CLAS-NOTE 1999-04

Tagger hit reconstruction software and tagger calibration overview.

April 15, 1999

E.Anciant [1], G. Audit [1], T. Auger [1], C. Marchand [1], P.Girard[2].

[1] *Service de Physique Nucléaire, DAPNIA/DSM, CEA-Saclay, France*
[2] *Departement of Physics and Astronomy, University of South Carolina, Columbia, SC.*

Abstract

We present here an overview of the CLAS tagger setting, and data treatment sequence in order to approach the time calibration procedure. We also include a tutorial for the tagger calibration software we used for the G1 experiment, and discuss its results.

Contents

Introduction.

During photon beam operations, the tagger time is used to determine the interaction time at the vertex. In order to identify the electron beam bucket which gives a photon in the radiator and to increase the time resolution, the tagger TDCs and timing have to be precisely calibrated with respect to the machine

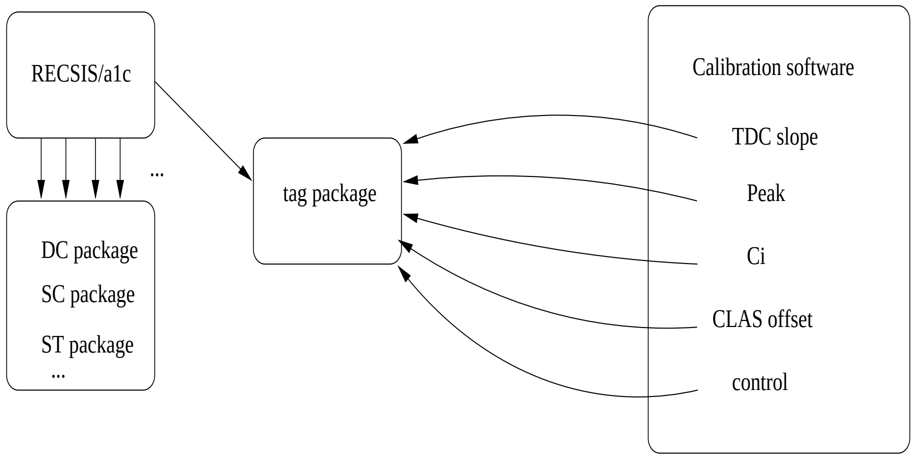

Figure 1: Calibration software organigram: The same code is used in the calibration software and in RECSIS/a1c

RF. This calibration will affect directly the particle identification in CLAS, since the tagger time will be taken in the event reconstruction as reference for the entire detector.

1. Calibration of the tagger is done by processing several times the datas using the RECSIS tagger code (tag package) (Fig. 1). Therefore, the first part of this document will describe the general tagger hits reconstruction procedure, used in RECSIS/a1c. Some details are given on the control parameters that allow to tune the processing. Explanations are given on the content of the package output bank TAGR. This part is of interest not only for people involved in the tagger calibration but also for anybody who wants to use the tagger reconstructed datas in RECSIS.

2. The second part describes the calibration methods that have been used to calibrate all the constants used in the tagger package, formerly via PAW kumacs, (running by hand several times RECSIS) and lately in a C++ program based on the CERN ROOT libraries, which automatically process RECSIS, and executes ten times faster than the kumacs. This part, rather technical, is mostly intended to peoples who want to enhance the calibration software.

3. The third part is a tutorial, which provides all informations on how to use this software and evaluate the quality of the calibration via control histograms.

4. The fourth part gives some additional informations on the tagger performances.

TAGT:

1	ID	I	1	62	! the address of the hit detector element
2	TDCL	I	0	4096	! tdc information (Left counters channels)
3	TDCR	I	0	4096	! tdc information (Right counters channels)

Table 1: Tagger T counter input bank TAGT

TAGE:

1	ID	I	1	384	! the address of the hit detector element
2	TDC	I	0	4096	! tdc information (channels)

Table 2: Tagger E counter input bank TAGE

1 Tagger software hits reconstruction. (RECSIS/a1c)

1.1 Overview

The tagger is composed of 61 T counters , which can be divided into 121 T-bins due to the overlaps between two counters. Each scintillator is equipped with two photomultiplicator tubes (PMTs) (left and right) model EMI 9954A, and the signal is treated by Constant Fraction Discriminator (Phillips 715 CFD). These counters are used for the time measurement. Similarly, the 384 physical E counters, used for the energy determination are divided into 767 E-bins, but are only equipped with one PMT. The T counters signal cable lengths were cut in order to obtain a time jitter in the trigger of less than 1 ns.

The Master OR trigger is generated by a T counter which has received a signal above threshold. The same T counter will also give the stop time in its own TDC. This way, T counter that give a trigger are self-timed.

The aim of the tagger analysis is to associate an E with a T counter (i.e gamma energy and time) in the tagger, the relative timing between T counters must be aligned in order to obtain an overall tagger time resolution which allows to discriminate unambiguously between two RF buckets (2 ns apart).

The tagger hit reconstruction package uses as inputs the TAGT BOS bank, containing informations relative to the T counters TDCs (left and right) and the TAGE BOS bank containing informations relative to the E counters. (Tables 1 and 2)

For one electron to be reconstructed as a valid hit in the tagger hodoscope, it has to match the following requirements: Hits must be found in the left and right TDCs of one T-counter, they must match in time with a hit in the TDC of one E counter (This will be called time matching). Those two detectors must furthermore be on an electron trajectory allowed by the tagger magnet optic (which we will call geometric matching).

Then, further processing can be done: Due to the overlap between counters, hits in adjacent counters are reconstructed as a single hit (one electron going through two adjacent E counters, or two adjacent T counters) if their timing is close enough. This procedure will be referred to as re-binning.

First of all, T-counters Re-binning is motivated by normalization purpose, when one electron goes through two adjacent counters, two hits are registered in two different counters that actually correspond to a single tagged photon. In other words, the sum of all raw hits in the tagger hodoscope is greater than the number of real tagged photons. It is necessary to estimate the ratio between those two numbers. It has also been observed, that probably due to electronic cross-talking, the time difference of one T counter relative to the machine RF time shifts when adjacent Ts also fire. This shift has been observed to be as large as 700 ps for certain T counters. The fine time alignment of the T counters relative to machine RF must take into account whether or note T fire alone or in coincidence with adjacent T's. Therefore, re-binning of T counters is used for better time alignment of the T counters. There are as many calibration constants for fine timing alignment to the RF as T bins, i.e 121.

Re-binning of the E counters also doubles the energy resolution of the hodoscope, since it doubles the number of channels within the tagging range.

After reconstruction, some hit configurations can remain ambiguous, for example, there can be hits in time in more than 2 adjacent E or T counters, or there can be several distinct hits in E counters that match a given hit in the T-counters. All those particular configurations must be listed, identified, and flags are raised in a status word in the tagger output bank TAGR.

For all those reasons, the reconstruction proceeds through 6 steps:

- Raw data filtering and time conversion.
- E - T geometric matching.
- E - T time matching.
- E and T rebinning.
- Hits fine timing adjustment to the RF, and CLAS time offset adjustment.
- Hit status flags.

We are going to review in more details each of those steps:

1.2 Raw data filtering and time conversion

1.2.1 Filtering:

TAGT and TAGE banks contain an overhead of useless datas, which includes: for the TAGT bank:

- Single ended (left or right) hits (noise, or low energy background ?)
- Overflows.

Example:

```
TAGT :   25 entries
T  4 : left=  1054 / right =   905
T  5 : left=  1068 / right =   850
T  30 : left=  2361 / right =   2109
T  31 : left=  2348 / right =   2153
T  37 : left=  286 / right =   68
T  39 : left=  4095 / right =   4095
T  52 : left=  2073 / right =   0
T  54 : left=  3225 / right =   0
T  3 : left=  0 / right =   4094
  ...
T  51 : left=  0 / right =   1890
T  53 : left=  0 / right =   3057
T  55 : left=  0 / right =   4094
```

For the E counters, due to the very big slope of their TDC (500ps/channel, 10 times more than the T counters) a lot of hits are out of time, and will never be matched with T counters hits. (the T counters TDC range is 200 ns, the E counters TDC range is 2 μsec).
For example :

```
 TAGE:  32 entries
E  1   :   12294 / E  22  :   968  / E  35  :   158  / E  42  :   594
E  43  :   591   / E  79  :   971  / E  86  :   2310 / E  109 :   520
E  166 :   1349  / E  181 :   2087 / E  182 :   2086 / E  220 :   837
E  221 :   840   / E  223 :   2261 / E  224 :   2261 / E  231 :   218
E  263 :   1588  / E  264 :   1594 / E  273 :   1048 / E  284 :   630
E  292 :   417   / E  296 :   1814 / E  299 :   1941 / E  300 :   1942
E  349 :   1511  / E  357 :   2282 / E  358 :   563  / E  359 :   571
E  360 :   758   / E  363 :   1363 / E  367 :   1908 / E  368 :   1906
```

Useless hits are filtered, using the following procedure:

- TAGE:
  - Only keeps hits for which E_TDC_MIN < TDC value < E_TDC_MAX. E_TDC_MIN and E_TDC_MAX are TCL switches that are set by default to 700, and 1200.
  - Only keeps hits for which $1 < \text{E id} < 384$.
- TAGT:
  - Only keeps hits for which Left and Right TDC values > 1
  - Only keeps hits for which Left and Right TDC values < 4094

1.2.2 Time conversion:

- TAGE:
  - Converts the TDC channels to time (nanoseconds) using the TDC slope in the Map Manager (typically -500 ps/ch)
  - Subtract the base peak positions, which are also in the Map Manager. (typically at -500 ns) Base peak position are position in the TDC of self triggered hits, they will be discussed in the next section on calibration. The slope and peak positions are negative because the E counter TDCs are common stop.
- TAGT:
  - Converts the TDC channels to time (nanoseconds) using TDC slopes that are stored in the Map Manager (typically 50 ps/ch)
  - Subtract the peak positions that are stored in the Map Manager (typically at 50 ns)

1.2.3 Result:

After filtering and time conversion, the above example looks like:

```
:
 T Counters :   5 entries
           Left    -    Right
 T  4 :    -12.3797 -    -11.9220
 T  5 :    -11.8290 -    -11.4260
 T  30 :     56.6795 -     55.9105
 T  31 :     56.5378 -     55.8736
 T  37 :    -46.0129 -    -46.7526
 ..........
 E Counters :   6 entries
 E  22 :    -11.5500
 E  79 :    -12.1500
 E  220 :     57.8500
 E  221 :     56.9000
 E  273 :    -47.2000
 E  360 :    100.4500
```

1.3 E -T geometric matching

Even when a narrow time coincidence is requested, T counters are sometimes in time with E counters that can not be geometrically on a valid tagger optic electron trajectory. This effect can be quite important as shown in figure 2 that corresponds to a normalization run (of G1 running period) that is to say a very low intensity run where one would expect no accidental coincidences between E counters and T counters.

Those hits in time seems to comes from 2 different effects:

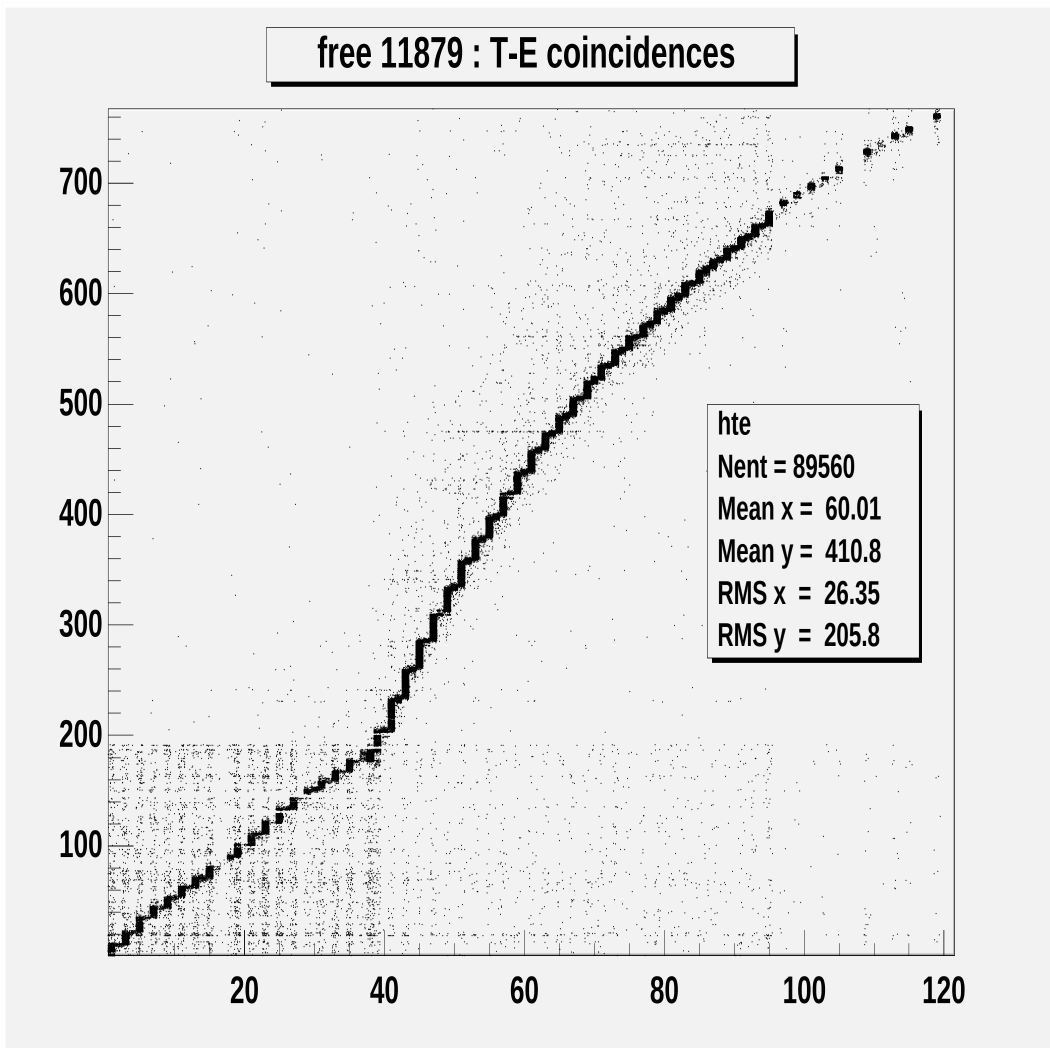

Figure 2: E.T pattern when matched using their timing, Y axis is the E bin, X axis is the T bin

- For the E counters with id > 192, it seems that a tail of hits appears for E counters with higher ID than the ones that should geometrically match a given T counter. This background can be due either to synchrotron radiations, or more probably to electrons which interact in the E counter.
- For the E counters with id < 192, there is an electronic problem, (may be the TDC modules are not properly reset after each event) that produces a great quantity of fake hits in time with the T counters. One can hope that those kind of flaw will disappear in the future.

Furthermore, at higher intensity the rate of accidentals between E counters and T counters will increase. Selecting hits that correspond to geometrically matched detectors, reduces the amount of accidentals.

E - T geometric matching is based upon the file named **tagETcoinc.dat** which stands in the CLAS_PARMS area, and which has been provided by Dan Sober (CUA), based on the optical characteristics of the tagger magnet. (see also clas-note 91-012 *A Guide to the Optics of the Tagged Photon Magnet*, D. I. Sober)

Result E - T spectra after matching is shown in figure 3.

1.4 E - T time matching.

In order to reduce the number of accidentals between E counters and T counters, especially during production runs, one also apply a time coincidence cut on the data: Hits are only kept if the time difference between the E counter and the T counter is less than half the value of the parameter ET_window, which is defined as a tcl switch. At the present time, this switch is set to 20 ns. This time comparison is made after peak position subtraction.

Figure 4 shows the E counters - T counters time correlation.

1.5 E and T rebinning.

Below is an example of the status of the reconstruction after filtering, geometric matching, and time matching:

```
T  4(   -12.3797/   -11.9220)
   <- E  22 (   -11.5500)
T  5(   -11.8290/   -11.4260)
   <- E  22 (   -11.5500)
T  30(    56.6795/    55.9105)
   <- E  220 (    57.8500)
   <- E  221 (    56.9000)
T  31(    56.5378/    55.8736)
   <- E  220 (    57.8500)
   <- E  221 (    56.9000)
T  37(   -46.0129/   -46.7526)
   <- E  273 (   -47.2000)
```

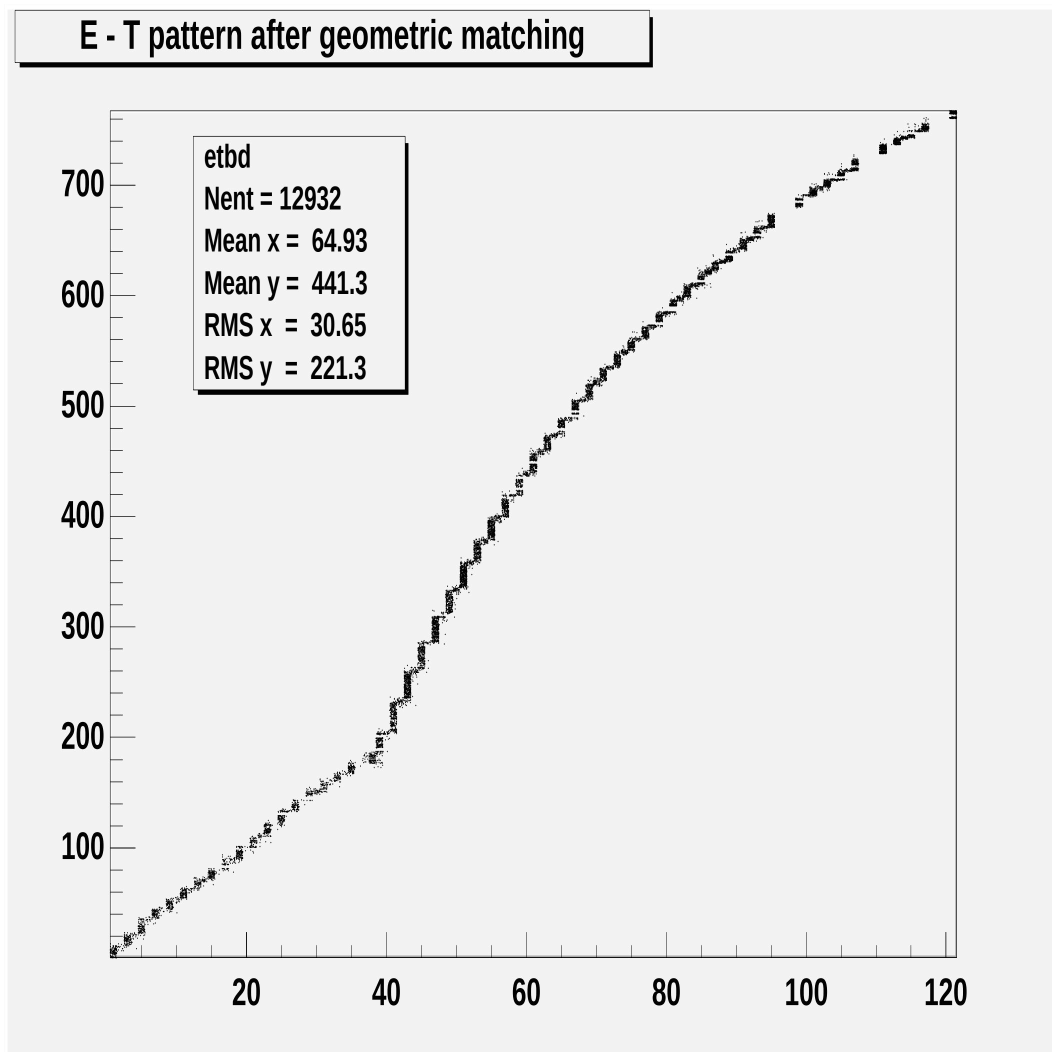

Figure 3: E.T pattern when matched geometrically, Y axis is the E bin, X axis is the T bin

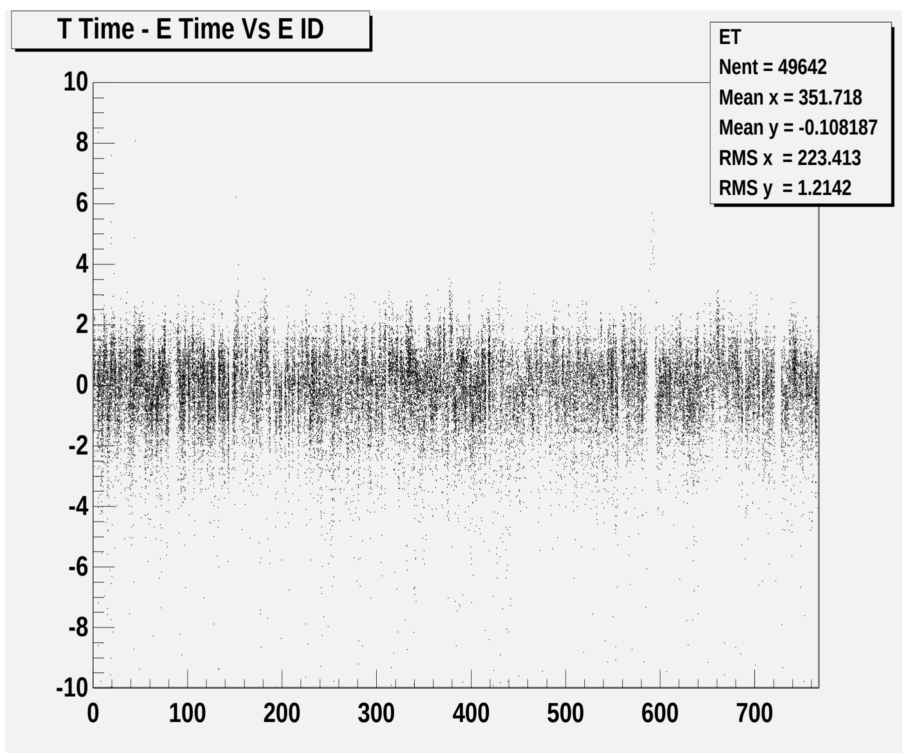

Figure 4: E.T time correlation, Y axis is the time difference in ns, X axis is the E id

Although there are 5 hits in the T counters and 4 hits in the E counters, it is clear that there have been only 3 electrons detected in the hodoscope (one went through T4,T5 and E22, the second through T30,T31,E220,E221, and the last one through T37, and E273).

Rebinning, is done using the following rules:

- TAGE:
  - If two adjacent E counters fire, and their time difference is less than half the value of the parameter ADJ_E_COINC, they are considered as coming from the same electron, and the two hits are gathered into one. At the time of this clas note ADJ_E_COINC is set to 20 ns.
  - If the electron goes through a single E counter, $E\ rebinned\ id = 2 \times E\ id - 1$.
  - If the electron goes through 2 E counters , $E\ rebinned\ id = 2 \times first\ E\ id$
- TAGT:
  - If two adjacent T counters fire, and their time difference is less than half the value of the parameter ADJ_T_COINC, they are considered as coming from the same electron, and the two hits are gathered into one. At the time of this clas note ADJ_T_COINC is set to 10 ns.
  - If the electron goes through a single T counter, $T\ rebinned\ id = 2 \times T\ id - 1$.
  - If the electron goes through 2 T counters, $T\ rebinned\ id = 2 \times first\ T\ id$

At the present time, the parameters ADJ_E_COINC and ADJ_T_COINC are rather large, this choice was made at a time when calibration procedures were not fully developed, the hardware setup not fully optimized, and considering that accidentals between adjacent E's are negligibles.

For G1 running period, with $10^7 e^-/s$ in the whole hodoscope (61 counters), there is roughly $2.10^5 e^-/s$ for each counter. With a window of 10 ns, the probability of an accidental between two T counters is 0.2 %.

For G6 running period this probability is multiplied by 3 (only 20 counters, for the same overall global rate), which leads to a probability of accidentals of 0.6 %. This is not so negligible anymore.

Now that the calibration procedure and the tagger reconstruction package is optimized they should probably be reduced.

In particular the fine T-counters time alignment constants (C_i) of odd T bins (single T counter hits) are used for T - rebinning, and therefore the precision should be better than 2 ns. (The reason why the precision is not equal to the intrinsic tagger time resolution (200 ps) will be explained in next section.)

After rebinning the previous example gives:

```
  1  ERG       F    0.  10.     ! Energy of the photon in GeV
  2  TTAG      F    -20.  200.  ! Time reconstructed in the Tagger
  3  TPHO      F    -20.  200.  ! RF time of the identified nearest bucket
  4  STAT      I    0   4096    ! Status
  5  T_id      I    1   121     ! T bin Id
  6  E_id      I    1   767     ! E bin Id
```

Table 3: TAGR: Tagger main output bank definition

```
T bin 8(   -12.3797/   -11.9220)(   -11.8290/   -11.4260)
    <- E bin  43 (   -11.5500)
T bin 60(    56.6795/    55.9105)(    56.5378/    55.8736)
    <- E bin  440 (    57.8500) (    56.9000)
T bin 73(   -46.0129/   -46.7526)
    <- E bin  545 (   -47.2000)
```

1.6 Final timing.

Once the rebinning is done, one can seek for the best timing alignment between T counters, in order to retrieve on an event by event basis the nearest beam bucket time. This is done through 121 constants (the so called C_i constants) that are at this step of the reconstruction subtracted to the mean left-right time value of the T counters (the First of the two T counters in case of an even T bin). An overall tagger to CLAS (TOF) offset constant is then added to this time to finally obtain the tagger output time ("TTAG" entry in the TAGR bank). The tagger to CLAS time offset is also added to the timing of the nearest beam bucket to give the entry "TPHO" in the TAGR bank. (table 3)

"TPHO" is the entry that should be used, since it is the most precise timing that can possibly be obtained (RF machine timing). "TTAG" is given as a control entry, to verify that the tagger to RF alignment looks good.

Details on the meaning of the C_i constants and how to determine them, will be discussed in the next section on calibration, in section 2.4.

Figure 5, shows the overall time difference between the tagger reconstructed time and the RF time for a production run, the typical sigmas are of the order of 150 ps, which is quite enough to unambiguously identify RF buckets.

1.7 Hit status flags.

The status entry of the TAGT bank (table 3) is an integer value that is the sum of various flags:

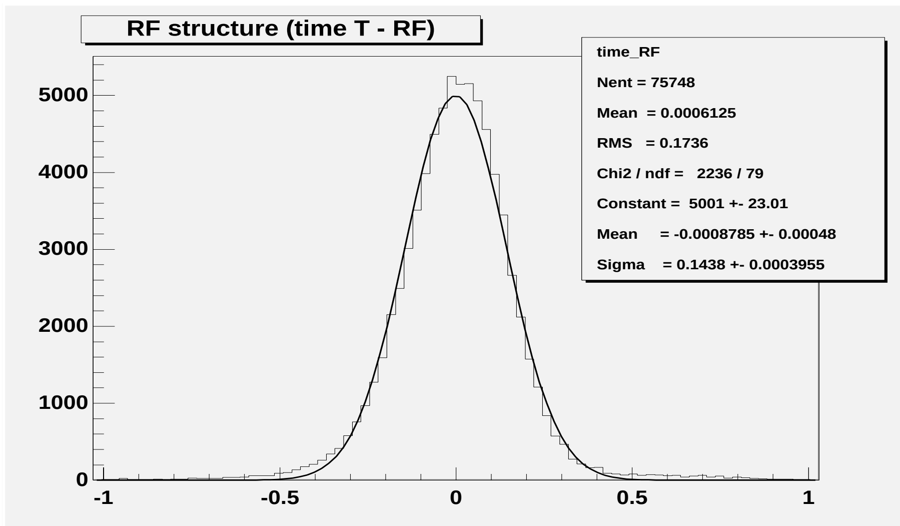

Figure 5: Tagger time resolution for a production run, using the RF machine time as reference. (X axis is nanoseconds)

1	Left T present
2	Right T present
4	E present
8	Multiple Ts
16	Adjacent T bins (i.e more than 2 adjacent T counters fire in time)
32	Adjacent E bins (i.e, more than 2 adjacent E counters fire in time, and also multiple hits in the SAME E counter (because of the multiple hit TDCs))
64	Multiple Es (That is to say several NON-ADJACENT E BINS in coincidence with ONE T bin)
256	Overflow (when the number of hits reconstructed is higher than the size of working arrays)

Flags 1, 2 and 4 indicates that the left T counter TDC, the right T counter TDC, and the E counter TDC were used to reconstruct the hit. At the present time left and right T counters are always required to reconstruct a tagger hit. On the other hand there is a tcl switch (Tagger_NO_E) that allows the tagger software to reconstruct hits even when no E counters hits are present (in that case, E-T geometric matching and E-T time matching steps are skipped), the purpose of this switch is to study E counter efficiencies and E-T hardware coincidence electronic. However for normal production data processing, E counters hits will always be present, so that ***all hits*** during normal processing will got at least those three flags. For example, if none of the other flags are present, the status will be $(1 + 2 + 3) = 7$, ***status=7 means that one unambiguous hit has been reconstructed in the tagger hodoscope***.

If more than one unambiguous hits are reconstructed in the tagger (figure 6 a), then the flag 8 will be present, thus giving for each of those hits 1 + 2 + 3 + 8 = 15. ***status=15 means that several unambiguous hit has been reconstructed in the tagger hodoscope***. The determination of which one (if any) correspond to the photon that produced the hadronic reaction is done by time comparison with the Start Counter, and further kinematic analysis.

Those two status values are those that analysis software should retain, other status values, haven't been fully investigated yet, and raise serious normalization problems

Other configurations are:

- ***More than 2 adjacents E counters fire in time*** (figures 6-b and 6-c). This configuration can occur for several reasons: scattering of the electron off the 2 first scintillators to the third one, secondary photon in the third scintillator, light leak from one scintillator to the next one. In all those cases though, it is likely that those multiple adjacent hits come from one single electron (and one single tagged photon going to CLAS).
  3 (or more) adjacents Es cannot fit into the rebinning scheme. In fact, all what the rebinning routine does is to gather the two first scintillators into one binned hit, and gathered additional adjacent Es hits in ***additionals*** binned hit, which is inappropriate.
  Therefore the ***flag 32 (adjacent Es flag)*** is set ***for the additional binned hits (not the first one)*** The reason for this is that one can consider that the first binned hit is a valid hit, that possibly corresponds to a tagged photon, but all adjacent additional hits are just artifacts of the rebinning scheme and should be ignored.

- Two ***non-adjacent E hits are reconstructed that match geometrically and are in time with one T counter hit*** (figure 6-d).
  explanations for this situations could be:
  - One electron in one E counters, electronic noise in the other.
  - Accidental between two electrons that go through the same T (since the T TDC are common start, only one hit can be recorded in the T counters).
  - One electron, and one secondary photon emitted by this electron.
  - Multiple scattering of one electron when middle E counter is dead.
  - Electronic flaw that produces ***in times*** hits (previous hits not properly erase ?). This effect is sometime far from been negligible as shown in figure 2 (normalization run).

  In that case ***both hits are flagged 64*** which means that we got two different energy candidates for one timed hit. Since those E counters must geometrically match one T counter, their energies are not far away, so analysis with wide energy divisions could possibly use them, but it is not recommended.

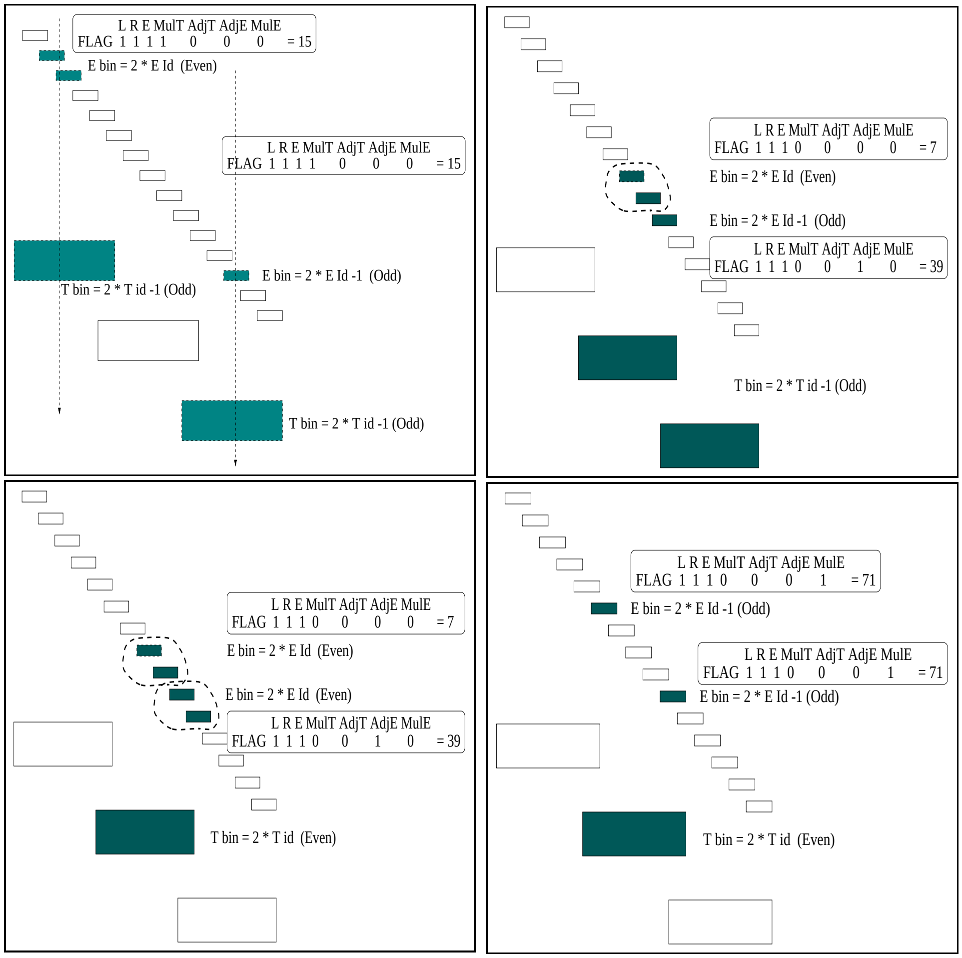

Figure 6: *Various hit configurations.*

- ***More than 2 adjacents T counters fire in time***
  This is mostly the case for T counters 18, 19 and 20. In the tagger design, in order to fit Rad-phi experiment requirements, the first 19 T-counters are significantly smaller than the rest. T counter 20 almost completely overlap T counter 19, which means that almost all electrons that go through T19 also go through T20 (which explains why bin 38 is more populated than bin 39, as opposed to all other T bins, where odd bins are more populated than even bins.) Therefore an electron going through T18 and T19 is likely to also go through T20. The rebinning scheme will reconstruct two hits in that case (one in bin 36 and one in bin 39), ***the second one is however an artifact, it is flagged 16, and should be ignored.***

TCl switch	definition	default value
Tagger_TTwin	Adjacent T time coincidence window in ns	10 ns
Tagger_EEwin	Adjacent E time coincidence window in ns	20 ns
Tagger_ETwin	E/ T time coincidence window in ns	30 ns
Tagger_ETDCmin	E TDC minimum accepted value (in channel)	700
Tagger_ETDCmax	E TDC maximum accepted value (in channel)	1200
Tagger_DSDwin	T- DSD time coincidence window (in ns)	15 ns
Tagger_PSOSwin	PS paddles time coincidence window (in ns)	30 ns
Tagger_NO_E	do no require E counters for analysis (if equal to 1)	0
Tagger_PC_TDC	requires hit in TDC to process PC (if equal to 1)	1
Tagger_PS_TDC	requires hit in TDC to process PS (if equal to 1)	1
Tagger_TAC_TDC	requires hit in TDC to process TAC (if equal to 1)	1

Table 4: TCl processing switches summary

1.7.1 Conclusion: Flags in the status entry of the TAGR bank and normalization.

At the present time, only hits with status 7 or 15 should be retained for analysis.

Hits with Flags 16 and 64 will require special normalization care, since they are ignored in the analysis, but produce hits in the scalers. They should probably be counted during analysis as a function of the T counters, and corrections should be applied for each T counter, of the form:

$$1 + \frac{N_T^{rejected}}{N_T^{accepted}}$$

Additional note: For G1 running period, the prescale scheme was affecting in an different way the TDC (recorded events) and the scalers, and it has been shown that normalization was extremely hard to estimate when there are several hits within a given time coincidence window with the Start Counter. In this particular case, one should only analyze events for which there are only one tagger hit within this window, and then correct the number of reconstructed events by the ratio:

$$1 + \frac{N_{several\ tagger\ hits}}{N_{1\ tagger\ hit}}$$

1.8 TCl switches summary

In table 4 is given a summary of the switches used in RECSIS processing, together with their defaults values.

```
1  IDT      I    0     121    ! T id
2  TIMEL    F    -999999.9 999999.9 ! time information (Left counters channels)
3  TIMER    F    -999999.9 999999.9 ! time information (Right counters channels)
4  IDE      I    1     767    ! E id
5  TIMEE    F    -999999.9 999999.9 ! time information (E counters)
6  TIMEMEAN F    -999999.9 999999.9 ! time information (left/right mean value | after alignment)
7  TRF      F    -999999.9 999999.9 ! time information (mean val - RF | after alignment)
8  NEXTTIME F    -999999.9 999999.9 ! time difference with the next T when in coincidence | after
```

Table 5: Tagger intermediate result bank TAGI

2 Calibration methods

2.1 Introduction

The issue is to associate calibration variables to a sequence of runs that behave the same way, usually during the same time period, in order to process the tagger data in the complete analysis procedure. These constants allow to substitute the tagger time with the proper RF time. This is a crucial part of the analysis since machine electron bunches are separated by 2.004 ns. An error of one RF bunch on the tagger calibration might propagate to the particle ID and compromise the ability to distinguish pions from Kaons (2.004 ns delays on the tagger time can introduces a variation of several hundreds of MeV of particle reconstructed mass).

Calibration of the tagger were previously done using PAW histograms produced by the *ana* package in RECSIS. This required to run several times RECSIS, by hand at each step of the calibration, feeding it with the temporary constants. A new programs has been written to automate this procedure, that will be described in next section. Both ways the method is to run RECSIS several times adjusting each time a new set of constant, taking into account what the previous calibrations were. Part of the informations are retrieve from the tagger main output bank "TAGR" (table 3) described in the previous section, the rest is retrieved from the tagger intermediate result bank "TAGI" (Table 5)

Down stream devices informations are taken from the "TACO", "PCO" and "PSO" output banks. (Table 6)

There are 4 sets of constants used in the tagger software reconstruction:

- TDC slopes. (2×61 constants for left and right T counters TDCs, 1 constant for E counters TDC's)
- Base peak position. (2×61 constants for left and right T counters, 384 constants for E counters, 1 constant for the TAC, 1 constant for the PC, 8 constants for the PS.)
- C_i RF fine adjustment constants. (121 constants for each T - bin)
- Overall tagger time offset with respect to CLAS detectors (TOF, Start counter,..) (1 constant)

```
TACO:

  1  ID       I    1          2         ! ID  1= TAC, 2 = USLG
  2  TIME     F    -999999.9  999999.9  !  TDC time
  3  ELT      F    -999999.9  999999.9  ! energy deposit TAC = left top
  4  ERT      F    -999999.9  999999.9  ! energy deposit TAC = right top
  5  ELB      F    -999999.9  999999.9  ! energy deposit TAC = left bottom
  6  ERB      F    -999999.9  999999.9  ! energy deposit TAC = right bottom
  7  ESUM     F    -999999.9  999999.9  ! energy deposit TAC = sum1
  8  ESUM2    F    -999999.9  999999.9  ! energy deposit TAC = sum2
  9  ESUM3    F    -999999.9  999999.9  ! energy deposit TAC = sum3
 10  TID      I    0     121            ! T id of the corresponding T


PSO:

  1  ID       I    1     8             ! Counter Id ()
  2  TIME     F    -999999.9  999999.9 !  TDC time
  3  ENER     F    -999999.9  999999.9  ! energy deposit
  4  TID      I    0     121            ! T id of the corresponding T


PCO:

  1  TIME     F    -999999.9  999999.9 !  TDC time
  2  ELT      F    -999999.9  999999.9  ! energy deposit PC = left top
  3  ERB      F    -999999.9  999999.9  ! energy deposit PC = right bottom
  4  ELB      F    -999999.9  999999.9  ! energy deposit PC = left bottom
  5  ERT      F    -999999.9  999999.9  ! energy deposit PC = right top
  6  EMAIN    F    -999999.9  999999.9  ! energy deposit PC = MAIN
  7  EVETO    F    -999999.9  999999.9  ! energy deposit PC = veto
  8  TID      I    0     121            ! T id of the corresponding T
```

Table 6: Down stream devices output banks.

All those constants depend on each others, and must be calibrated in the following order:

1. The TDC slopes.
2. The Base peak positions.
3. C_i's (relative alignment of T's)
4. alignment to CLAS.

TDC slopes is however easier to understand after the timing alignment issues have been reviewed, therefore we will discuss first timing alignment issues, and then TDC slopes calibrations.

2.2 Timing alignments (base peak and C_i's) overview

Because the tagger is working in a self triggered mode, and because it is composed of several channels (the various T-counters) which produce triggers that are not aligned in time with respect to the physical event (production of the bremsstrahlung photon, hadronic reaction in the target) with the required precision (less than 200 ps), calibration of the tagger requires some care. In figure 7 is depicted the 3 ***independent*** time delays in the electronic setup that have to be taken into account for the tagger timing reconstruction:

- The detection arm delay D_i is the delay when a tagged photon is produced in the tagger radiator for the corresponding logical signal to be output from the Constant fraction discriminator.
- The trigger arm delay T_i is the delay for the discriminator output signals to trigger the acquisition.
- The TDC arm delay TDC_i is the delay for the discriminator output signals to stop the TDC.

Those ***absolute*** delays will never be measured, but the ***relative*** delays, i.e differences in those delays from one T counter to the others are the quantities we will have access to. In the following discussion, when we will talk about one of these delays, we will therefore implicitly talk about ***relative delays from one T counter to the other***.

Ultimately, when the timing of all T counters will be calibrated with respect to each other, one last relative timing adjustment will have to be done between the tagger/RF reconstructed time, and CLAS detectors reconstructed time.

Before E-T time matching, E rebinning and T rebinning, the base peak position are subtracted to the reconstructed T-counters and E counters time. We will look in more detail to what delays refer the base peak, (*Base peak calibration*) and we will show that this method is not good enough to get a precise time calibration (for T counters timing), but can be used, without problems, for less accurate processing (E rebinning, E-T binning).

In a second part, we will show how to get the precise timing alignment constants (*C_i's*) for the T counters.

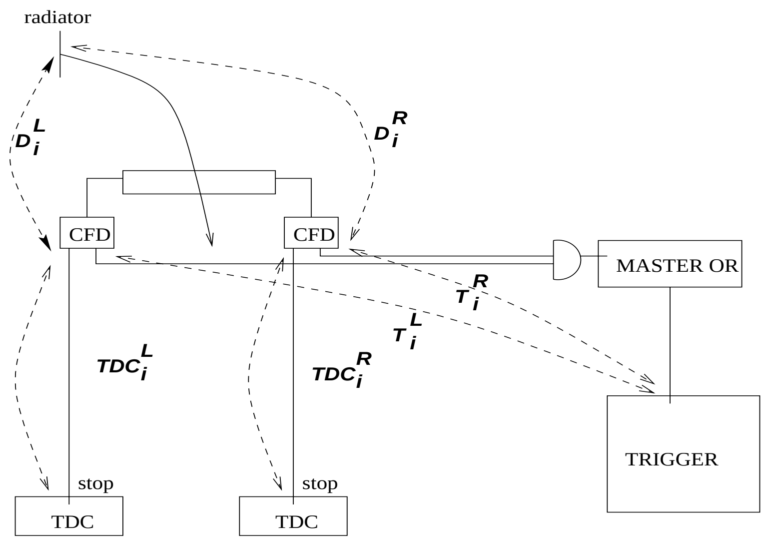

Figure 7: Schematics of electronic delays.

2.3 Base peak calibration

2.3.1 T counters

When looking at a raw T-counter TDC spectrum a peak can clearly be seen. At higher intensity, on can also see a background continuum. The peak corresponds to self triggered hits, that is to say when the T counter we are looking at produced the trigger. The continuum comes from hits when other T counters give the trigger. If there were only one PMT for each T counter, the peak would be a Dirac, yet because T counters are double ended, those peaks most of the time have a width corresponding to the transversal dispersion of the hit along the T counter scintillator when the actual trigger timing is given by the other end of the scintillator. (figure 8). For this reason, distributions are not gaussians, but most of the time half gaussians that stop abruptly to dirac, when the trigger timing flips from one side of the PMT to the other.

One can however fit the left and right peaks. By doing this the delays that are compared between T's are:

$$< Peak(L)_i >=< TDC(L)_i > - < T(L/R)_i >$$

$$< Peak(R)_i >=< TDC(R)_i > - < T(R/L)_i >$$

$< peak(L)_i >, < peak(R)_i >$ are the calibration constant that will be written in the Map Manager.

$< T(L/R)_i >$ and $< T(R/L)_i >$ are not well defined constants, they take into account the fact that sometimes the trigger timing is given by the left arm, and sometimes is given by the right arm, this is not a major issue though, because for the rest of the discussion, the quantity that will always be used is the mean value of the left and right time (t_i^L and t_i^R) after subtraction of the base peaks, i.e :

$$
\begin{aligned}
t_i^{mean} &= \frac{(t_i^L - Peak(L)_i) + (t_i^R - Peak(R)_i)}{2} \\
t_i^{mean} &= \frac{t_i^L + t_i^R}{2} - \frac{<TDC(L)_i> + <TDC(R)_i>}{2} + \frac{<T(L/R)_i> + <T(R/L)_i>}{2}
\end{aligned}
$$

Therefore for the rest of this discussion we will define, in order to simplify the notations:

$$TDC_i = \frac{TDC(L)_i + TDC(R)_i}{2}$$

$$T_i = \frac{T(L/R)_i + T(R/L)_i}{2}$$

$$< TDC_i >= \frac{< TDC(L)_i > + < TDC(R)_i >}{2}$$

and

$$< T_i >= \frac{< T(L/R)_i > + < T(R/L)_i >}{2}$$

Those constants are not well defined constants, but there are *constants* that can be measured with reasonable precision (better than 1 nanosecond), which is good enough for the purpose of E-T matching and E rebinning. For more accurate time measurements, (T counters and RF) we will see that those *fixed* quantities will vanish.

As expected for a self triggered hit, with those definitions:

$$t_i^{mean} = TDC_i - T_i - (< TDC_i > - < T_i >)$$

is close to zero.

When triggered by another T counter l:

$$t_i^{mean} = (D_i + TDC_i) - (D_l + T_L) - (< TDC_i > - < T_i >)$$

When comparing the timing of two T counters i and j fired by the same electron when the trigger is given by a second electrons going through a T counter l:

$$
\begin{aligned}
t_i^{mean} - t_j^{mean} = &\quad ((D_i + TDC_i) - (D_l + T_L) - (< TDC_i > - < T_i >)) \\
&- ((D_j + TDC_j) - (D_l + T_L) - (< TDC_j > - < T_j >)) \\
t_i^{mean} - t_j^{mean} = &\quad (TDC_i - < TDC_i > + < TDC_j > - TDC_j) \\
&+ (D_i + < T_i >) - (D_j + < T_j >)
\end{aligned}
$$

One sees that this quantity will be close to zero only it the quantities $D_i + T_i$ are close between T counters, in other words, if the T counters signals alignment is good at the Trigger level.

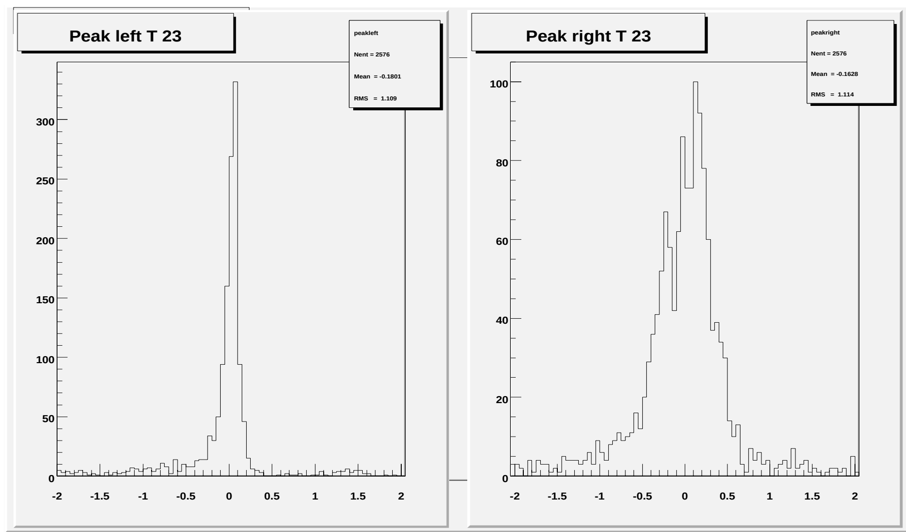

Figure 8: Left and right peaks for one given T, X axis is in ns, for this T most of the time the trigger timing was given by the left arm, thus giving a narrow peak on the left, and a wider peak on the right (due to transversal dispersion of the electrons hits)

If one only subtract the base peak to compare T counter timing, the precision obtained cannot be better than the hardware alignment of the T counters signal at the trigger level. Yet, this misalignment of the signals will be taken into account through the constant C_i which will be described in the next subsection. Those constants will provide a timing alignment better than one nanosecond independently of the quality of the alignment of the signal at the trigger level.

2.3.2 E counters

In the E counter TDC spectra, the peak also comes from self triggered electrons, and the continuum from hits that occur in this E-counters while the trigger was produced by another electron. The peak width is larger in the E counter TDC spectra because the trigger timing is given by the corresponding T counter, the width of the the E counter spectra is the sum of the jitter in the trigger due to transversal dispersion of hits along the T counter, the E counter intrinsic resolution, and also, when the coincidence for one given E is done with different T counters, in which case the trigger timing can be different. (figure 9).

For one E counter j in coincidence with the T counter i, self triggered:

$$t_j^E = (D_j^E + TDC_j^E) - (D_i + T_i)$$

As we just said, if the trigger is given by a different T counter $D_i + T_i$ will be different. When one fits the self triggered peak of the row E counter spectra, one actually fits an average of trigger delays coming from various T counters that can possibly match this E counter, weighted by the probability for this match to occur:

$$Peak_j^E =< t_j^E >= (< D_j^E > + < TDC_j^E >) - < w_{i.j} \dot{(} < D_i > + < T_i >) >_{i.j}$$

This formula doesn't need to be looked into much details, the only important thing here is that ***the better the alignment of T counters trigger signal is at the trigger level, that is to say the closer*** $< D_i > + < T_i >$ ***are for different i, the narrower the E counter peaks will be***

For other hits (not self-triggered), when the trigger is given by a T counter l, one has:

$$t_j^E = (D_j^E + TDC_j^E) - (D_l + T_l)$$

therefore, using all the previous expressions:

$$\begin{aligned} t_j^E - Peak_j^E = & \quad \big((D_j^E + TDC_j^E) - (D_l + T_l)\big) \\ & - \Big((< D_j^E > + < TDC_j^E >) - < w_{i.j} \dot{(} < D_i > + < T_i >) >_{i.j}\Big) \\ t_j^E - Peak_j^E = & \quad (D_j^E + TDC_j^E) - (< D_j^E > + < TDC_j^E >) + \\ & < w_{i.j} \dot{(} < D_i > + < T_i >) >_{i.j} - (D_l + T_l) \end{aligned}$$

One sees that systematic errors in comparing the timing of two E counters for E rebinning will come from the terms $D_i + T_i$, i.e uncertainties in the alignment of the T counters signal at the trigger level.

2.3.3 E - T time matching

$$\begin{aligned} (t_j^E - Peak_j^E) - t_k^{mean} = & \quad \epsilon_j + < w_{i.j} \dot{(} < D_i > + < T_i >) >_{i.j}) - (D_l + T_l) \\ & - ((D_k + TDC_k) - (D_l + T_L) - (< TDC_k > - < T_k >)) \\ (t_j^E - Peak_j^E) - t_k^{mean} = & \quad \epsilon_j + (TDC_i - < TDC_i >) \\ & + \Big(< w_{i.j} \dot{(} < D_i > + < T_i >) >_{i.j}) - (D_k + < T_k >)\Big) \end{aligned}$$

The dominant error factor is the differences between $D_i + T_i$***, i.e misalignment of T counters at the trigger level. Contrary to the T counters, no fine timing alignment is done at the present time for E counters, so alignment of the T counters at the trigger is a limitation to the precision of the E - T time matching.***

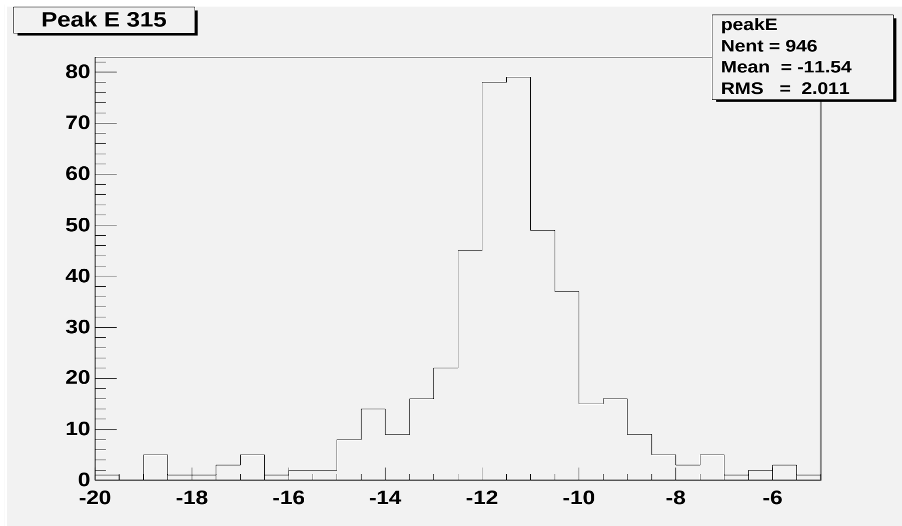

Figure 9: E counter peak. X Axis is in ns, the size of the bins corresponds to the resolution of the TDC (500ps/ch).

2.3.4 conclusion on peak position calibration.

Besides the intrinsic resolution of the E counters and T counters, the main errors in time matching between adjacent E's, and E's and T's, are misalignment of the T counters signals at the trigger level. This is the reason why coincidences windows between E's and T's have been originally set to 20 ns in the reconstruction packages

Subtracting the base peak to make timing coincidence is not correct in its principle, we will see in the next subsection which quantities really have to be calibrated to get ***exact*** timing comparison. This exact procedure is used to align the T counters, so that those constants (the C_i's) can be use for T counters rebinning, thus providing a precision better than one nanosecond in the T alignment. This precision cannot be brought down to the overall tagger resolution of 150 ps, because it has been seen that timing of a single T counter can shift up to 700 ps depending whether or not the adjacent T also fired. Fine T counter timing alignment cannot be better than those erratic shifts ***before*** the rebinning, and thus can't be use for the rebinning itself.

This exact procedure of alignment hasn't been used for the E counters, therefore, during E-rebinning and E-T time matching, misalignment of the T counters signal at the trigger level brings systematic errors in the time comparison. Yet, the hardware T counters signal alignment has been brought within one nanosecond, which is better than the intrinsic E counters time resolution, so changing the calibration procedure would not really improve E rebinning and E-T matching.

2.4 C_i calibration

After the previous hit reconstruction has been done, one now wants to use the reconstructed tagger time to identify which RF bucket produced the reconstructed hit.

Since there are RF buckets every 2.004 nanoseconds, to avoid edge effects, the RF machine signal is prescaled by 40. This signal is then split, one of the two resulting signals goes through a delay of 40 ns, and finally both are sent to TDCs. In short, the two TDCs receive 80.16 ns periodic signal with a phase shift of roughly 40 ns. The RF time is reconstructed (channel to time conversion, choice of RF1 or RF2, phase offset between RF1 and RF2) in the CL01 package, which provides at the end ***one RF time relative to the trigger,*** t_{RF}. Yet the tagger hits are spread over a range of 200 ns, even the coincidence window between the tagger and the start counter is 15 ns, all those timing informations spread over several beam buckets, and nothing tells which one corresponds to the RF time given by the CL01 package.

In fact the RF time t_{RF} should be understood as the time of an undetermined RF bucket, or as "several buckets went through the radiator during the acquisition period at times $t_k = t_{RF} + k \times 2.004ns$ relative to the trigger". ***the RF time*** t_{RF} ***is not an absolute time versus the trigger, but gives information on the phase shift between the RF machine time periodic signal (with period 2.004 ns) and the trigger. In other words, there is a "bucket offset"*** $k(event)$***, between the timing of the tagged photon*** $tpho$ ***and the timing of the RF*** t_{RF}***, which is different for each event.***

$$t_{RF} = t_{pho} + k_{event} \times 2.004$$

The purpose of this calibration is to align the T counters with such a precision that the value of k_{event} can be determined on an event by event basis. This will allow then to deduce the value we are looking for (t_{pho}) from the value of t_{RF}.

2.4.1 Reference detector

Instead of looking at all the electron beam buckets that reached the tagger radiator, one can look directly at the tagged photon, this can be done for example using the Total Absorption Counter, that detects those photons, or the start counter, that detects hadronic reaction produced by those photons. Any detector whose timing is correlated to the production of the tagged photon can be used. Let's call it "reference detector", let's call D_{ref} the electronic delay between the time the tagged photon is produced in the radiator, and the time the corresponding electronic signal(s) stops the reference detector TDC(s). If the trigger is given by T counter l, the time recorded in the reference detector is:

$$t_{ref} = D_{ref} - (D_l + T_l)$$

Let's assume this photon as been tagged by the T counter i, the time recorded in this T counter is:

$$t_i = (D_i + TDC_i) - (D_l + T_l)$$

But for the purpose of E-T matching and E and T rebinning this time as already been corrected by the base peak value, to give what we called t^{mean}:

$$t_i^{mean} = (D_i + TDC_i) - (D_l + T_l) - Peak_i$$
$$t_i^{mean} = (D_i + TDC_i) - (D_l + T_l) - (< TDC_i > - < T_i >)$$
$$t_i^{mean} = (D_i + < T_i >) + (TDC_i - < TDC_i >) - (D_l + T_l)$$

the difference between the reference detector time and t^{mean} is thus:

$$t_i^{mean} - t_{ref} = (D_i + < T_i >) + (TDC_i - < TDC_i >) - D_{ref}$$

$(TDC_i - < TDC_i >)$ is a jitter centered on zero (even probably a dirac). The only systematic error that prevents us from having this time independent of which T we are looking at is the misalignment of the T counters signal at the level of the trigger : $(D_i + T_i)$. But we just found a way to measure it : If one fits this quantity $t_i^{mean} - t_{ref}$ for a given T counter i, one finds a constant we can called C_i^{ref} which measures this misalignment:

$$C_i^{ref} = < t_i^{mean} - t_{ref} >$$
$$C_i^{ref} = (< D_i > + < T_i >) - < D_{ref} >$$

If the jitter in the reference detector ($D_{ref} - < D_{ref} >$) is large it will affect the precision of the measurement of C_i^{ref}. Figure 10 shows the distribution $t_i^{mean} - t_{ref}$ that allows to determine C_i^{Ref} for one given T counter.

2.4.2 RF

One "reference" signal has a much more better resolution (80 ps): the RF signal itself, as shown in figure 11. The problem is, as we pointed out before, that there is an uncertainty concerning what is the "bucket offset" between the bucket corresponding to the RF time and the bucket that corresponds to the tagged photon. Which can be written :

$$t_{RF} = t_{bucket} + k(event) \times 2.004 - (D_l + T_l)$$

t_{bucket} is the time of the bucket that produced the photon. k(event) is different ***for each event***. Therefore the only constants that can be measured are the overall RF phase shifts with the T counters:

$$C_i^{RF} = < (t_i^{mean} - t_{RF} + 1.002) mod 2.004 > -1.002$$
$$C_i^{RF} = (< D_i > + < T_i > - < t_{bucket} > +1.002) mod 2.004 - 1.002$$

(The +1.002 -1.002 trick is just to get distributions centered on zero.)
This expression can also be written:

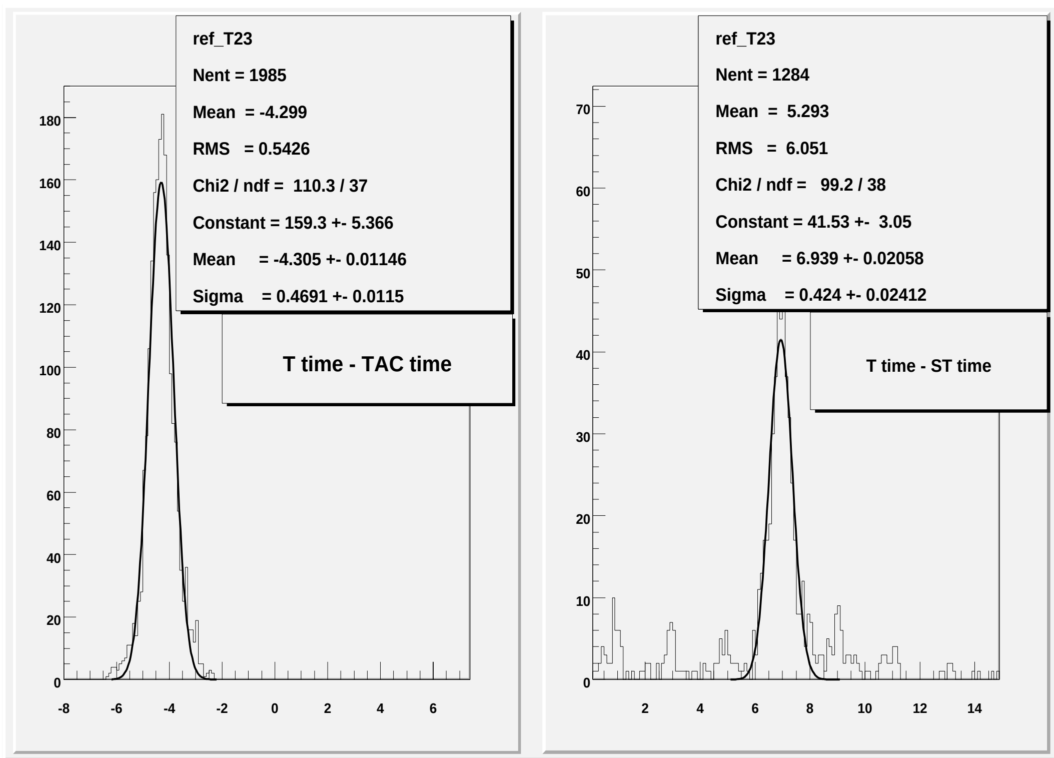

Figure 10: C_i^{ref}: T23 time - Ref time, X axis is in ns, with Ref being the TAC for a normalisation run, and the start counter for a production run. The resolution of the TAC, and the ST is not good enough to get a perfect C_i^{ref}

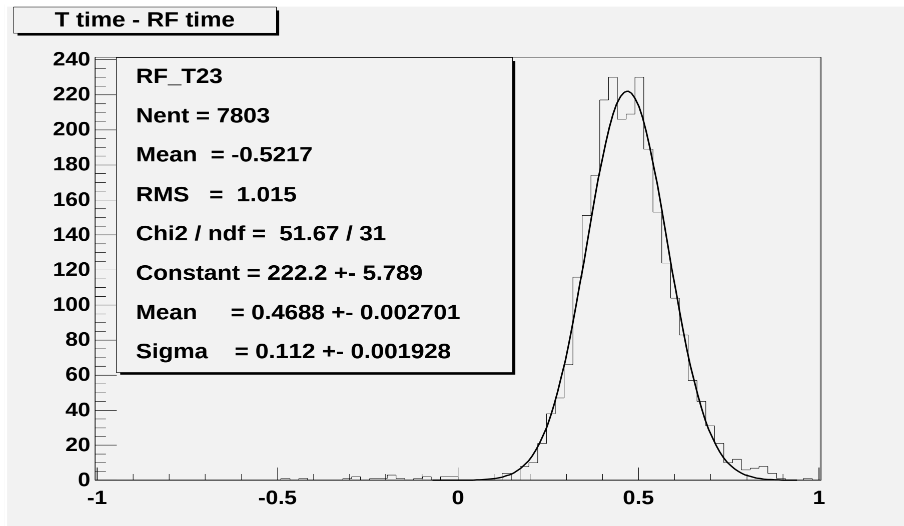

Figure 11: C_i^{RF}:(T23 time - RF time) mod 2.004 for T 23, X axis is in ns, the resolution of the RF allows to get a precise fit.

$$C_i^{RF} = < D_i > + < T_i > - < t_{bucket} > + k_i \times 2.004$$

k_i is an integer value ***than can be different for different T's***. It is the translation that has to be applied to the value of $< D_i > + < T_i > - < t_{bucket} >$ in order to have: $-1.002 < C_i^{RF} < 1.002$ Since the RF jitter is very small, this measurement is very precise, but it doesn't give us the value of k_i. Figure 11 shows the distribution $(T_i^{mean} - t_{RF} + 1.002) mod 2.004 - 1.002$ used to determine C_i^{RF} for a given T.

We can now compare the two constants : C_i^{ref} and C_i^{RF}:

$$C_i^{ref} - C_i^{RF} = < D_{ref} > + < t_{bucket} > - k_i \times 2.004$$

There are all equal to a constant ***independent of the T counter*** plus k_i times 2.004, this allows us to determine k_i and thus to determine the exact C_i's:

$$C_i = < D_i > + < T_i > - < t_{bucket} >$$

$$C_i = C_i^{RF} - k_i \times 2.004$$

with k_i determined by comparing C_i^{RF} and C_i^{ref}

Figures 12 and 13 shows the distribution $(C_i^{ref} - C_i^{RF}) mod 2.004$ which allows to determine the mean constant $<< D_{ref} > - < t_{bucket} >>$ which is used to determine k_i for individuals T bins.

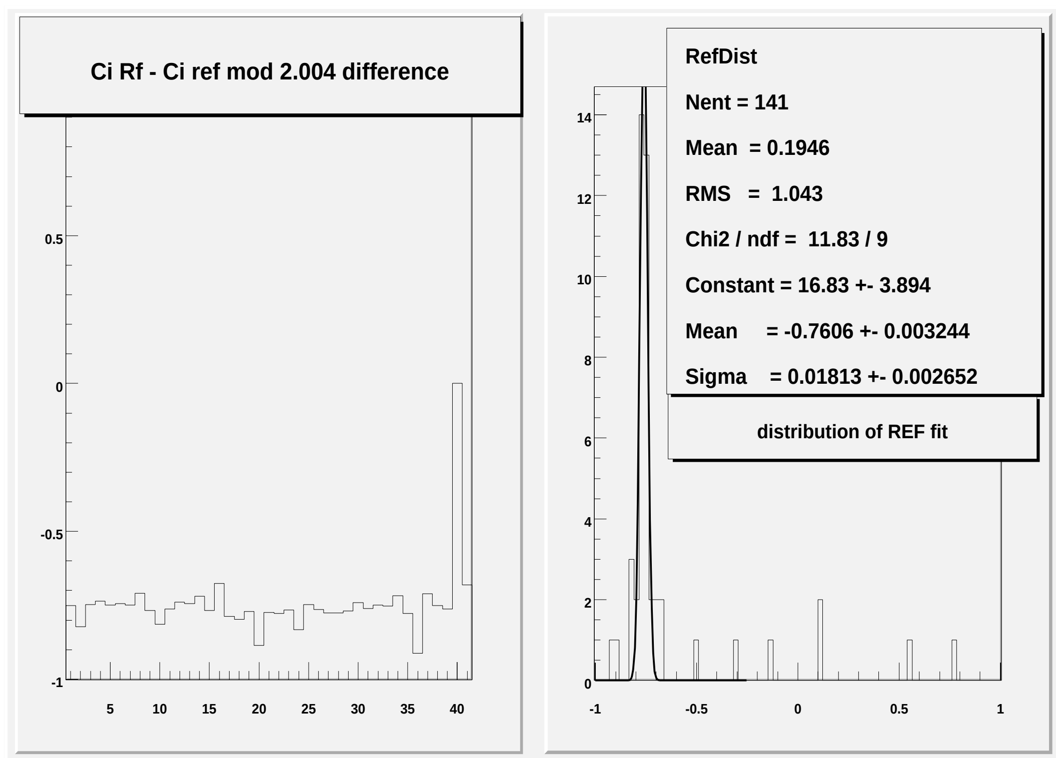

Figure 12: Distribution $(C_i^{ref} - C_i^{RF}) mod 2.004$ (Y axis of left plot, in ns) versus T id (X axis of left plot), right plot is projection on the time axis. shows that the measurement of C_i^{ref} are precise enough to pick the correct bucket (k_i) for each individual T counter. This is using the TAC for a normalisation run of G6 running period.

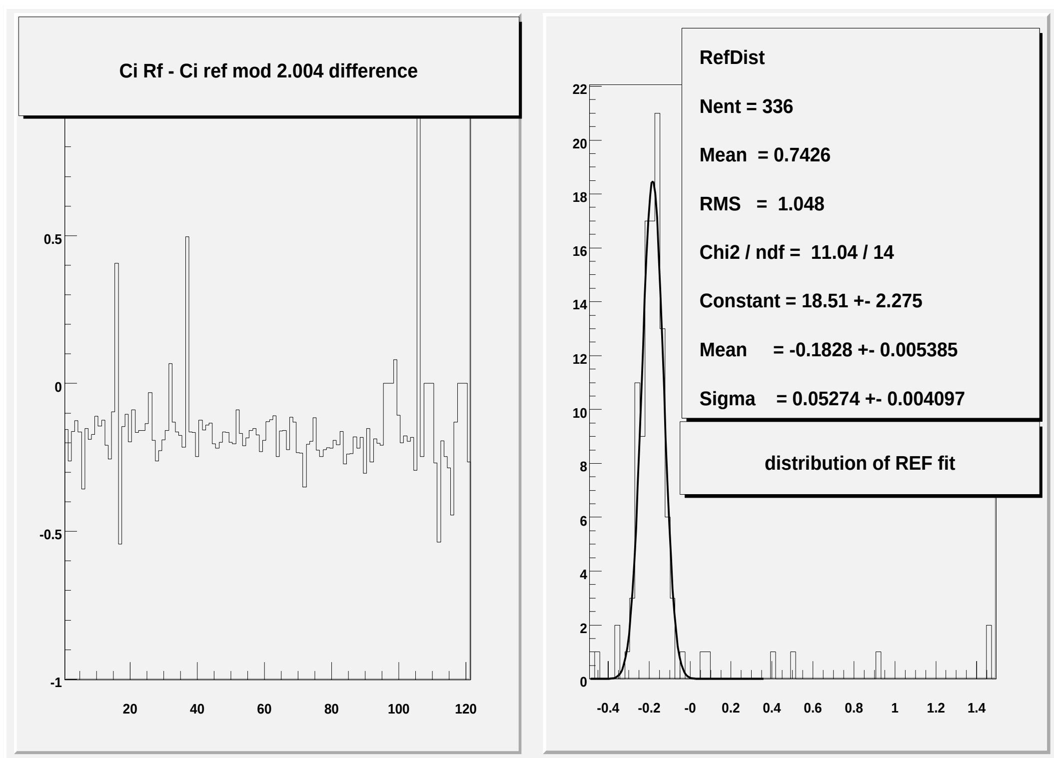

Figure 13: Distribution $(C_i^{ref} - C_i^{RF}) mod 2.004$ (Y axis of left plot, in ns) versus T id (X axis of left plot), right plot is projection on the time axis. shows that the measurement of C_i^{ref} are precise enough to pick the correct bucket (k_i) for each individual T counter. This is using the Start counter for a production run of G6 running period

2.4.3 conclusion on C_i calibration.

After this calibration have been done, one can on ***an event by event basis***, produce the final time of the tagger reconstruction :

$$t_i^{tago} = t_i^{mean} - C_i$$

$$\begin{aligned} t_i^{tago} = & \quad ((D_i + TDC_i) - (D_l + T_l) - (< TDC_i > - < T_i >)) \\ & -(< D_i > + < T_i >) + < t_{bucket} > \\ t_i^{tago} = & < t_{bucket} > +(D_i - < D_i >) + (TDC_i - < TDC_i >) - (D_l + T_l) \end{aligned} \quad (1)$$

It is therefore equal to the time of the bucket, plus some T counter jitter $((D_i - < D_i >) + (TDC_i - < TDC_i >))$ of the order of 100 ps. This time can be compare to the RF time:

$$\begin{aligned} t_i^{tago} - t^{RF} = & \quad < t_{bucket} > +(D_i - < D_i >) + (TDC_i - < TDC_i >) \\ & -(t_{bucket} + k(event) \times 2.004) \\ t_i^{tago} - t^{RF} = & \quad (< t_{bucket} > - t_{bucket}) + (D_i - < D_i >) + (TDC_i - < TDC_i >) \\ & -k(event) \times 2.004 \end{aligned} \quad (2)$$

The jitter is small enough to allow us to determine unambiguously $k(event)$ and to also provide in the tagger output Bank the real time of the RF bucket, also called $tpho_0$ (time of the photon) providing thus the best possible time resolution :

$$tpho_0 = t_{bucket} - (D_l + T_l) = t^{RF} - k(event) \times 2.004$$

2.5 Offset to CLAS detectors

Once the time of the RF bucket has been determined, in a way that makes this determination independent of which T-counter gives the signal, the last remaining steps is to determine the offset between this RF bucket time and all CLAS detectors timing. ***The convention that has been taken is that the time value given by the tagger should be, relatively to CLAS detector's timing, the time of the tagged photon when it reaches the center of CLAS target.*** Practically, this offset is determined during calibration of the Time Of Flight scintillators. It consists basically to select highly relativistic light particles (pions) and, assuming there β is one, to reconstruct the time of vertex interaction. We want the tagger package to provide this same vertex interaction time. Therefore an additional constant called "tag2tof" (in the tagger calibration Map) is added to the times reconstructed in the previous subsection in order to get the reconstructed tagger time "ttag" and rf bucket time "tpho" projected to the center of CLAS target.

After this last correction, the tagger time can be directly subtracted to the TOF time to get particles times of flight. This can be written, for an event where the trigger was given by T counter l:

$$t_{TOF} = t_{bucket} + D_{TOF} + TOF - (D_l + T_l)$$

where D_{TOF} are delays in the Time of Flight scintillators electronic, TOF is the particle time of flight we want to measure. t_{bucket} is taken by convention to be the time of the RF bucket when the photon reaches the center of CLAS target. The value of TOF for highly relativistic pions identified by $\frac{dE}{dx}$ in the TOF scintillators, is reconstructed thanks to the Drift chamber measurement of the particle track length. Then one can fit the value of:

$$-tag2tof = < t_{TOF} - TOF - tpho_0 > = < D_{TOF} >$$

to get the electronic offset between the tagger and the TOF scintillators.

The two entries given in the TAGR bank are finally :

$$t_i^{tag} = t_i^{tag_0} + tag2tof$$

$$t_i^{tag} = < t_{bucket} > + (D_i - < D_i >) + (TDC_i - < TDC_i >) - < D_{TOF} > - (D_l + T_l)$$

$$tpho = tpho_0 + tag2tof = t_{bucket} - < D_{TOF} > - (D_l + T_l)$$

Although this calibration doesn't strictly belongs to intrinsic tagger calibration, it is nice when tagger recalibration are done, that they don't affect this overall alignment with respect to CLAS. This is simply done by measuring the position of the trigger peak $< ttag >$ before the recalibration, and once the recalibration is done to readjust the constant tag2tof so that this peak stays at the same position.

2.6 Down stream device peak calibration

Some other base peaks are calibrated for normalisation runs, which are the base peaks of the down stream devices (Total Absorption Counter, Pair Spectrometer and Pair Counter). Since the constant "tag2tof" is added to the timing of the T counter, as well as the "C_i" fine alignments, the "tag2tof" constant is also added to the timing of the Down Stream Device, as well as the mean value of the "C_i"s, so that time information are consistent between the tagger output bank TAGR, and the Down Stream Devices output banks (TACO, PSO and PCO). This is important for normalisation analysis done on a normalisation run, when one wants to time match T counters with hits in the Down Stream Devices.

For a production run, there is usually not enough statistic (PC and PS) or no statistic at all (TAC) to recalibrate the down stream device peaks. Therefore the base peak of the Down Stream Devices are not recalibrated, but they are corrected by the variation of the mean base peak value of all T's during the T counter base peak calibration. The reason for this is that when switching from a normalisation run to a production run the timing of the trigger arms (the T_i's) shifts of about 12 nanoseconds, so that all signals are shifted by 12 ns. If the base peaks of the E's and T's are recalibrated, the down stream device base peaks have to be readjusted in the same way.

With this adjustment, the timing of the PC and the PS in their output BOS bank is still consistent with the timing of the T's.

2.7 TDC slopes calibration

2.7.1 overview

Two independent calibrations can be done on the slopes:

The left - right slope balance: When left and right slopes are not correctly calibrated one versus the others, the tagger time resolution widen a little bit because of the left - right dispersion of the hits along the T counter. Depending on the transversal position of the hit the value of $\frac{t_R+t_L}{2}$ is slightly different. This is a minor effect. Yet, in the future, the value of $t_R - t_L$ will be used during coherent bremsstrahlung production to measure the position of the hit along the scintillator, if the left - right slope balance is incorrect, then this value will depend on the timing of the hit, which we want to avoid.

T counter absolute (mean) slope: When the absolute slope of one T counter is not properly calibrated, then its timing versus other detectors, or the RF, is biased depending on which part of the TDC the hit falls in. The absolute slope of one T counter can be considered to be the mean value of its left and right slopes.

In the calibration procedure those two aspects (left versus right and absolute slopes) are separated as much as possible, which means in particular that a very bad left/right balance calibration can still allow to get a quite decent absolute calibration. This is a nice feature of this method.

Figure 14 shows how the time difference $t_i^{mean} - t_{bucket}$ drifts when the T counters TDC absolute slope isn't properly calibrated. This is a fine effect, the T counter time only drifts of one nanosecond away from the good time given by the RF, on the whole TDC range (200 ns).

Calibrating the T counters absolute slopes consists of straightening this plot (assuming that RF TDC slope itself has already been properly calibrated), that is to say to cancel this drift.

A similar technic can be used to find the correct left/right TDC slope balance, which is to plot $t_i^{left} - t_i^{right}$ versus $t_i^{left} + t_i^{right}$, and to straighten it. When the correct TDC slopes are found, the left-right time distribution has to be the same (same mean value) wherever the hits fall along the TDCs.

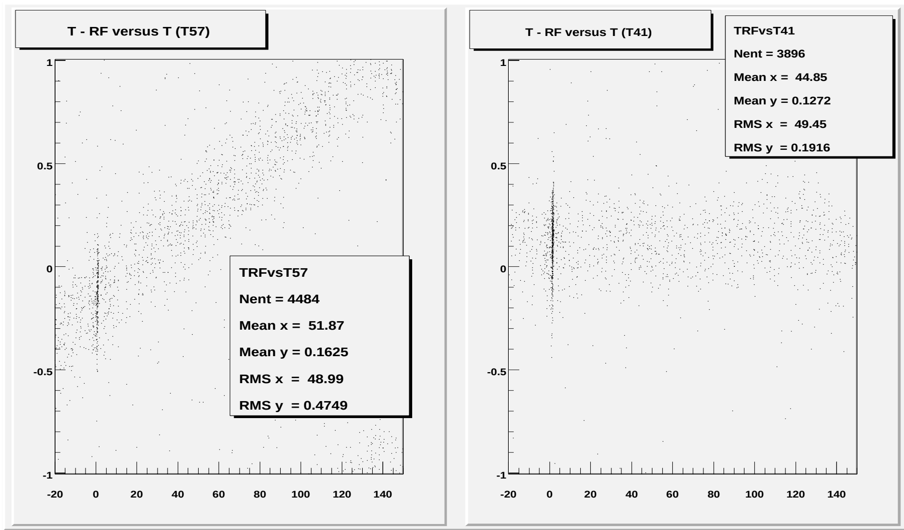

Figure 14: $t_i^{mean} - t_{bucket}$ (Y axis) versus t_i^{mean} (X axis) for one counter that is not calibrated T57, and one counter already calibrated (T41)

Let's call $s_{L/R}$ the possibly incorrect slopes used to reconstruct the hits, and $C_{L/R}$ the TDC channel where the hits fall:

$$t_L = s_L \times C_L - k_L$$

$$t_R = s_R \times C_R - k_R$$

(k_L and k_R are arbitrary constants, that aren't important for our purpose)

Let's call $S_{L/R}$ the correct slopes we want to determine, that would have allowed us to determine the true times $T_{L/R}$

$$T_L = S_L \times C_L - k'_L$$

$$T_R = S_R \times C_R - k'_R$$

Then:

$$t_L = \alpha_L \times T_L - k''_L$$

$$t_R = \alpha_R \times T_R - k''_R$$

where

$$\frac{1}{\alpha_{L/R}} = \frac{S_{L/R}}{s_{L/R}}$$

is the correction factor we have to apply to the current slopes to get the correct ones.

All the equations that will follow can be more easily written using r the ratio between the left and right TDCs:

$$r = \frac{\alpha_R}{\alpha_L}$$

and α_M the mean value of left and right corrections, which will affect the absolute slope calibration of the T counter:

$$\alpha_M = \frac{\alpha_L + \alpha_R}{2}$$

Once r and α_M will be measured, it will be easy to retrieve the α's:

$$\alpha_L = \frac{2}{1+r}\alpha_M$$

$$\alpha_R = \frac{2}{1+1/r}\alpha_M$$

2.7.2 Left - right balance: Determination of r

In order to find r, one solution would be to plot t_L versus t_R; Besides the left-right jitter, due to the dispersion of the hits along the scintillators, those points should be spread along a line with slope r. This fit is not easy to realize though. A different technic has been used, inspired by the T versus RF method (that will be described in the next paragraph) which consists of fitting the $t_L - t_R$ distribution for various ranges of t^{mean}, and by this mean to measure the slope $\beta_{R/L}$ of the distribution showed in figure 15.

$$\beta_{R/L} = \frac{\frac{<t_R - t_L>}{2}(t_2^{mean}) - \frac{<t_R - t_L>}{2}(t_1^{mean})}{t_2^{mean} - t_1^{mean}}$$

$$\frac{< t_R - t_L >}{2} = \beta_{R/L}\frac{< t_R + t_L >}{2} + K = \beta_{R/L}.t^{mean} + K$$

Let's express $\beta_{R/L}$ as a function of r and α_M:

$$\frac{< t_R - t_L >}{2} = \frac{\alpha_R. < T_R > -\alpha_L. < T_L >}{2}$$

If one recalls that left and right TDC hits come from the same electron hit, that occurred at time $\frac{T_R+T_L}{2}$, with only one transversal jitter:

$$T_R = \frac{T_R + T_L}{2} + \frac{T_R - T_L}{2}$$

$$T_L = \frac{T_R + T_L}{2} - \frac{T_R - T_L}{2}$$

one can easily deduce:

$$\frac{< t_R - t_L >}{2} = \frac{\alpha_R - \alpha_L}{2}.\frac{< T_R > + < T_L >}{2} + \frac{\alpha_R + \alpha_L}{2}.\frac{< T_R > - < T_L >}{2} + K$$

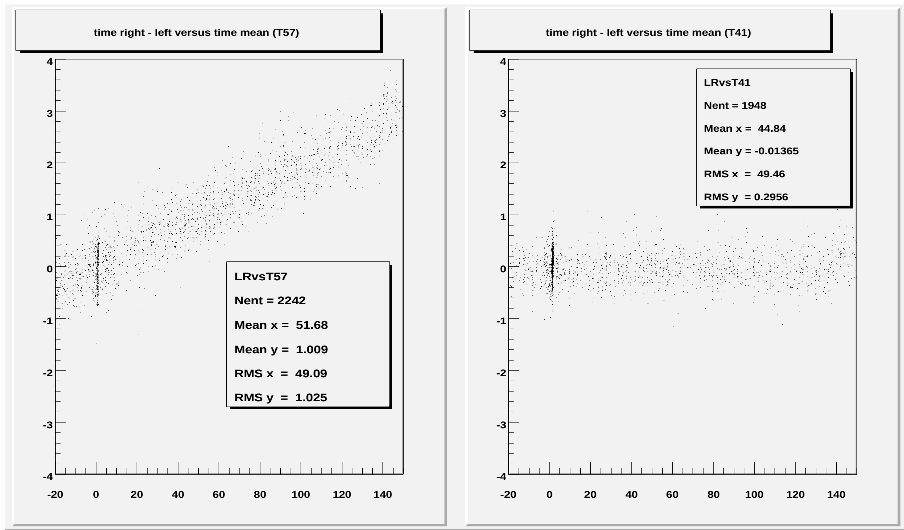

Figure 15: $\frac{t_R - t_L}{2}$ (Y axis) versus $\frac{t_R + t_L}{2}$ (X axis) for one counter not calibrated (T57) and one counter already calibrated (T41)

Since $T_{L/R}$ are the true times, $\frac{<T_R> - <T_L>}{2}$ is an arbitrary constant that doesn't change along the TDC range, on the contrary if the slopes are badly calibrated $((\alpha_R - \alpha_L) <> 0)$, $\frac{<t_R - t_L>}{2}$ drifts depending of which part of the TDC range we are looking at, as a function of $\frac{<T_R> + <T_L>}{2}$):

$$\frac{< t_R - t_L >}{2} = \frac{(\alpha_R - \alpha_L)}{2} . \frac{< T_R > + < T_L >}{2} + K_3$$

Using $t_{L/R} = \alpha_{L/R} . T_{L/R}$, and assuming α_L and α_R are close, and close to α_M, we can approximate the above expression to:

$$\frac{< t_R - t_L >}{2} = \frac{\alpha_R - \alpha_L}{2.\alpha_M} . \frac{< t_R > + < t_L >}{2} + K_3$$

thus:

$$\beta_{R/L} = \frac{r - 1}{r + 1}$$

$$r = \frac{1 + \beta_{R/L}}{1 - \beta_{L/R}}$$

and:

$$\alpha_L = (1 - \beta_{R/L}).\alpha_M$$

$$\alpha_R = (1 + \beta_{R/L}).\alpha_M$$

2.7.3 Absolute slope correction: Determination of α_M

In order to determine the value of α_M, one measures the slope of the drift of $t^{mean} - t_{bucket}$ as a function of t^{mean} (figure 14).

This slope is:

$$\beta_{RF} = \frac{< t^{mean} - t_{bucket} >_{(t_2^{mean})} - < t^{mean} - t_{bucket} >_{(t_1^{mean})}}{t_2^{mean} - t_1^{mean}}$$

we can express β_{RF} as a function of α_M:

$$t^{mean} - t^{bucket} = \frac{\alpha_R.T_R + \alpha_L.T_L}{2} - t_{bucket} + K_4$$

$$t^{mean} - t^{bucket} = \frac{(1+\beta_{R/L}).\alpha_M.T_R + (1-\beta_{R/L}).\alpha_M.T_L}{2} - t_{bucket} + K_4$$

$$t^{mean} - t^{bucket} = \alpha_M.\frac{(T_R + T_L)}{2} + \alpha_M.\beta_{R/L}\frac{(T_R - T_L)}{2} - t_{bucket} + K_4$$

Therefore, pointing out that $< T_R > - < T_L >$ is an arbitrary constant, and that $< t_{bucket} >= \frac{<T_R>+<T_L>}{2} + K_5$:

$$< t^{mean} - t^{bucket} >= (\alpha_M - 1).\frac{< T_R > + < T_L >}{2} + K_6$$

$$\beta_{RF} = \alpha_M - 1$$

$$\alpha_M = 1 + \beta_{RF}$$

$$\alpha_L = (1 - \beta_{R/L}).(1 + \beta_{RF})$$

$$\alpha_R = (1 + \beta_{R/L}).(1 + \beta_{RF})$$

2.7.4 conclusion on Slope calibration.

By measuring the slope $\beta_{R/L}$ of the drift of the quantity $\frac{<t_R - t_L>}{2}$ versus t^{mean}, and the slope β_{RF} of the drift of the quantity $< t_i^{mean} - t_{bucket} >$ versus t^{mean}.

On can deduce easily the correction $\alpha_{L/R}$ to apply to the slopes previously used, and thus the new (correct) slopes:

$$S_L = s_l \times \frac{1}{(1-\beta_{R/L}) \times (1-\beta_{RF})} \simeq (1+\beta_{R/L}) \times (1+\beta_{RF})$$

$$S_R = s_r \times \frac{1}{(1+\beta_{R/L}) \times (1-\beta_{RF})} \simeq (1-\beta_{R/L}) \times (1+\beta_{RF})$$

Typical values of the Tagger T counters TDC slopes are given in table 7.

41.	49.620	50.250
42.	49.890	50.070
43.	49.620	50.150
44.	49.630	50.230
45.	49.430	49.810
46.	49.680	49.150
47.	50.070	49.620
48.	49.400	49.660

Table 7: some T counters TDC slopes (left and right) given in ps/channel, for calibrated counters.

2.8 Conclusion on calibration methods.

All those constants depend on each others, and must be calibrated in the following order:

1. The TDC slopes.
2. The Base peak positions.
3. C_i's (relative alignment of T's).
4. alignment to CLAS.

At each of those steps some specific modifications of RECSIS processing are done:

TDC slopes: T counters base peak are temporarily recalibrated. E counters hit are not requested in the reconstruction to get more statistic on T's even when some E counters are dead.

Base peak positions: E rebinning, T rebinning, E-T matching coincidences windows are set very large so that no hits are missed because previous calibrations were inappropriate.

C_i's: Depending whether the run is a production or a normalisation run, the reference detector used is the start counter or the TAC. If in the future the T rebinning time window is set by default to be very narrow (of the order of 2 nanoseconds), then this window will have to be widen at this step to insure that T rebinning is done even if the previous calibration was not good.

3 Calibration software tutorial

An automatic treatment has been developed in order to obtain the calibration constants needed by the tagger package. This software is written in C++, and based on the ROOT CERN libraries. See *http://root.cern.ch* for more information on the ROOT analysis framework.

It is checked in CLAS CVS repository in the directory packages/clasroot/tagger_calib. This code depends on the bosio, clasbos, bosbank and photonBeamAna, and arguments packages, which can only be used on Sun platform at the time of this writing.

Fits are obtained with the ROOT version of MINUIT. The program will create ASCII files in which the calibration constants are stored, and a ROOT file which contains control histograms.

3.1 setting up things

First, one has to have the ROOT environment setup, i.e to have the environment variables **ROOTSYS** and **LD_LIBRARY_PATH** pointing to the location of the ROOT binaries and libraries (cf. *http://root.cern.ch*). At Jlab, this can be done with: setup root

At the time of this writing, ROOT packages are not part of the librarian distribution, because they can only be compiled on Sun platform, therefore individuals have to compile their own libraries, and executable: This is done by retrieving the appropriate packages from the repository.

```
cvs co clasbos
cvs co bosbank
cvs co clasroot/arguments
cvs co clasroot/photonBeamAna
cvs co clasroot/tagger_calib
```

and compile them:

```
cd clasbos; make;
cd bosbank; make;
cd clasroot/arguments; make
cd clasroot/photonBeamAna; make;
cd clasroot/tagger_calib; make;
```

In case of problem, one can e-mail the hall B mailing list "clas_root", to get help.

At this last step a binary $TOP_DIR/bin/SunOS/ROOTcalib_sh is produced This binary is the one that will be used to calibrate the tagger.

The full calibration proceeds through 5 steps:

1. Slopes calibration.
2. Peaks positions calibration.
3. Ci constants calibration.
4. tag2tof adjustment.
5. Monitor histograms to check the calibration quality.

During each of those steps, the tag package is run, using the calibrations that were obtained during the previous steps and for the remaining constants the calibrations that are in the Map Manager.

At the begining of the execution, the old constants are put in text files, of the format:

```
tag<constant>.old.<run number>
```

At each step of the calibration new constants are determined and put in new files:

```
tag<constant>.new.<run number>
```

ROOTcalib_sh -h will print out the usage and different options:

```
Usage: calib_sh [-f <file1> <file2> ... ] [-o <rootfile>]  ...
Options are:
        [-o] output root file name
        [-f] file name
        [-h] print this message
        [-M] mode of process : 0 TDC Slope calibration (production run ONLY)      (then 1)
                               1 Calibration of peak position                     (then 2)
                               2 Calibration of Ci constants                      (then 4)
                               4 tag2tof final adjustement + control histograms   (then STOP)
                               9 Update CLAS parms tagger map, using local text file
        -n (-> normalisation run, therefore use TAC for Ci alignement ..  )
        -N[#]   Only analyze # number of events
         [-d] display histograms while processing
         [-D] debug mode histogram creation
        [-v] verbose
```

The -M options tells at which level to start the calibration. -M0 tells to start from the TDC slope calibration, but most of the time, if the TDC modules aren't changed, the TDC slopes don't have to be recalibrated.

The software requires the use of -f option to name the raw data input file. This file can be either a normalization run or a production run, with the following restrictions:

1. TDC slopes calibration can only be done with a production run.

2. In case of a normalisation run, the option -n must be used to tell the software to use the TAC as reference detector instead of the Start Counter.

-N options sets the number of events to proceed. The default number is 250 000, however good calibrations can be obtain starting at 100 000 events. In order to obtain systematically good fits, a minimum statistic requirement has been set to 10 events for each bin. We empirically noticed that this requirement is fullfiled for 200 000 events analyzed.

for a standard calibration the command line will be :

ROOTcalib_sh -f /path/clas_99999.A00 -M0 -N300000 -o calib99999.root

this will produce the following text files:

```
tag2tof.new.99999
tag2tof.old.99999
tagCalCi.new.99999
tagCalCi.old.99999
tagTDCCal.new.99999
tagTDCCal.old.99999
tagTDCCalE.old.99999
tagTTransTable.old.99999
tagdsdcal.new.99999
tagdsdcal.old.99999
tagposEpeak.new.99999
tagposEpeak.old.99999
tagposTpeak.new.99999
tagposTpeak.old.99999
```

plus the ROOT output file calib99999.root

(if the program is run with the -M1 option (starting from peak position calibration) there will be no tagTDCCal.new.99999 file produced, with the -M2 option no new peak position files, and so on..)

Once the calibration is done, one has to control the quality of the calibration, looking at control histograms that are contained in the ROOT output file. If those control histograms are satisfying, one can then put the constants written in the text file into the Map Manager by running the program with the -M9 option.

We are now going to describe how to interpret the control histograms, and how to put the new constants into the map manager.

3.2 ROOT output file manipulation

First, one must start ROOT:

```
> root

  *******************************************
  *                                         *
  *        W E L C O M E  to  R O O T       *
  *                                         *
  *   Version   2.21/06  18 February 1999   *
  *                                         *
  *  You are welcome to visit our Web site  *
  *          http://root.cern.ch            *
  *                                         *
  *******************************************

CINT/ROOT C/C++ Interpreter version 5.13.89, Feb 17 1999
Type ? for help. Commands must be C++ statements.
Enclose multiple statements between { }.
root [0]
```

At this point, three usefull commands are: **.h** to get help, **.q** to quit root, and **.!<command>** to invoque a shell command (for exemple, .!ls list the file in the current directory). ***All line starting by a point are interpreted as ROOT commands, all the other lines are interpreted as C++ code lines.***

Once in the interpretor, the root file has to be opened, using C++ interpreted code;

```
root [3] TFile f("calib99999.root")
```

(We just created a C++ TFile class object f, and associated it with the file "calib99999.root".)

Then, the content of the file can be obtained with : .ls

```
root [5] .ls
TFile**         calib99999.root
 TFile*         calib99999.root
  KEY: TDirectory       pass00;1        peak adjustment for TDC slope calibration
  KEY: TDirectory       pass0;1 TDC slope calibration
  KEY: TDirectory       pass1;1 Peak position calibration
  KEY: TDirectory       pass2;1 Ci calibration
  KEY: TDirectory       pass4a;1        Tag2tof adjustment
  KEY: TDirectory       pass4b;1        Control of calibration
```

This file contains several directories, one (sometimes two) for each step of the calibration. Each directories contains the histograms that have been used for the calibration, they can be viewed by experts in case something went wrong during the calibration. For just the purpose of controling the quality of the calibration, one just have to look in the last directory ("pass4b", Control of calibration). To do this, the method cd() of the object pass4b belonging to the class TDirectory has to be invoqued (C++ interpreted code):

```
root [6] pass4b.cd()
(Bool_t)1
root [7] .ls
TDirectory*             pass4b  Control of calibration
 KEY: TH1F      nbhits;1        Number of Hodoscope hits reconstructed per event
 KEY: TH1F      distri_T61;1    T counters 61 distribution
 KEY: TH1F      distri_T121;1   T bin 121 distribution
 KEY: TH1F      distri_E48;1    E bin 48 distribution
 KEY: TH1F      distri_E384;1   E counters 384 distribution
 KEY: TH1F      distri_E767;1   E bin 767 distribution
 KEY: TH2F      tdc_left_vs_right;1     TDC calibration check, left right balance
 KEY: TH2F      tdc_T_vs_RF;1   TDC calibration check, T - RF balance
 KEY: TH2F      tdc_T_RFvsRF1;1 TDC calibration check, (T-RF) vs RF1
 KEY: TH2F      tdc_T_RFvsRF2;1 TDC calibration check, (T-RF) vs RF2
 KEY: TH1F      time_PC;1       PC timing spectra
 KEY: TH2F      time_PSvsPSid;1 PS timing spectra
 KEY: TH1F      time_TAC;1      TAC timing spectra
 KEY: TH1F      time_T;1        T timing spectra
```

```
 KEY: TH1F      time_T_fine;1   T timing spectra arround peak
 KEY: TH1F      time_T_PC;1     T - PC timing spectra
 KEY: TH1F      time_T_TAC;1    T - TAC timing spectra
 KEY: TH2F      time_T_PSvsPSid;1       T - PS timing spectra
 KEY: TH1F      time_T_ST1;1    T - ST1 timing spectra
 KEY: TH1F      time_T_STR;1    T - STR timing spectra
 KEY: TH2F      time_T_ST1vsSTid;1      T - ST1 timing spectra
 KEY: TH1F      time_RF;1       RF structure (time T - RF)
 KEY: TH2F      time_adjT;1     time T(i) - T(i+1) versus T id
 KEY: TH2F      time_TvsTid;1   time T vs T id
 KEY: TH2F      time_E_TvsEid;1 time T - time E vs E id
 KEY: TH2F      time_T_RFvsTid;1        time T - time RF vs T id
 KEY: TH2F      time_T_PCvTid;1 time T - time PC vs T id
 KEY: TH2F      time_T_PSvsTid;1        time T - time PS vs T id
 KEY: TH2F      time_T_TACvsTid;1       time T - time TAC vs T id
 KEY: TH2F      time_T_ST1vsTid;1       time T - time St1 vs T id
 KEY: TH2F      time_T_STRvsTid;1       time T - time StR vs T id
 KEY: TH2F      time_PCvsTid;1  time PC vs T id
 KEY: TH2F      time_PSvsTid;1  time PS vs T id
 KEY: TH2F      time_TACvsTid;1 time TAC vs T id
 KEY: TH2F      time_ST1vsTid;1 time ST1 vs T id
 KEY: TH2F      time_STRvsTid;1 time STR vs T id
 KEY: TTree     TreePBA;1       photon Beam event tree
```

This diretory contains a list of histograms. ROOT identifies histograms by their names, to visualise them, you can invoque the Draw() method of the histograms class:

```
root [8] nbhits.Draw()
Warning in <MakeDefCanvas>: creating a default canvas with name c1
```

If one wants to browse back to another directory, one has to go back to the file directory, using the cd() method on the file, and the cd() method for the other directory:

```
root [9] f.cd()
(Bool_t)1
root [10] pass00.cd()
(Bool_t)1
```

We will now describe which histograms of the pass4b directory must be viewed to control the processing of each step of the calibration:

3.3 TDC slopes check

The quality of the left/right balance calibration can be viewed via the histogram *tdc_left_vs_right*. This histograms shows the quantity $\frac{t_R - t_L}{2}$ versus t^{mean} for all T's in one plot, in increasing order from left to right, and from bottom to top (Left-bottom plot is for T1, right bottom plot is T8, left top plot is T57). Figure 16 shows how this histogram looks like before, and after calibration.

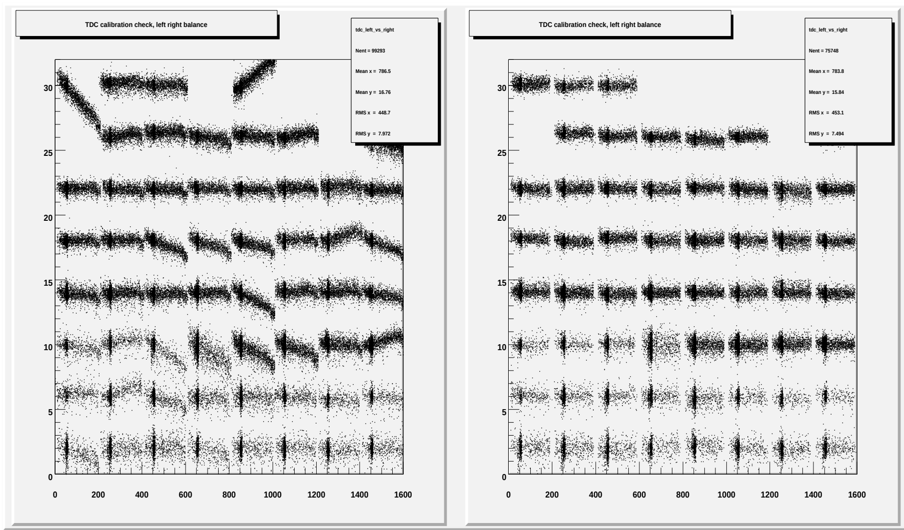

Figure 16: Control histogram for left right TDC balance, when the TDCs aren't properly calibrated they show drifts (slopes) (left histogram), when they are properly calibrated they should be horizontal lines (right histogram)

The quality of the absolute versus RF calibration can be viewed via the histogram *tdc_T_vs_RF*. This histograms shows the quantity $t^{tag} - t_{bucket}$ versus t^{tag} for all T's on one plot, in the same order than for the *tdc_left_vs_right* histogram. Figure 17 shows how this histogram looks like before, and after calibration.

3.4 T counter signals alignment at the trigger.

This is not stricly speaking a check of the tagger calibration, because the T counter signals alignment at the trigger level is an hardware property that has nothing to do with software, or software calibrations. Yet the importance of this alignment for software reconstruction, has been underlined in the previous section. Furthermore, this alignement is crucial for the trigger itself in order to make the coincidence window as narrow as possible with the Start Counter, and to lower the probability of accidentals between the tagger and the start counter in the trigger.

This trigger alignement can be checked through the histograms *time_PCvsTid*, *time_PSvsTid* and *time_TACvsTid* for a normalisation run, or *time_ST1vsTid* and *time_PSvsTid* for a production run. Those histograms show the position of the base peak of the reference detector as a function of which T counters gave the trigger, thus showing the quantity:

$$< t_{ref} > - < peak_{ref} > = < D_{ref} > - < peak_{ref} > -(D_l + T_l)$$

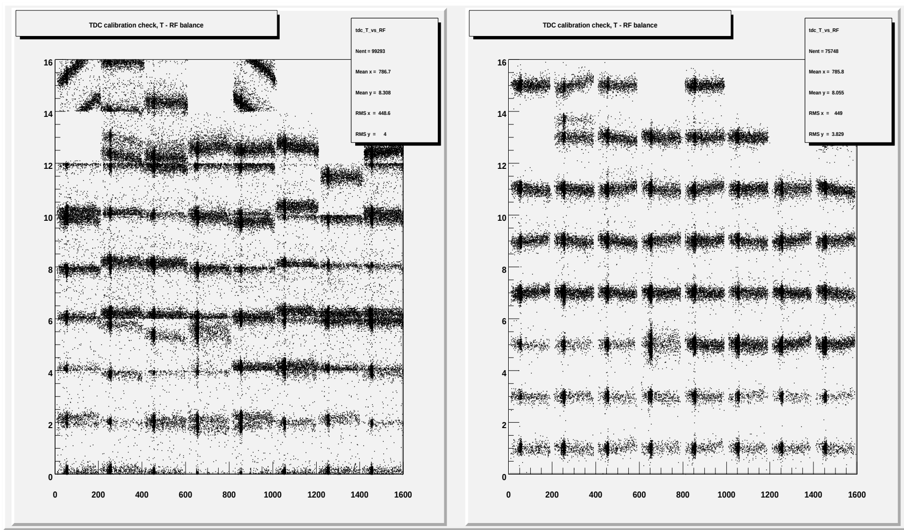

Figure 17: Control histogram for absolute TDC calibration, when the TDCs aren't properly calibrated they show drifts (slopes) (left histogram), when they are properly calibrated they should be horizontal lines (right histogram). In this particular exemple, the C_i constants were also wrong before calibration, thus the plots were not centered on 0.

The quantity $< D_{ref} > - < peak_{ref} >$ is close to zero and is independant of the T counter that gave the trigger. On the contrary the quantity $(D_l + T_l)$ depends on the T counter and shows directly the variation in timing of the signals as they go through the whole tagger electronic to the trigger input.

Figure 18 shows this alignment for a normalisation run, and figure 19 for a production run.

3.5 Base peak calibration check

T base peak versus E base peak calibration can be checked via the *time_E_TvsEid* histogram. It shows the time difference between E's and T's versus E id. It should be flat and centered on zero. It should be within the coincidence window (+/- 10 ns). Figure 20 shows this histograms before and after calibration.

3.5.1 Down Stream Devices

For a normalisation run, one can have a look at the down stream devices base peak calibration, with the histograms *time_T_PC*, time_T_TAC and time_T_PSvsPSid. (figure 21 shows those histograms before calibration for a G6 normalisation run).

For a production run, one can check with the same histograms that the down stream devices base peak as been properly adjusted to reflect the shift in the E

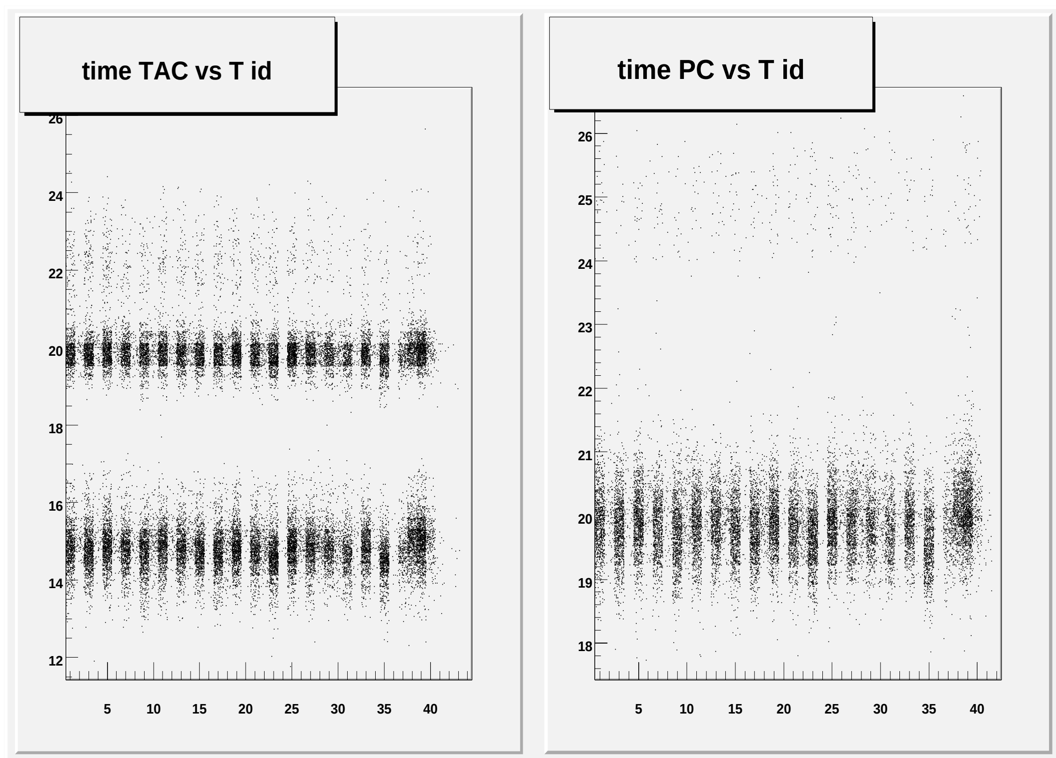

Figure 18: T counters signals alignment at the trigger level for a normalisation run of the G6 period, X axis is the T bin Id, Y axis is the time in nanosecond. Right plot is using the PC as reference, left plot is using the TAC as reference (because of the particular trigger of G6 normalisation run: TAC/50+PC+PS, the TAC spectra shows 2 peaks corresponding to the TAC giving the trigger and to the PC giving the trigger. Is is however still possible to check the alignment in time of thoses peaks versus the T counter Id.

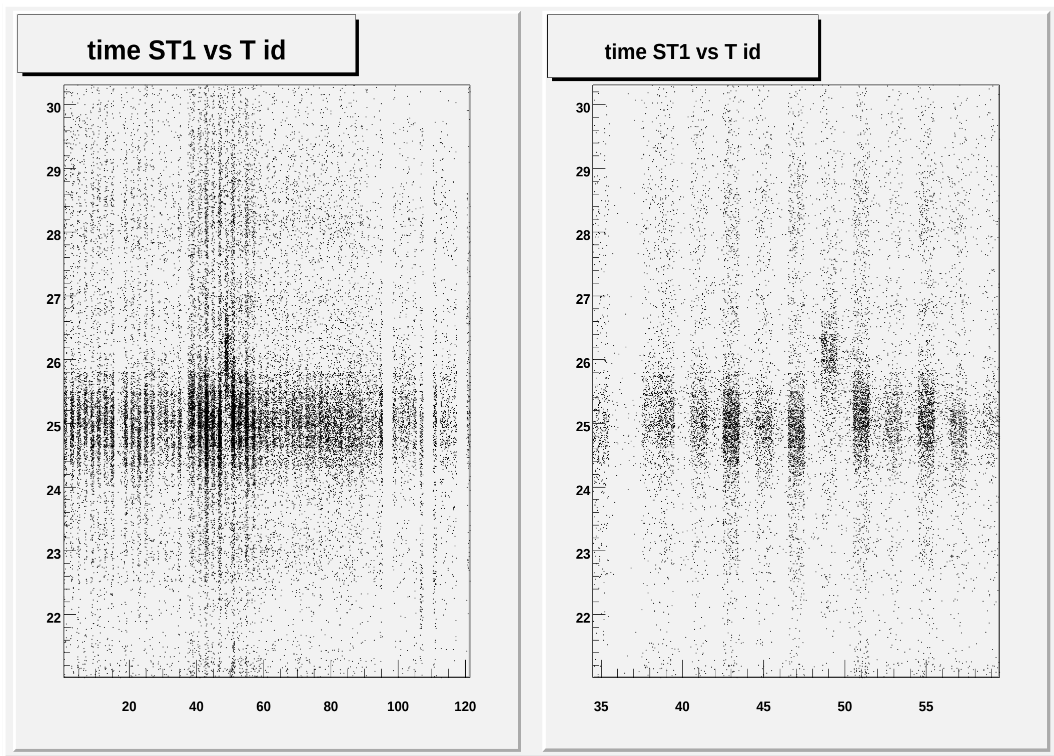

Figure 19: T counters signals alignment at the trigger level for a production run of the G1 period, using the start counter, X axis is the T bin Id, Y axis is the time in nanosecond. the left plot is a zoom on T counter 25 (bin 49) which is stlightly misaligned.

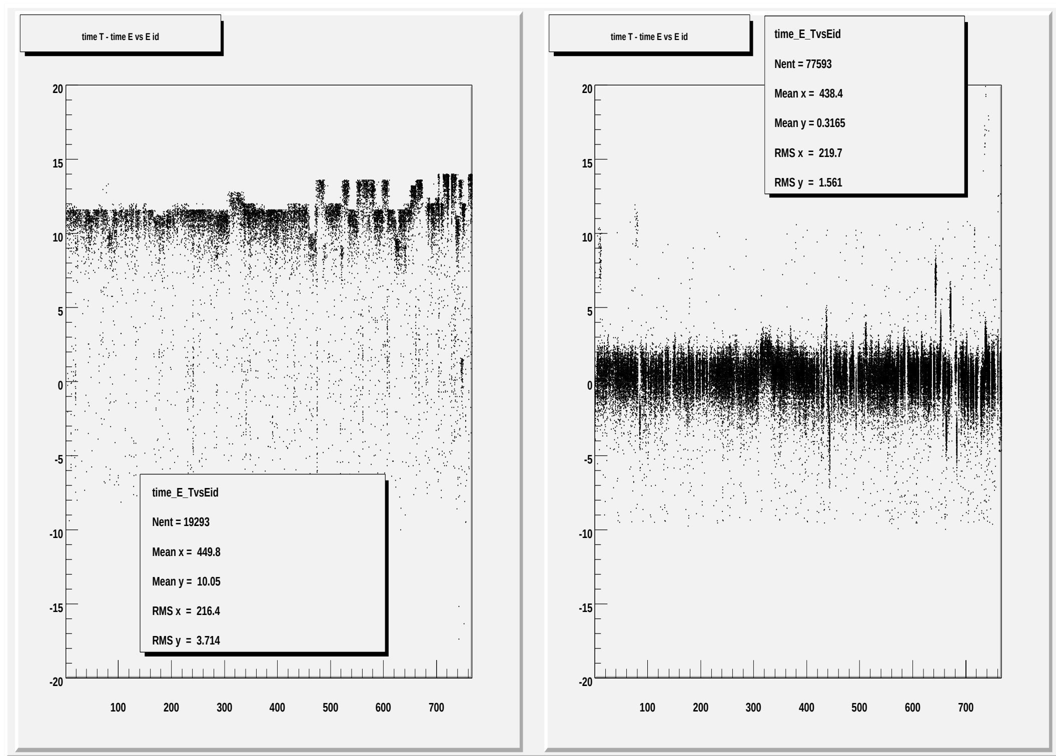

Figure 20: E - T coincidence plot. Left plot shows a case when the global calibration is off, that is to say E.T matched counters time difference are often out of the coincidence window (the coincidences peaks are on the edge of the coincidence window, and cut out.). Right plot shows a better distribution even though some E counters could still be a little bit better calibrated.

and T counter base peaks. (figure 22 shows those histogram after calibration for a G1 production run)

If the base peak position of the E's and T's counters are updated in the Map Manager, after a new calibration, it is necessary to also update the down stream devices base peak values in the map manager, using the new constants produced during the same calibration.

3.6 Ci calibration check

IT is the most crucial calibration check, the one that garanties that the tagger package will pick up the correct RF bucket for time of flight measurement and Particule Identification.

C_i calibration check is done via three histograms:

1. *time_RF* shows the overall fine alignment of the T's versus the RF.

2. *time_T_RFvsTid* shows the fine alignment with respect to the RF versus the T bin id.

3. *time_T_TACvsTid* for a normalisation run, or *time_T_ST1vsTid* and *time_T_PSvsTid* for a production run show the absolute alignment with respect to the reference detector that proves that alignement has been done to the same bucket for all T's.

Fig 23 shows *time_RF* and *time_T_RFvsTid* histograms, before calibration. In this exemple, T's are overall off by one nanosecond (not centered on 0), and T to T alignement is bad, especially for id greated than 90.

Fig 24 shows *time_RF* and *time_T_RFvsTid* histograms, after calibration.

This is however not enough to control the alignment of the T counters, because T counters can possibly be aligned to different buckets, giving histograms* time_T_RFvsTid *and* time_RF *that looks perfectly good while some T's are off by multiples of 2.004 nanoseconds.

Alignment to the same RF bucket is done via the histograms *time_T_TACvsTid* for a normalisation run or *time_T_ST1vsTid* and *time_T_PSvsTid* for a production run.

time_T_ST1vsTid shows directly what the timing of the tagger is versus the start counter, i.e versus the hadronic reaction timing, and therefore allows to tell if one of the T counters is off by one or several buckets with respect to the other T counters. It is however difficult to read because there are a lot of accidentals between the tagger and the start counter, which give hits that are off by one or several buckets that can easily hide T counters that are really off by some buckets. Figures 25 and 26 shows that with some cautions this histogram can however be used to detect misaligned T counters.

Figure 27, taken during a G1 production run, shows that the Pair Spectrometer can also be used to check that all T counters are aligned to the same bucket. Pair spectrometer hits during a production run can be found in coincidence with those T's that fire but don't trigger. The statistic is low, it is equal

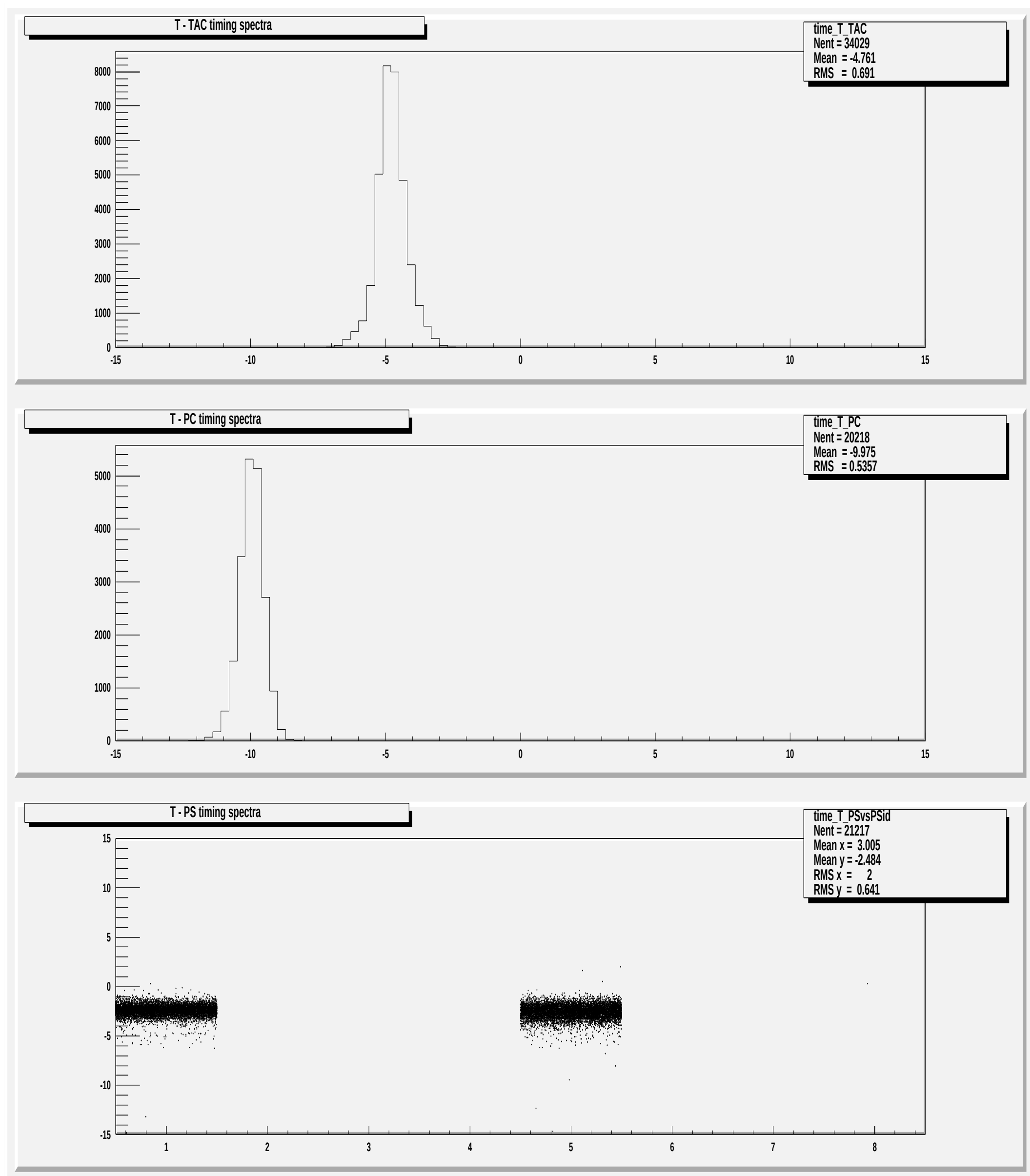

Figure 21: NORMALISATION run, before calibration: first plot is T - TAC time, second plot is T - PC time, third plot is T - PS time versus PS id. Before the calibration those plots are not centered on 0. This is a G6 run, therefore only two PS paddles were operational.

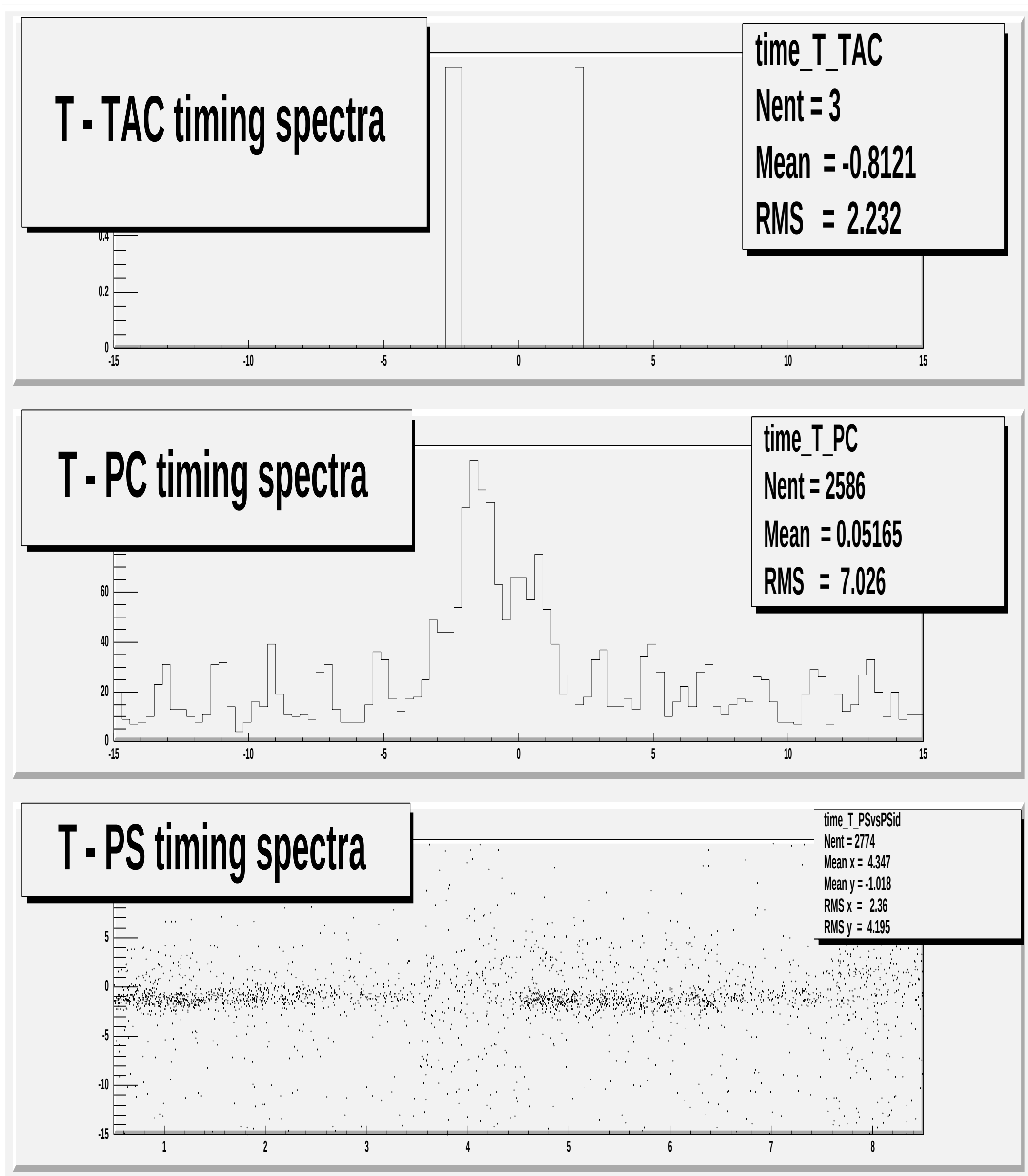

Figure 22: PRODUCTION run, after calibration: first plot T - TAC time is almost empty, second plot is T - PC time shows a lot of back-ground, third plot is T - PS time versus PS id.

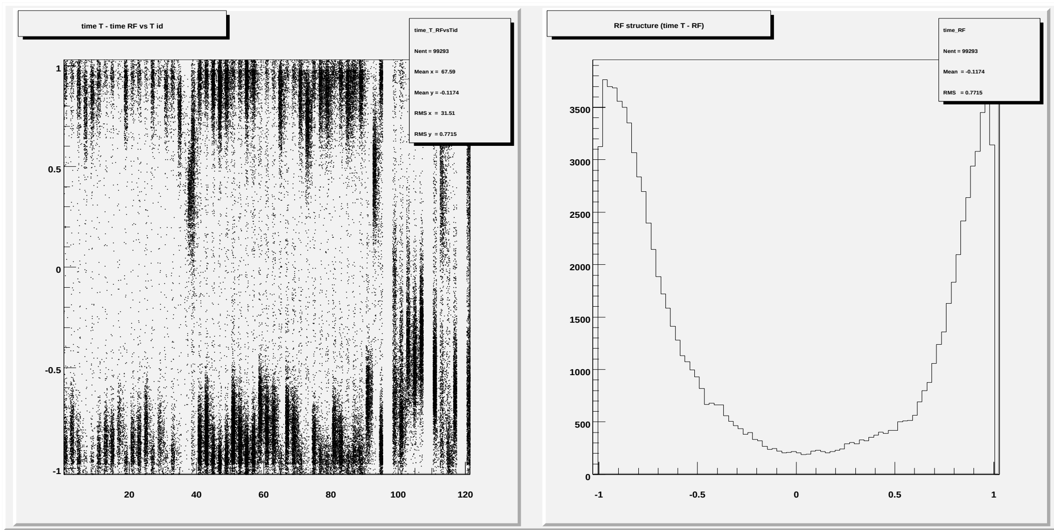

Figure 23: BEFORE CALIBRATION: fig (a) : Alignment of the T counters to the RF, (X Axis is T bin Id, Y Axis is time difference.) fig (b) : projection of (a) onto the time axis.

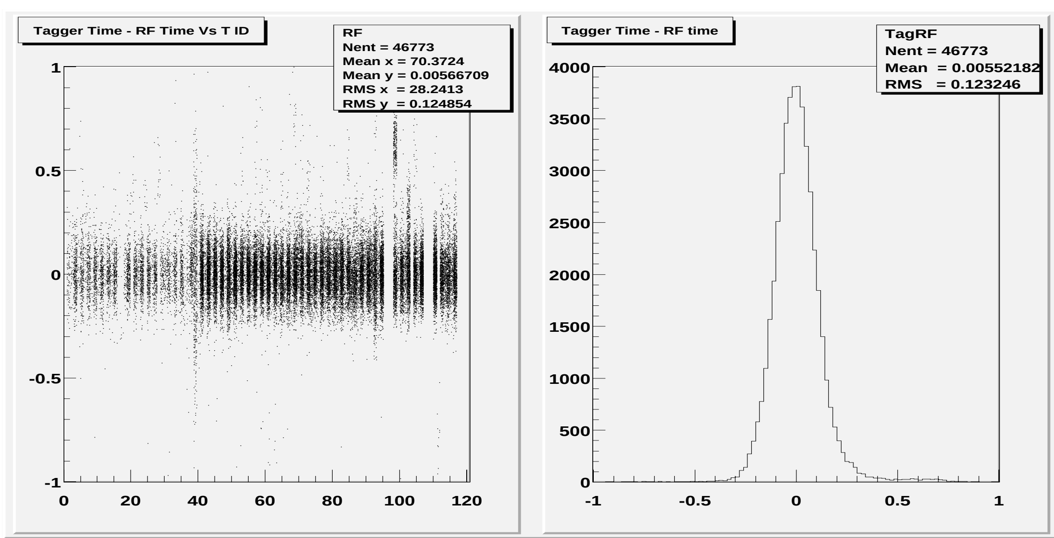

Figure 24: AFTER CALIBRATION: fig (a) : Alignment of the T counters to the RF, (X Axis is T bin Id, Y Axis is time difference.) fig (b) : projection of (a) onto the time axis.

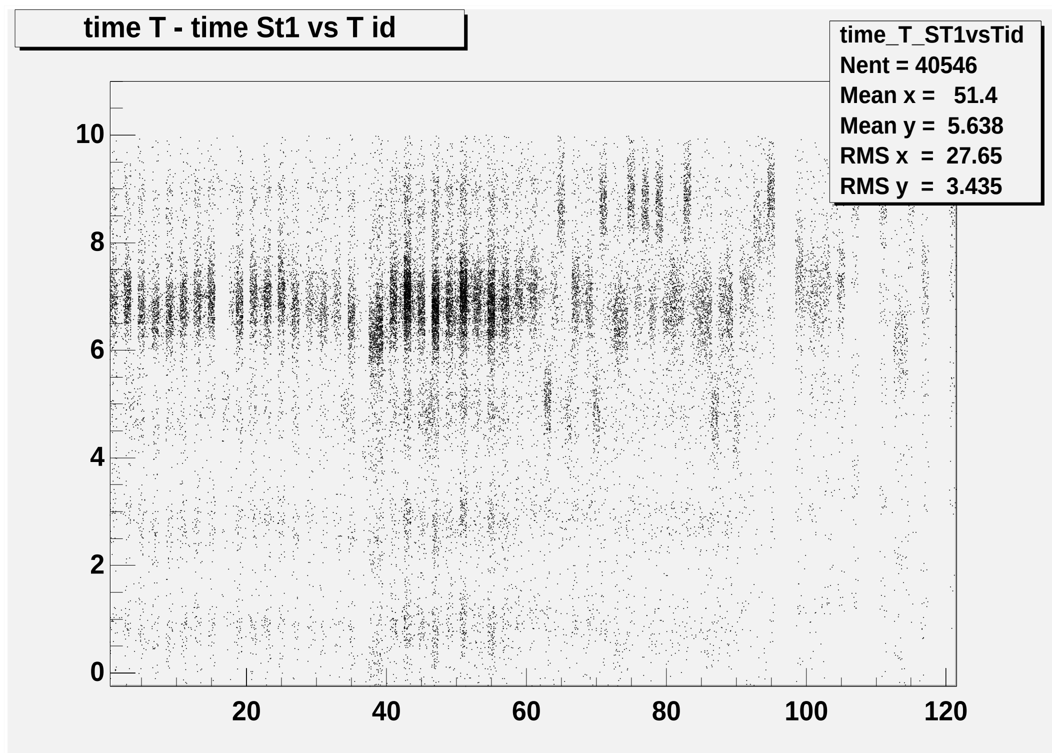

Figure 25: BEFORE CALIBRATION: tagger - Start counter time difference versus the T id. X axis is the T bin Id, Y axis is the time difference in nanoseconde. One sees that the T's are not perfectly alligned, and that some of them (bins 63,65,67 foe exemple) are aligned to some wrong buckets.

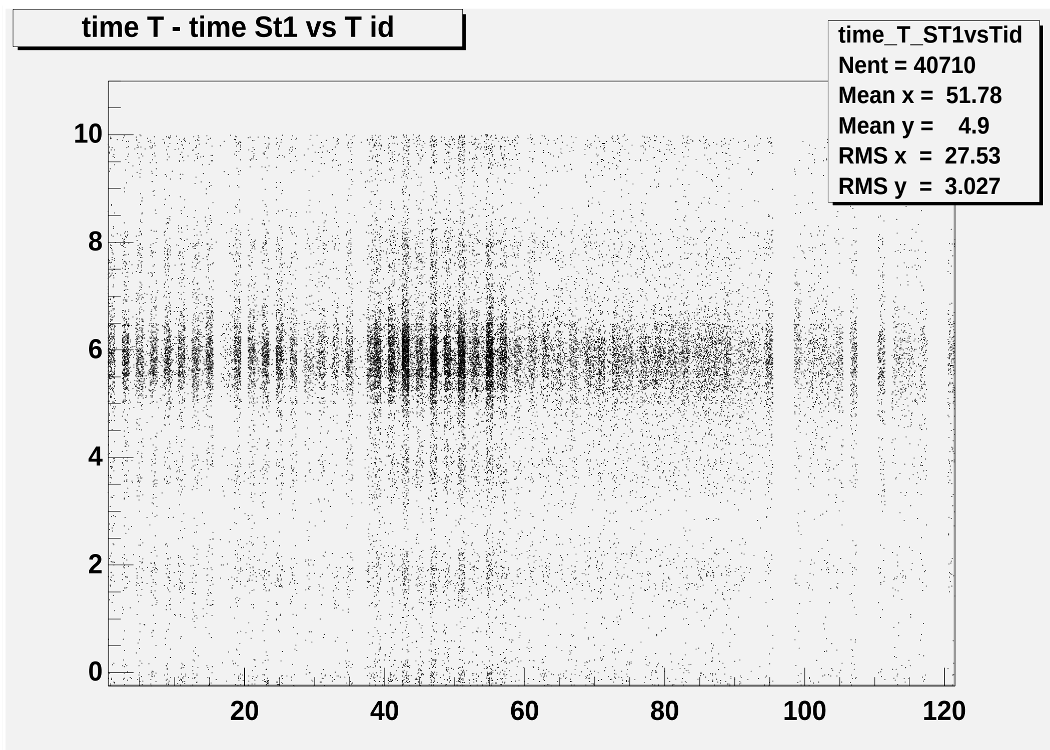

Figure 26: AFTER CALIBRATION: tagger - Start counter time difference versus the T id. X axis is the T bin Id, Y axis is the time difference in nanoseconde. The alignment is clear.

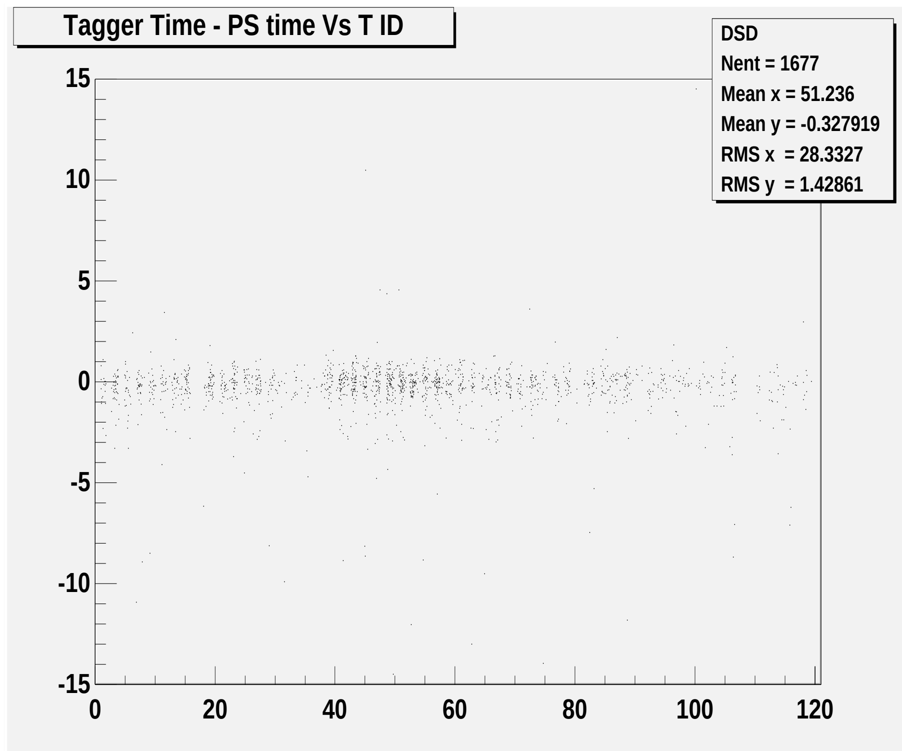

Figure 27: AFTER CALIBRATION: Tagger - PS time difference. (X axis is the T id, Y axis is the time difference in nanosecond. This is for a production run, the statistic is lower than for the start counter, but the alignment is also clear.

to the probability of having one T counters firing in the TDC range, when another T counter gave the trigger, times the efficiency of the pair spectrometer. This is however enough to get a clear plot.

3.7 T rebinning

T rebinning is also based on the C_i's and can be checked via the histogram *time_adjT* , this histogram shows the time difference between two T's when they are gathered into one hit, versus the T id. Figure 28 shows this distribution. One must just verify that all peaks are within the time window, which means adjacents hits in time are proprely gathered into one hit. This histogram shows in particular, that even after C_i calibration and correction, the T counters timing shows shifts of the order of a few hundreds of picoseconds, when they fire alone or in coincidence with the next T counter. This explains why the T rebinning cannot be done with a resolution better than 2ns.

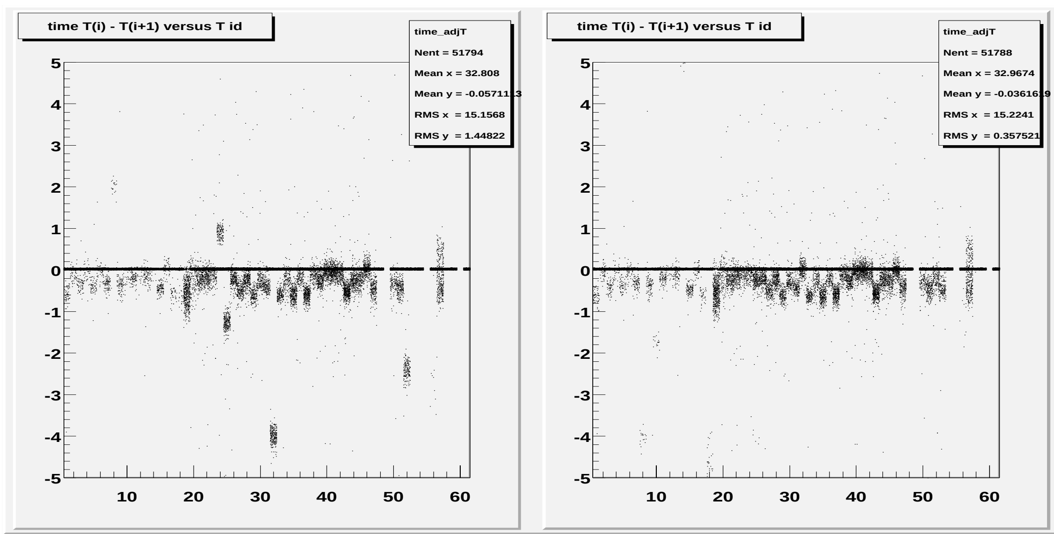

Figure 28: Difference in time between T's that are binned together, before and after calibration. (X Axis is the T counter number, Y axis is the time difference between the two counters.) One sees that even after calibration, there are offsets of the order of a few hundreds of picosecond, which means that the value of the C_i of one T counter is different when it fires alone or in coincidence with the next T counter.

3.8 Puting the new constants in the Map Manager

This is done via the command:

ROOTcalib_sh -M9

The user is prompted for a run number at which to put the calibration constants in the Map Manager, and the name of the file containing the new calibration constants (usually this will be tag<constant>.new.99999).

Note that one can edit the text files to adjust some constants, if necessary, before puting them in the map. This is typically what has been done to put the T-translation table in the map, the "tagTTransTable.new.99999" was in this case produced by hand.

If one doesn't want to change a particular set of constants, one can just enter a dummy name file, like "x". (as shown in the exemple below). A typical session (no TDC slope calibration, all the other calibrations) will be :

```
 Put values at which RUN ?
99999
 Reading T Translation table in what file ?
x
 !! No tagger T translation file  !!
 Reading E TDC calibration constants from file ?
x
 !! No tagger E-counter TDC calibration file  !!
 Reading T TDC calibration constants from file ?
x
 !! No tagger T-counter TDC calibration file  !!
 Reading Ci calibration constants from file ?
tagCalCi.new.99999
Error code = MAP_USER_WARN_NOFIND
warning: map_rem_arr: No values exist for this time (99999)
 for subsystem (tag_t), item (ci).
 Reading T peak position calibration constants from file ?
tagposTpeak.new.99999
Error code = MAP_USER_WARN_NOFIND
warning: map_rem_arr: No values exist for this time (99999)
 for subsystem (tag_t), item (dt_left).
Error code = MAP_USER_WARN_NOFIND
warning: map_rem_arr: No values exist for this time (99999)
 for subsystem (tag_t), item (dt_right).
 Reading E peak position calibration constants from file ?
tagposEpeak.new.99999
Error code = MAP_USER_WARN_NOFIND
warning: map_rem_arr: No values exist for this time (99999)
 for subsystem (tag_e), item (dt).
 Reading DSD calibration constants from file ?
tagdsdcal.new.99999
Error code = MAP_USER_WARN_NOFIND
warning: map_rem_arr: No values exist for this time (99999)
 for subsystem (pc), item (ped).
```

```
Error code = MAP_USER_WARN_NOFIND
warning: map_rem_arr: No values exist for this time (99999)
 for subsystem (pc), item (tdc_w).
Error code = MAP_USER_WARN_NOFIND
warning: map_rem_arr: No values exist for this time (99999)
 for subsystem (pc), item (peak).
Error code = MAP_USER_WARN_NOFIND
warning: map_rem_arr: No values exist for this time (99999)
 for subsystem (ps), item (ped).
Error code = MAP_USER_WARN_NOFIND
warning: map_rem_arr: No values exist for this time (99999)
 for subsystem (ps), item (tdc_w).
Error code = MAP_USER_WARN_NOFIND
warning: map_rem_arr: No values exist for this time (99999)
 for subsystem (ps), item (peak).
Error code = MAP_USER_WARN_NOFIND
warning: map_rem_arr: No values exist for this time (99999)
 for subsystem (ps), item (Ethresh).
Error code = MAP_USER_WARN_NOFIND
warning: map_rem_arr: No values exist for this time (99999)
 for subsystem (ps), item (walk).
Error code = MAP_USER_WARN_NOFIND
warning: map_rem_arr: No values exist for this time (99999)
 for subsystem (tac), item (ped).
Error code = MAP_USER_WARN_NOFIND
warning: map_rem_arr: No values exist for this time (99999)
 for subsystem (tac), item (tdc_w).
Error code = MAP_USER_WARN_NOFIND
warning: map_rem_arr: No values exist for this time (99999)
 for subsystem (tac), item (peak).
Error code = MAP_USER_WARN_NOFIND
warning: map_rem_arr: No values exist for this time (99999)
 for subsystem (tac), item (Escale).
 Reading tag2tof constant from file ?
tag2tof.new.99999
Error code = MAP_USER_WARN_NOFIND
warning: map_rem_arr: No values exist for this time (99999)
 for subsystem (tag2tof), item (value).
```

One doesn't have to worry about the warning messages, they come from the fact that the program tries to erase constants that might have been put at this entry previously.

4 Other results

Some other particularities of the tagger behaviour must be pointed out.

4.1 Tagger to RF drift with intensity

A drift in the tagger to RF alignment has been noticed betwen normalisation and production runs (of the order of 50 ps). This variation seems to be related to the beam intensity: normalization run typically have 0.1 nA beam intensity, production run have 8 nA for G1 running period, up to 40 nA for G6 running period.

This variation is shown on fig 29.

If one use to process a production run, the C_i calibration constants obtained with a normalization run, a shift in the alignment between T counters and the RF appears.

Yet calibration obtained for production run are very stable over periods of several days.

This drift is therefore suspected to be related to the variations in the counting rate of the T counters and could be explained for exemple by a lowering of the CFD base ground with the intensity.

After recalibration with a production run, the new constants applied to a normalization run show a reversed effect.

As we underlined several times already, production runs have to be precisely calibrated, this can only be done with a production run. If the tagger constants are calibrated using a normalisation run, they have to be recalibrated straigh away with the first production run that follows this normalisation run.

4.2 Contamination from other hall at low intensity (normalisation run)

On fig 30-a and -b, one can observe, for a properly calibrated normalisation run, that a non-negligible number of hits are not centered on 0, and the pattern is present for all counters. A projection along the Y axis (time axis) shows that the remaining hits distribution is centered on -0.682 ±0.002 ns. These events come from electrons that are timed with bucket that should normally end up in Hall A. This contamination of incorrectly steered electrons from another hall represents in this case 2% of the total number of events. During the G6 period, machine optic problems occurred that lead sometimes to a 30 % contamination (during low intensity runs).

When hall B is running at very low intensity (100 pA) while the other halls are running at high intensity (100 μA), this problem can hardly be avoided. A 10^{-8} halo, one centimeter aside the beam (and therefore incorrectly steered to the wrong hall at the level of the RF separator), which is already a great performance of the machine beam steering, when applied to a 100 μA in the other halls gives a 1 pA current that will be visible in hall B (1 % of hall B current).

Being able to see this contamination gives us confidence in the tagger timing performance and the calibration procedure.

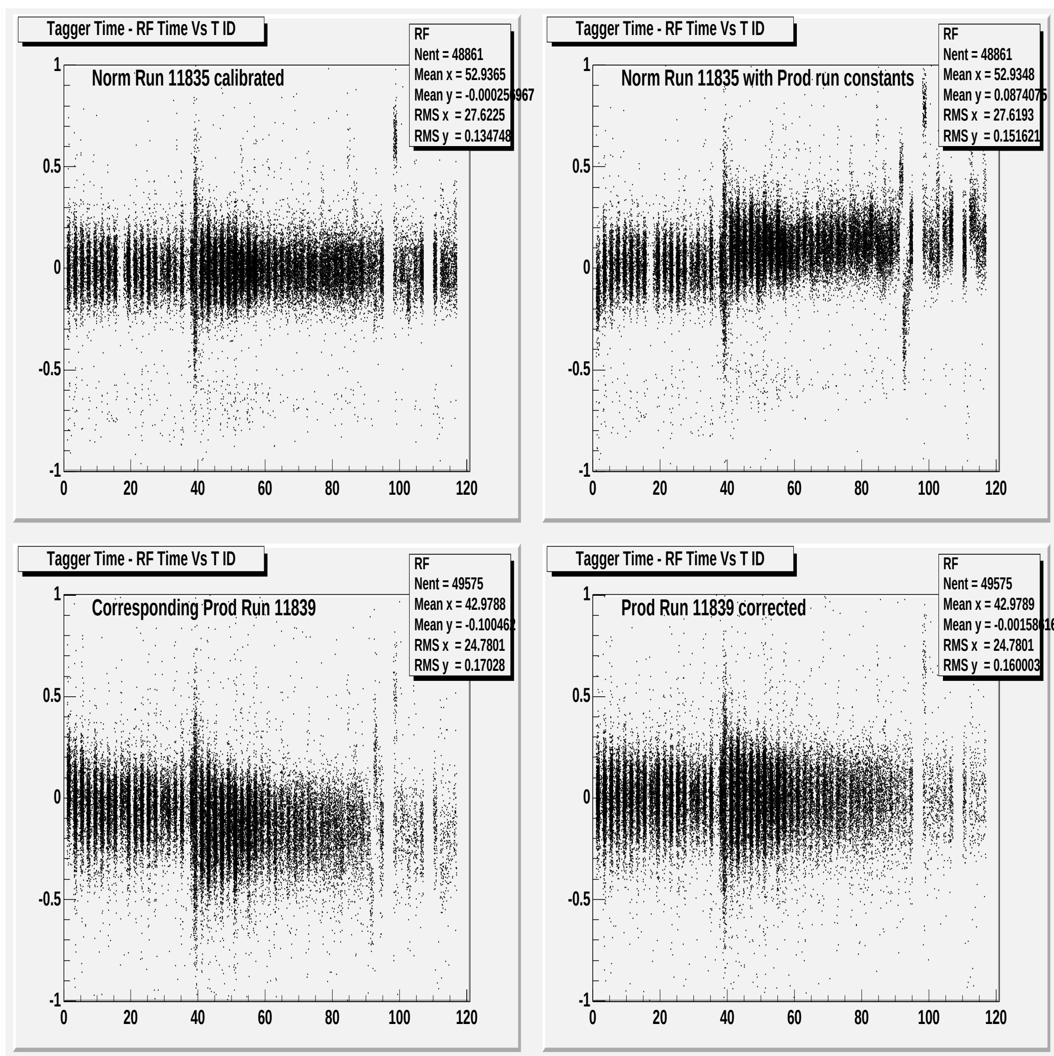

Figure 29: Calibration variation between low and high intensity run

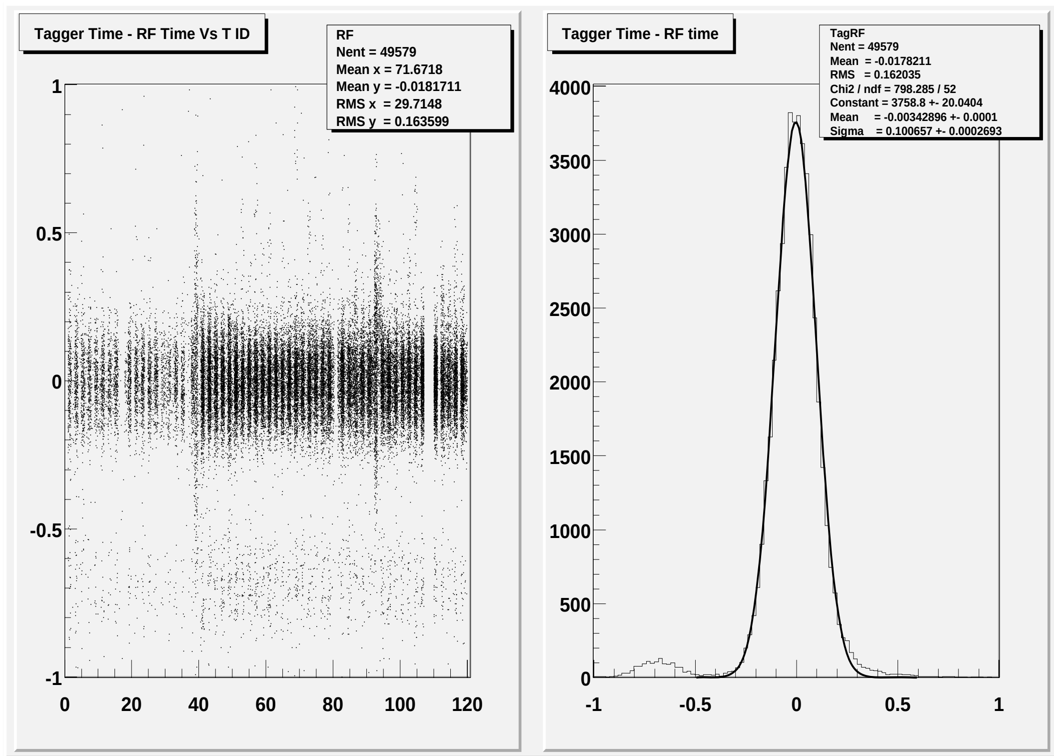

Figure 30: Hall A electron seen by the Tagger

Because those contaminating electrons are not on the normal beam trajectory (halo electrons) their emittance is very poor, the angular aperture of the corresponding tagged photon larger than the regular beam, they are therefore likely not to be able to make it to the Total absorption counter. This halo will therefore biased the measurement of the tagger efficiency during normalisation run, and artificially lower it. For production run, this contamination becomes negligible. But one must keep an eye on this effect, because if such a contamination occurs during a production run it will induce timing error when the tagger time will be replaced by the corresponding RF time in the tagger package analysis, introducing for some events a 0.68 ns offset of the vertex time.

The distribution on fig 30-b includes all the T counters. The mean value obtain after fit is 0.0034 $\pm$0.0001 ns, with a width (FWHM) of 0.283 ns.

4.3 Base peak distribution

As we show on fig 31 for normalization run 11774, the E peak positions can show differences up to 10 nanoseconds from one counter to the others which justifies the fact that those peak positions have to be measured and subtracted to get

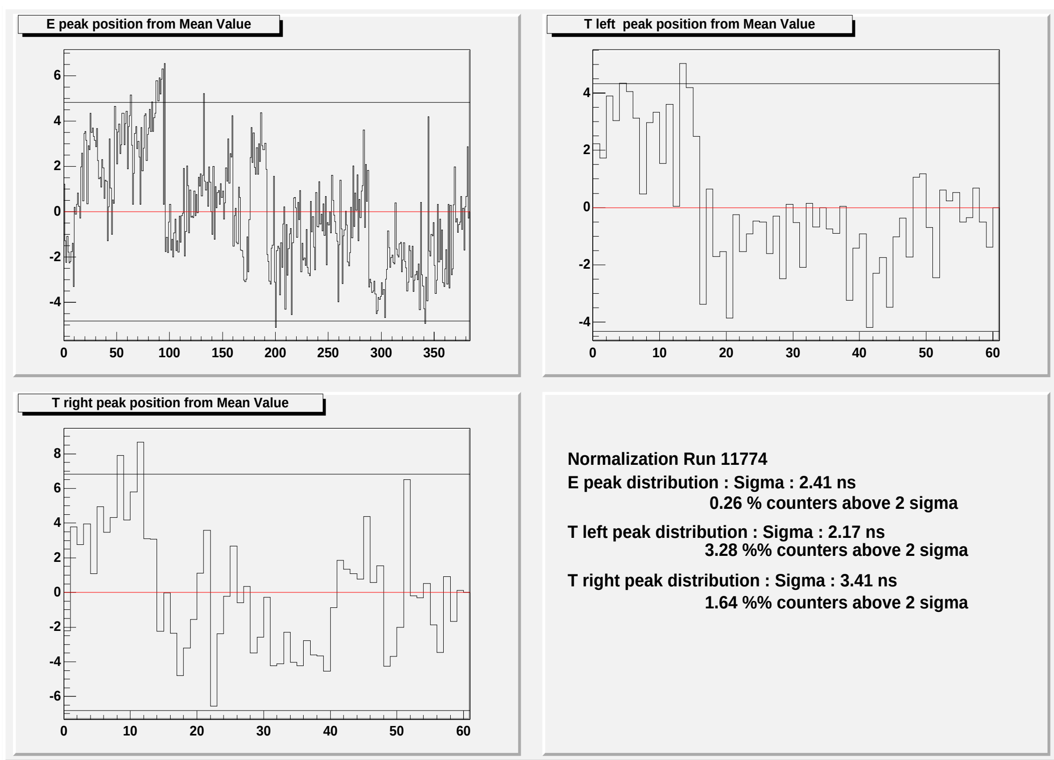

Figure 31: Distance from mean value for E and T peak positions

better E-T time matching. Variation of the peak position for the T counters are also shown. One can clearly see the difference in the electronic setup between the first 19 T counters that are used by the rad-phi experiment and the rest of the T counters.

5 Conclusion

The new calibration software has been used for G1a period. The tagger calibrations are stable over a period of time of several days (more than 200 runs). The main reason for recalibrating the tagger has always been changes in the hardware setup (TDC modules replacement, High voltage changes, cabling modifications.)

Is is recommended to use, for production runs processing, calibration constants obtained with a production run. On the other hand it can be useful to calibrate normalisation runs in order to fully study and understand some aspect of the normalisation.

Now that a fast and efficient software is available for the tagger calibration, a reasonable scheme could be to use the natural frequency of the normalisation runs as a support to tagger calibrations updates, that is to say, to recalibrate the tagger for each normalisation runs, and each following first production runs.

Of course, the tagger also has to be recalibrated each time some hardware setup are modified, for example when cables are changed, electronic modules changed or PMT high voltage modified. In the following table is summarized at which step calibration constants are used, and in what case they should be recalibrated:

Set of constant	Used for	Must be recalibrated if
TDC slopes	ALL	TDC module are changed
Base peaks	E-T time matching, E rebinning	Any change in hardware, TDC slopes recalibration
C_i	T rebinning, output Bank	Any hardware change, any recalibration of slopes or base peak.
offset to CLAS	(not used internally) output bank	all the above

List of Tables

List of Figures