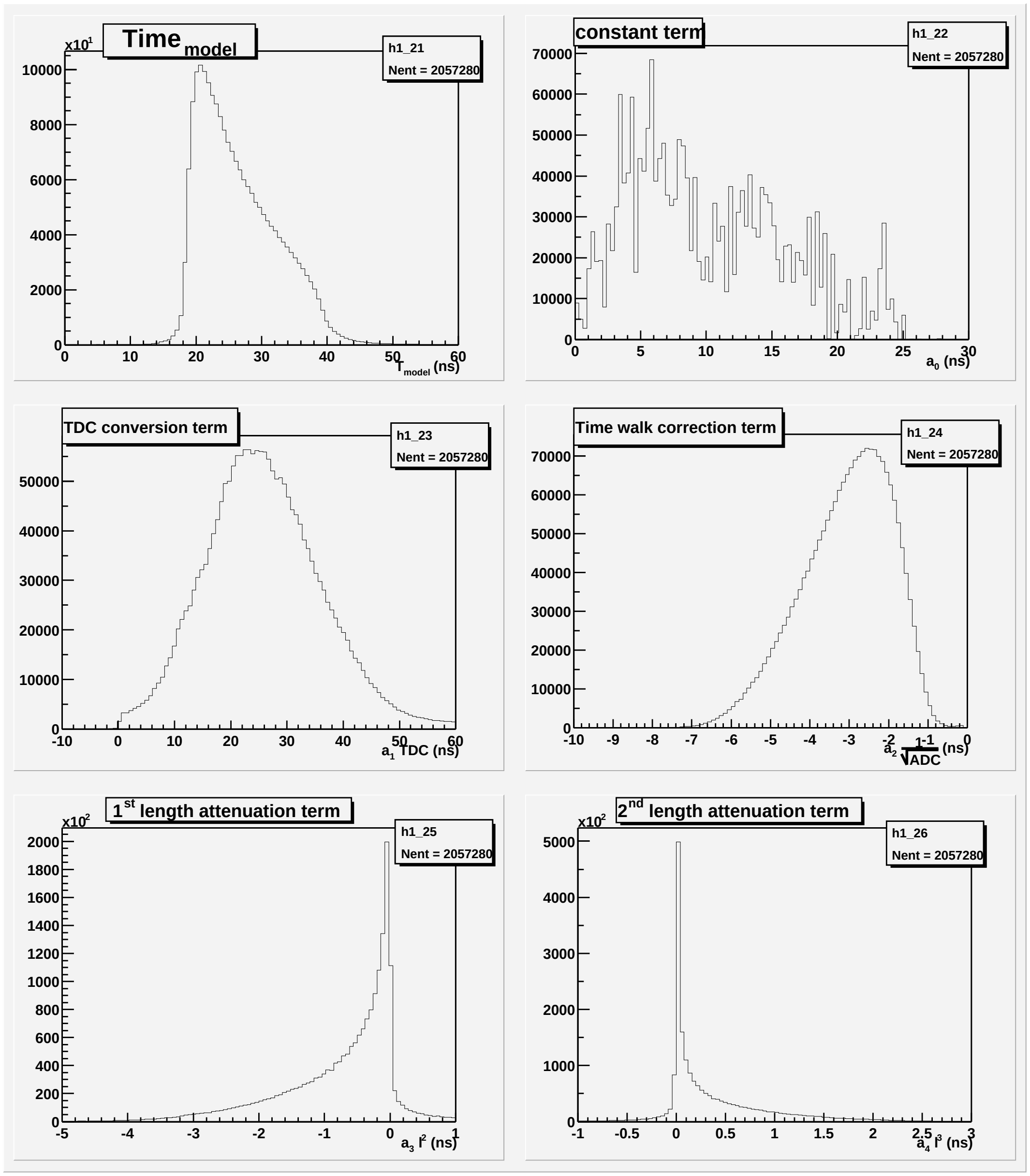


Figure 11: *Relative contributions of the terms in the time model*

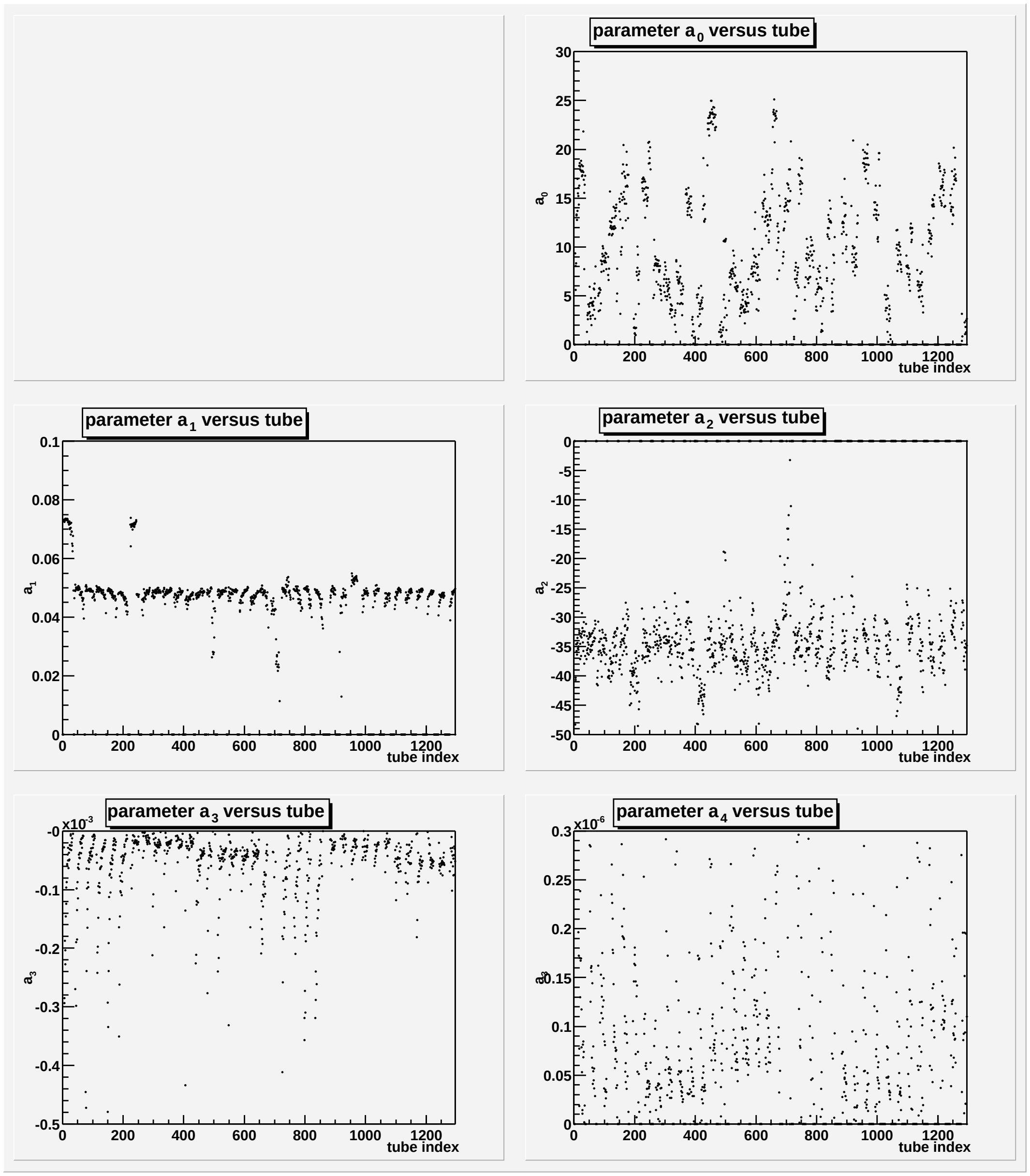


Figure 12: *Calibration constants values*

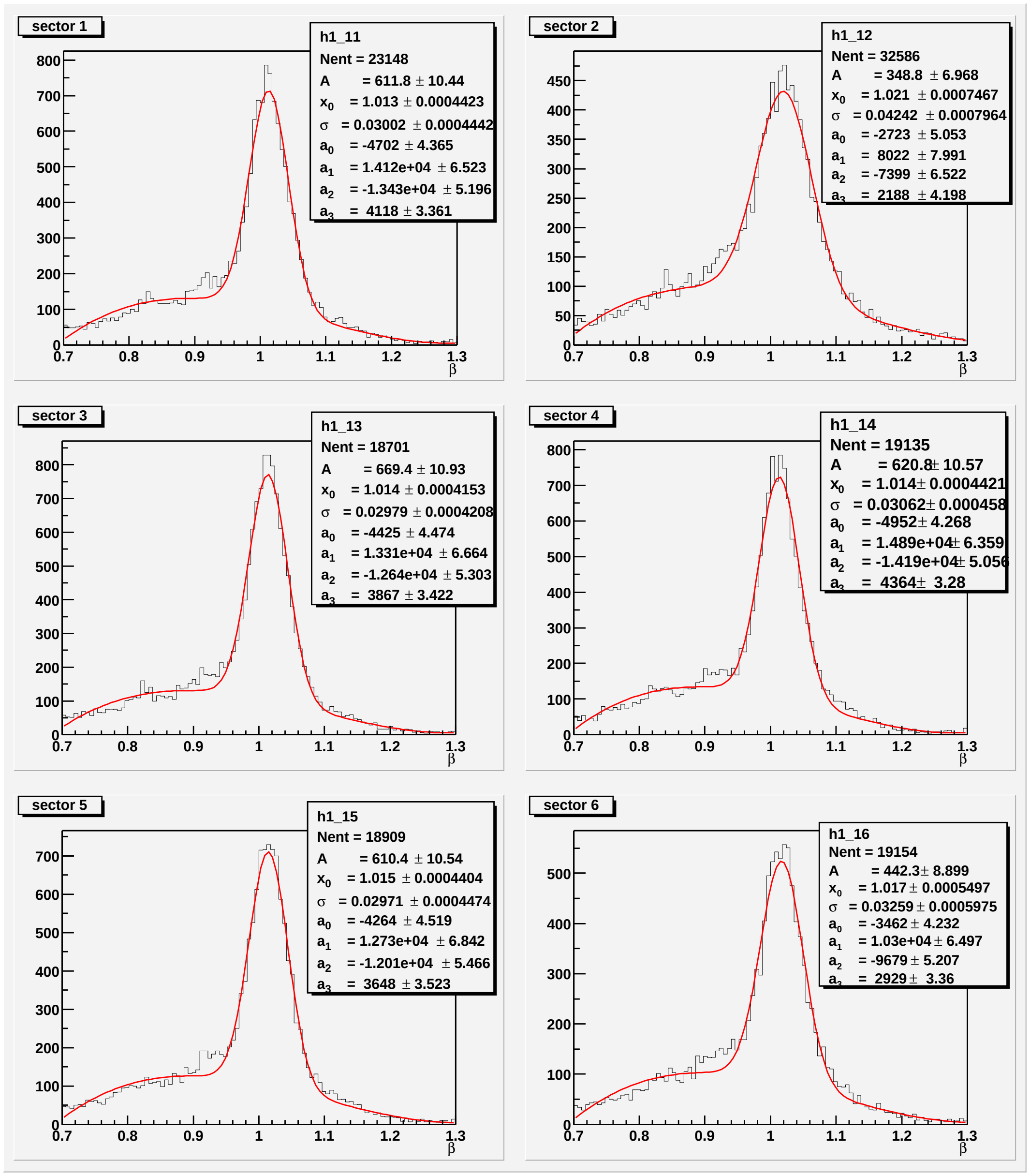


Figure 13: *β for neutral particles, sector per sector*

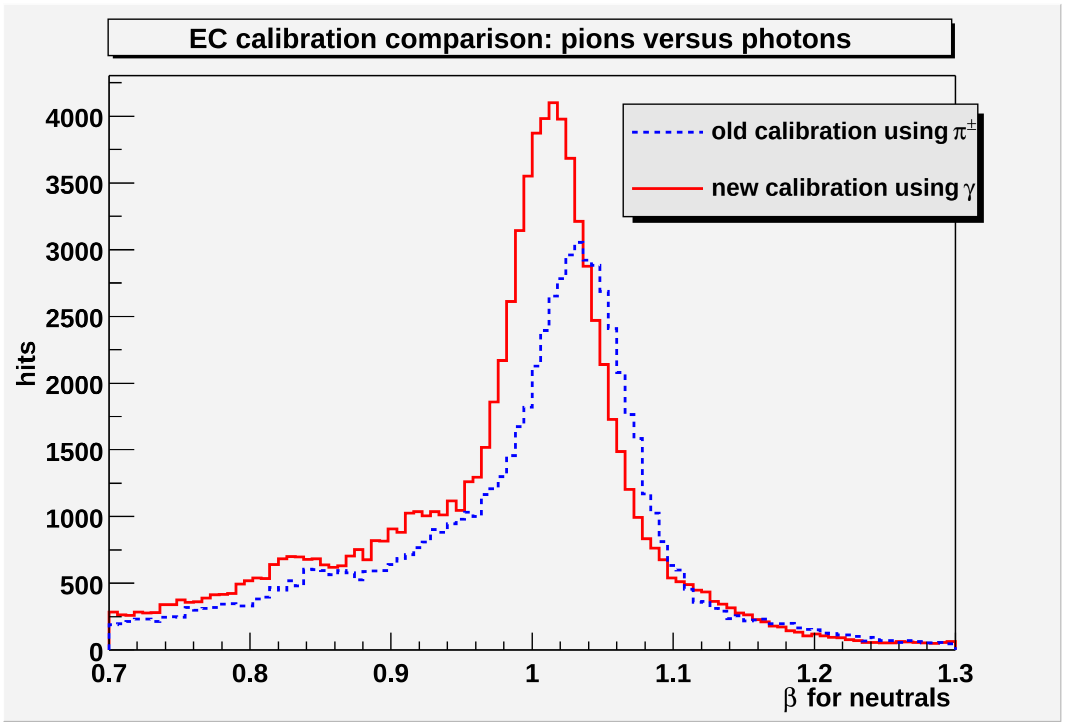


Figure 14: *Comparison between the old and the new calibration methods*