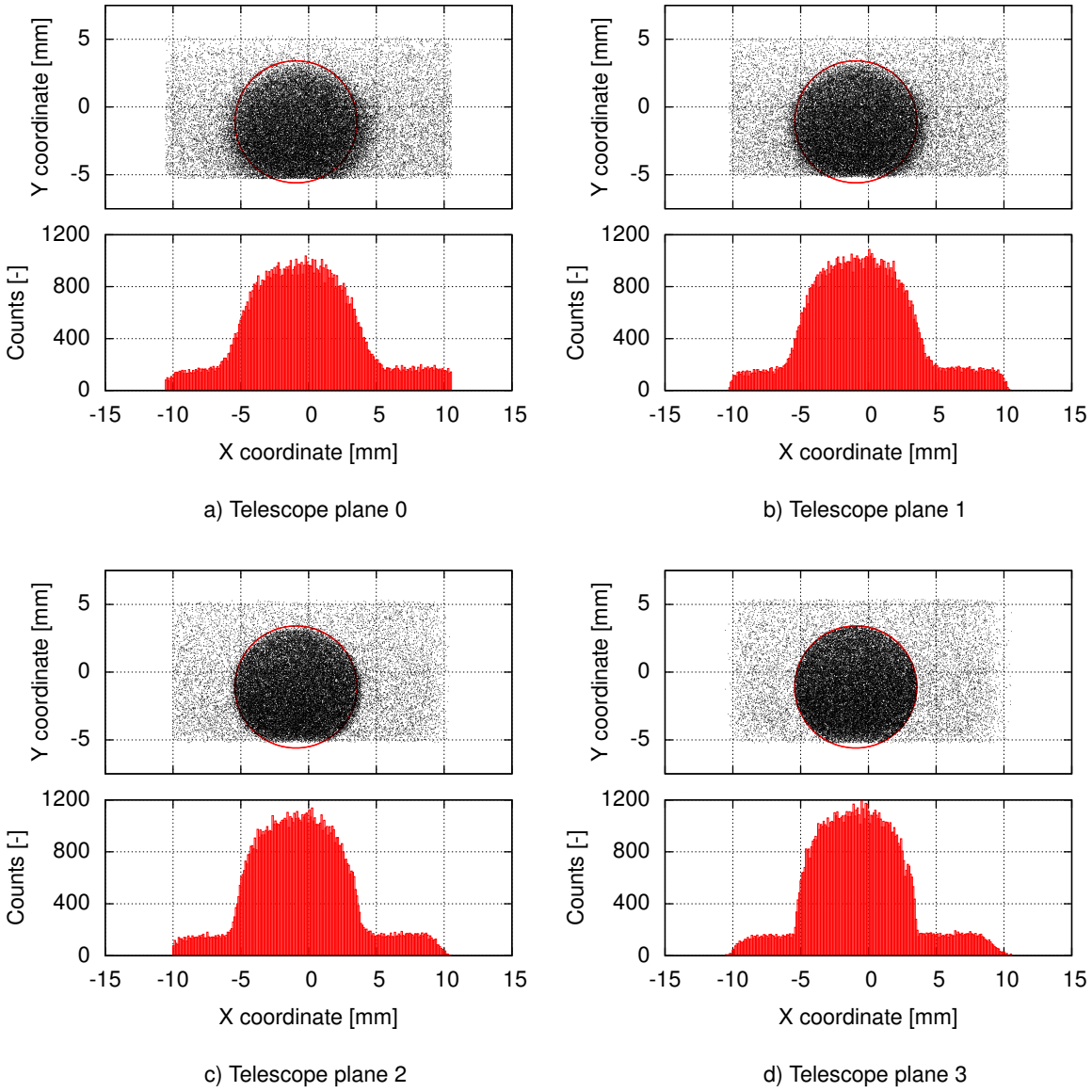
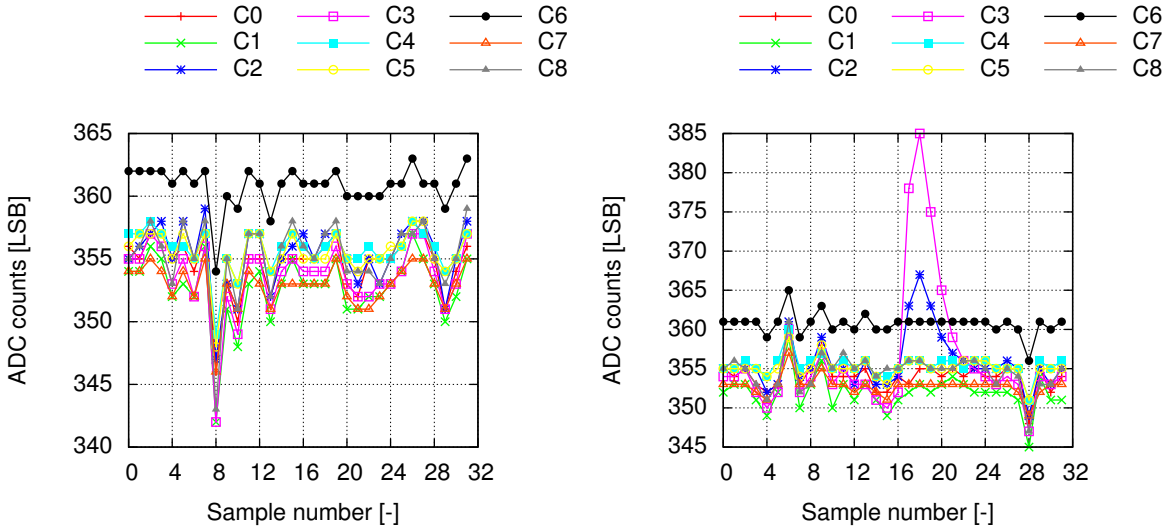


Figure 2.22: Maps of the tracks telescope hits (after data processing, top) together with X axis histograms (bottom). The red circle represents the hole in the anti-coincidence scintillator.

cussed already in the previous section, can be clearly seen in Figure 2.22d. The average number of tracks passing the hole was estimated above 95 %, in good agreement with the expected scintillation efficiency. The reconstructed telescope tracks were afterwards used for the position reconstruction analysis, presented in section 2.3.3.

2.2.2 Time and amplitude reconstruction

Since the beams provided by the PS and DESY II accelerators are continuous (i.e. there is no clock associated with the incoming particles), the LumiCal readout module utilizes an asynchronous sampling mode, as it was described in sections 1.4.1 and 1.4.2. However, one has to note that the example of continuous, asynchronous sampling of the front-end response, shown in Figure 1.22,



a) Example of event without signal

b) Example of event with signal (channels C2 and C3)

Figure 2.23: Example of a raw LumiCal data. Only nine channels (C0–C8) are presented for clarity.

is very simplified and does not include various aspects affecting the readout response. Contrary to this idealized image, an example of real raw data collected during the testbeam is presented in Figure 2.23. For clarity, only nine channels C0–C8 are shown. Since the expected event rate was rather low, 32 ADC samples per event were collected in order to boost the data processing. The two major problems can be clearly seen in Figure 2.23 - the front-end output constant value, so-called pedestal or baseline, varies between channels, mainly due to the spread of the front-end parameters, typical for sub-micron technologies. Especially the channel C6 pedestal differs significantly from the others in the given example. However, this problem can be easily resolved by the pedestal removal procedure, described in details below.

The second, even more important problem, is connected to common mode disturbances. Apart from the pedestal spread, the baseline of each channel should, in ideal case, vary in time only due to the noise generated in the readout chain. Since the noise generated in different channels is uncorrelated, also the baseline variations in different channels should be uncorrelated. However, even without detailed analysis, a strong correlations between channel baselines can be clearly seen in the presented example, e.g. on the 8th sample in Figure 2.23a, or between samples 3rd and 16th in Figure 2.23b. A common mode variations are mostly caused by disturbances and fluctuations of some parameters, like power supply voltage, affecting the baselines in separate channels in the same way. To illustrate the scale of this problem in the given LumiCal readout system, one can compare the amplitude of common mode disturbance at sample 6 in Figure 2.23b, of around 10 LSB, to the signal amplitude at channel C2, slightly lower than 20 LSB. Since the disturbance reaches above 50 % of the signal amplitude, an efficient CMS procedure is required.

Pedestal removal and Common Mode Subtraction (CMS)

Since the front-end baseline value depends on various parameters, e.g. the ASIC temperature, it can vary in a larger time interval depending on changes of these parameters. The chosen record of 32 ADC samples per event allows calculation of the pedestal value in each event independently. As can

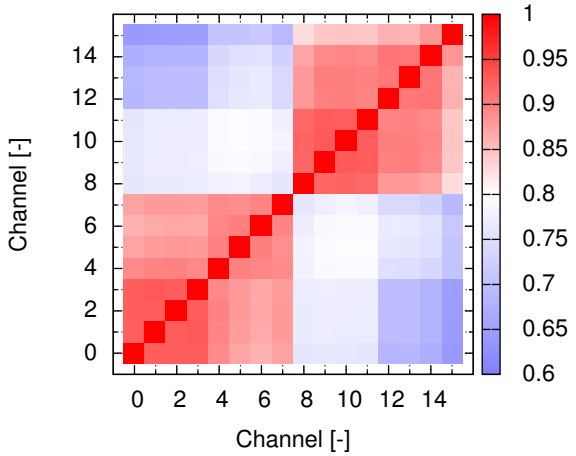
be seen from Figure 2.23b, the trigger position in the event was set to around 17th sample, and so the pedestal P_n^k of channel k in event n can be calculated as an average of the first fourteen samples as follows:

$$P_n^k = \frac{1}{14}\sum_{i=0}^{13} S_{i,n}^k \,, \tag{2.10}$$

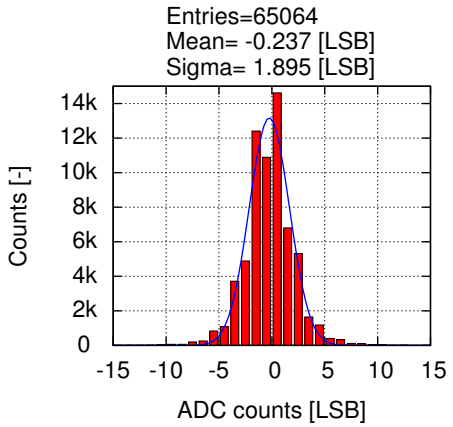
where $S_{i,n}^k$ is sample i from event n in channel k. The average pedestal value is then subtracted from all the samples $(S_{0,n}^k, S_{1,n}^k, ..., S_{31,n}^k)$:

$$\overline{S_{i,n}^k} = S_{i,n}^k - P_n^k \qquad \forall_{i\in(0,1,..31)} \,, \tag{2.11}$$

resulting in the events comprising data with baselines equalized to zero.



a) Correlation coefficients for baseline samples for 16 channels



b) Exemplary baseline histogram of arbitrary chosen channel 9

Figure 2.24: Correlation coefficients for baseline samples for 16 channel and an exemplary baseline histogram for one channel for raw data after pedestal removal.

In order to estimate the correlation between different channel baselines, a Pearson product-moment correlation coefficient [72, 73] was calculated. An exemplary map of these coefficients calculated for 16 channels (two ASICs) is presented in Figure 2.24a. A very clearly distinguishable areas, related to two separate 8-channel ASICs, can be easily found. Within each ASIC the coefficients remain above 0.8 indicating a very strong correlations between baseline samples. The correlations between channels from separate ASICs are weaker, but still quite high since the coefficients remain above 0.6. Within each ASIC a two separate regions, related to two groups of four channels with different gain, can also be found. The channels with the same gain are clearly more correlated to each other than to the ones with different gain. A histogram presenting the baseline variations after pedestal subtraction is presented in Figure 2.24b. The sigma of Gaussian fit, slightly below two Least Significant Bits (LSBs), was obtained in this case. A proper CMS procedure should decrease this value even more.

Contrary to the pedestal removal procedure, where the mean pedestal is calculated as an average over time (sample number i) for fixed channel k, the mean common mode CM_n^i in event n is an

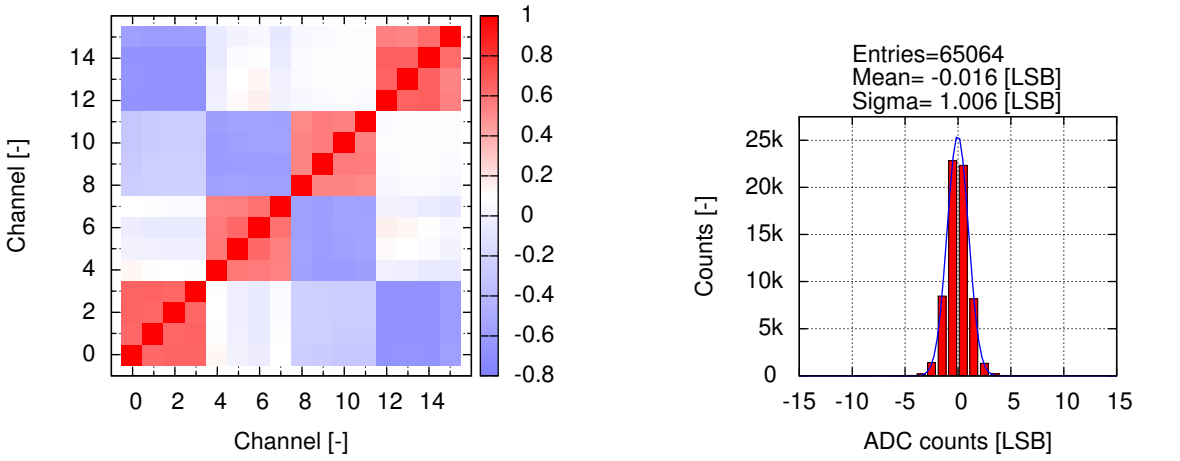
average over channels for fixed sample number i, as follows:

$$CM_n^i = \frac{1}{(k_e - k_s)} \sum_{k=k_s}^{k_e} S_{i,n}^k \ . \tag{2.12}$$

Analogously to the pedestal removal procedure, the mean common mode is subtracted from sample i in each channel k in range (k_s, k_e):

$$\widetilde{S_{i,n}^k} = \overline{S_{i,n}^k} - CM_n^i \qquad \forall_{k\in(k_s,\ k_s+1,\ldots,\ k_e)} \ . \tag{2.13}$$

Since the common mode subtraction procedure increases the statistical (uncorrelated) noise to $\widetilde{\sigma}$ according to $\widetilde{\sigma}^2 = \sigma^2 + \sigma^2/K$, where σ is noise Root Mean Square (RMS) and $K = (k_e - k_s)$ is number of channels taken to estimate the mean common mode, one should expect that higher K value should improve the procedure performance. It is true, assuming that all channels behave in similar way, which is not true for the analyzed data, for which different patterns exist for different groups of channels, as seen in Figure 2.24a.



a) Correlation coefficients for baseline samples for 16 channels

b) Exemplary baseline histogram of arbitrary chosen channel 9

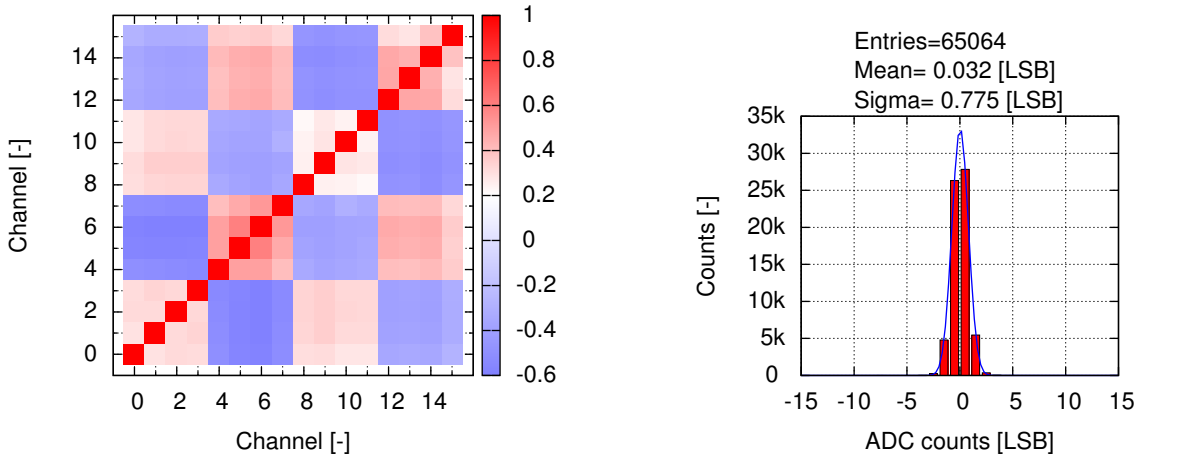
Figure 2.25: Correlation coefficients for baseline samples for 16 channel and an exemplary baseline histogram for one channel for mean common mode calculated for all 32 channels altogether.

In order to verify the influence of the range (k_s, k_e) of channels taken for the mean common mode CM_n^i calculation, three cases are considered. In the first one, a single value of mean common mode is calculated for all 32 channels, i.e. $(k_s, k_e) = (0, 31)$, resulting in highest possible $K = 32$. The result of this type common mode subtraction is shown in Figure 2.25. The sigma of the baseline histogram, presented in Figure 2.25b, is almost two times smaller then the one obtained for the data just after pedestal removal procedure. However, a correlation coefficients map, shown in Figure 2.25a, indicates that this procedure is not fully efficient, since a very strong positive correlation (with coefficient above 0.6) can still be found within groups of four equal-gain channels, while a strong negative correlations (coefficient below -0.5) are introduced between different ASICs, especially for channels with different gains. This can be understood since these groups were initially weakly correlated (see Figure 2.24a), therefore the subtraction of the same mean common mode value introduces a negative correlation.

To avoid introducing of negative correlations between separate ASICs, a second case of CMS procedure can be considered. In this case, the mean common mode is calculated independently for each ASIC, i.e. four mean common mode values CM_n^i are calculated in channels ranges:

$$(k_s, k_e) \in [(0,7);\ (8,15);\ (16,23);\ (24,31)]\ . \tag{2.14}$$

The results of this CMS procedure are presented in Figure 2.26. The sigma of the baseline histogram (Figure 2.26b) is reduced compared to the first case, and it is 2.5 times smaller than the initial one. The correlations between channels with the same gain, even in separate ASICs, were significantly reduced, with the coefficients below 0.3, although a strong negative correlations were introduced within each ASIC between channels with different gain.



a) Correlation coefficients for baseline samples for 16 channels

b) Exemplary baseline histogram of arbitrary chosen channel 9

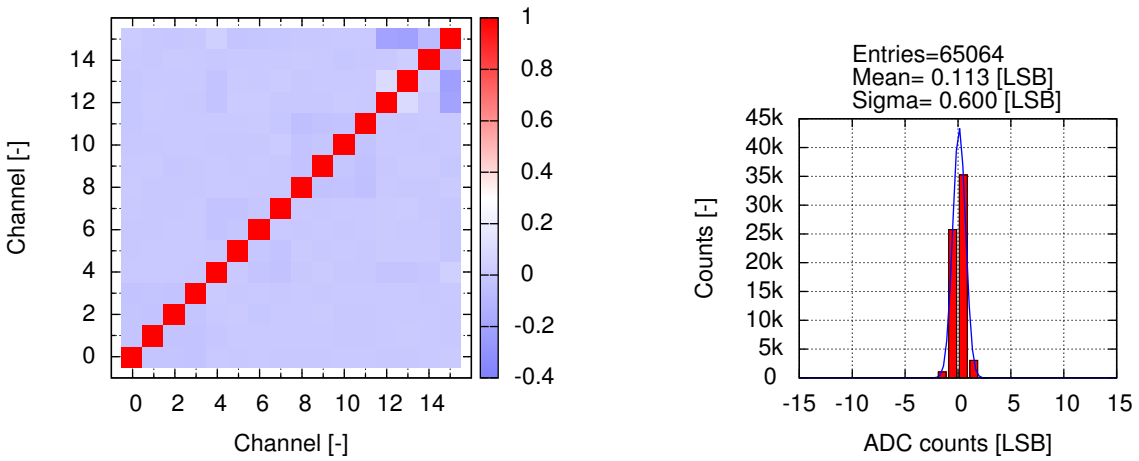
Figure 2.26: Correlation coefficients for baseline samples for 16 channel and an exemplary baseline histogram for one channel for mean common mode calculated for each ASIC independently (4 groups of 8 channels each).

From the above results, as well as from the initial correlation map (Figure 2.24a), the third CMS case can be inferred, with the mean common mode calculated for each group of equal-gain channels independently, i.e. eight CM_n^i values calculated in channels ranges:

$$(k_s, k_e) \in [(0,3);\ (4,7);\ (8,11);\ (12,15);\ (16,19);\ (20,23);\ (24,27);\ (28,31)]\ . \tag{2.15}$$

In this case $K = 4$, so the CMS procedure is expected to increase the uncorrelated noise level to $\tilde{\sigma}^2 = 1.25\sigma^2$. Since the uncorrelated noise level is significantly lower than the common mode disturbances, therefore a better CMS performance may be more important than the uncorrelated noise level increase. The results obtained for this case are presented in Figure 2.27. The resulting coefficients map (Figure 2.27a) shows a uniform pattern of a weak negative correlations. The sigma of baseline histogram (Figure 2.27b), equals 0.6 LSB, is more than 3 times lower then the initial one, and it is the smallest of all hitherto obtained. The results indicate that, despite of very low statistics (only 4 channels) and the uncorrelated noise level increase, this method provides the best common mode rejection capability, therefore it is used as a default CMS procedure for the LumiCal data processing.

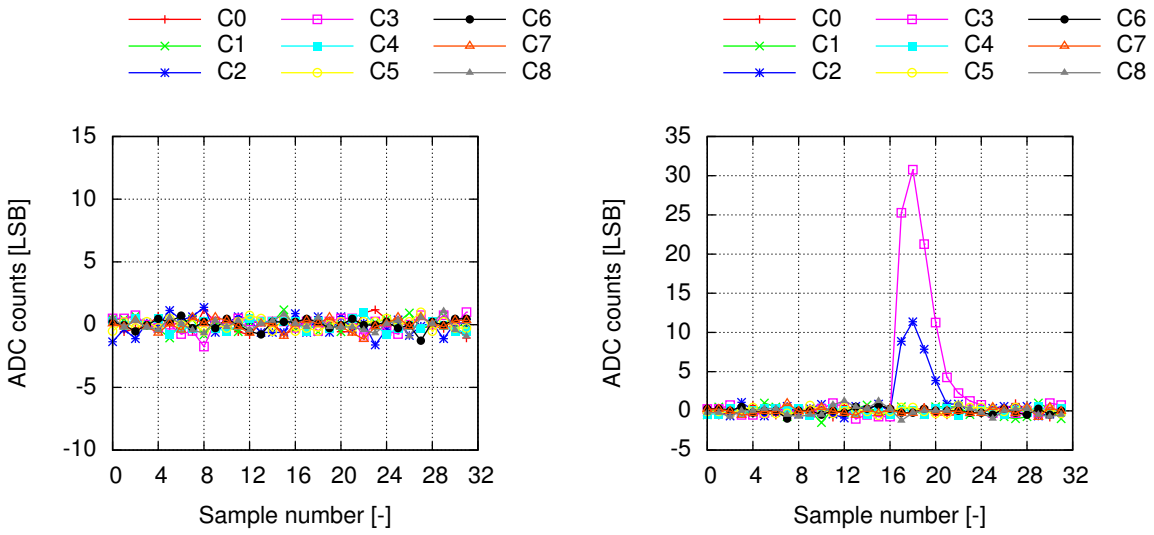
In order to illustrate the pedestal removal and CMS procedure results, an exemplary raw data, presented in Figure 2.23, is shown in Figure 2.28 after initial processing.



a) Correlation coefficients for baseline samples for 16 channels

b) Exemplary baseline histogram of arbitrary chosen channel 9

Figure 2.27: Correlation coefficients for baseline samples for 16 channel and an exemplary baseline histogram for one channel for mean common mode calculated for each equal-gain channels group in each ASIC independently (8 groups of 4 channels each).



a) Example of event without signal

b) Example of event with signal (channels C2 and C3)

Figure 2.28: Examples of a LumiCal data after pedestal removal and CMS procedures. Only nine channels (C0–C8) are presented for clarity.

Deconvolution method

The initial data processing, described in previous section, prepares the LumiCal data for the main part of signal reconstruction. As it was already described in Section 1.4.1, the charge deposited in active sensor volume is directly proportional to the front-end pulse amplitude. Since the LumiCal readout utilizes an asynchronous ADC sampling scheme, the pulse amplitude is not directly available from the initially processed data and, therefore, has to be reconstructed. The simplest reconstruction

algorithm is based on a theoretical pulse shape fitting, e.g. using the least squares method. The time response of the complete front-end chain, for a general $CR-(RC)^n$ shaping, is given by formula 1.23. The particular LumiCal front-end utilizes a simple $CR-RC$ shaping, for which the formula 1.23 simplifies to the form:

$$|\mathrm{V}_{\mathrm{front-end}}(t)| = \frac{\mathrm{q_{in}}}{\mathrm{C_{feed}}}\left(\frac{t}{\tau_{\mathrm{sh}}}\right)e^{-\frac{t}{\tau_{\mathrm{sh}}}} , \tag{2.16}$$

with the maximum value given as:

$$\mathrm{V}^{\mathrm{max}}_{\mathrm{front-end}} = \mathrm{V}_{\mathrm{front-end}}(T_{peak}) = \frac{\mathrm{q_{in}}}{\mathrm{C_{feed}}}\cdot\frac{1}{e} . \tag{2.17}$$

In such case, a function $p(t)$, given by the formula:

$$p(t) = \begin{cases} b & \text{for } t < t_0 \\ \alpha\left(\frac{t-t_0}{\tau_{\mathrm{sh}}}\right)e^{-\frac{t-t_0}{\tau_{\mathrm{sh}}}} + b & \text{for } t \geq t_0 , \end{cases} \tag{2.18}$$

can be fitted to the each pulse via α, b, t_0 parameters in order to reconstruct the pedestal b, pulse start time t_0 and pulse amplitude $A = (\alpha/e)$. However, the pulse fitting is a rather slow procedure which requires an initial guess of the fit parameters set, and due to the required complex calculations can be done only during the offline computer analysis. In addition, its complexity would be even higher if pile-up effects were included.

In order to enable a fast amplitude reconstruction, a deconvolution method, proposed for pulse processing in HEP experiments at the beginning of 90's [74], was proposed for the LumiCal readout. The great advantage of this method is the possibility to implement it inside the system DSP block, reducing significantly the total amount of data transmitted from the detector. Since the output pulse of the front-end electronics is a convolution of its pulse response with the sensor current signal (and so, deposited charge), an inverse procedure, so-called deconvolution, can be performed in order to reconstruct the input signal. Initially the deconvolution method was implemented in early 90's in Analog Pipeline Voltage (APV) ASIC for the CMS experiment at LHC, where it was performed by an analogue pulse shape processor [75]. Nowadays, with the ADC integrated in each readout channel, the deconvolution can be performed in digital manner using digital filters.

The digital filters are very common blocks of modern DSP systems. Similarly to their analogue counterparts, they attenuate or amplify certain bands of frequencies, although they operate on discrete-time, discrete-value samples instead of continuous ones, used in analogue filters. The most common architectures of digital filters are Finite Impulse Response (FIR) filter, Infinite Impulse Response (IIR) filter and an Adaptive Digital Filter [76]. The first one has the simplest architecture, performing a simple weighted average (convolution) of the series of N data samples. The IIR has a recursive linear architecture, with each output sample depending on both the series of N data samples and the previous output samples. In other words, the IIR architecture comprises a feedback circuit and, therefore, this architecture is most similar to the analogue filters. The Adaptive Digital Filter has a most sophisticated architecture which can learn and adapts itself to a desired signal.

The output sample s_k of the simplest FIR architecture filter can be expressed, in general, as:

$$s_k = \sum_{i=0}^{N-1} w_i v_{k-i} , \tag{2.19}$$

where w_i represents weight associated to input sample v_{k-i}. The filter order depends on the length N of the series of data samples, therefore a higher order filter requires a more processing power

for its implementation. Since the deconvolution filter has to represent the inverse front-end pulse response function, a higher $CR-(RC)^n$ shaping utilized in the shaper induces a higher order of FIR filter. Therefore, a simple $CR-RC$ shaping was implemented in the LumiCal front-end electronics in order to reduce the complexity of the deconvolution filter. Since the transfer function $H(s)$ of the $CR-RC$ shaper can be derived from 1.20 by putting $n=1$, the front-end response $V_{sh}(s)$ in a Laplace domain can be expressed as:

$$V_{sh}(s) = \frac{1}{s}\cdot H(s) = \frac{\tau_{\mathrm{sh}}}{(1+s\tau_{\mathrm{sh}})^2} = \frac{1}{\tau_{\mathrm{sh}}}\cdot\frac{1}{\left(s+\frac{1}{\tau_{\mathrm{sh}}}\right)^2}\ . \tag{2.20}$$

Aside the constant $(1/\tau_{\mathrm{sh}})$ factor, the deconvolution filter transfer function $D(s)$ is therefore given by:

$$D(s) = \frac{1}{V_{sh}(s)} \cong \left(s+\frac{1}{\tau_{\mathrm{sh}}}\right)^2 = (s-s_0)^2\ . \tag{2.21}$$

Such kind of transfer function, comprising only a double zero and no poles, cannot be easily implemented in analogue circuitry, although, it may be easily achieved in discrete-time domain. To transform the $D(s)$ from a continuous domain s into a discrete domain z, a Z transform, utilizing a pole-zero mapping technique, can be used. In such case, each S-plane pole (or zero) s_0 is replaced by z_0 in Z-plane according to:

$$z_0 = e^{s_0 T_{smp}}\ , \tag{2.22}$$

where T_{smp} is sampling period. The formula 2.21, comprising double zero $s_0 = -1/\tau_{\mathrm{sh}}$, can be transformed in $D(z)$ as follows:

$$D(z) = (z-z_0)^2 = \left(z-e^{-\frac{T_{smp}}{\tau_{\mathrm{sh}}}}\right)^2 = z^2 - 2e^{-\frac{T_{smp}}{\tau_{\mathrm{sh}}}}z + e^{-\frac{2T_{smp}}{\tau_{\mathrm{sh}}}}\ . \tag{2.23}$$

Since z^2 represents the sample which will be received after two sampling periods in the future, equation 2.23 can be multiplied by z^{-2}, what corresponds to delaying all the samples by two periods, resulting in the final equation:

$$D(z) = 1 - 2e^{-\frac{T_{smp}}{\tau_{\mathrm{sh}}}}z^{-1} + e^{-\frac{2T_{smp}}{\tau_{\mathrm{sh}}}}z^{-2}\ , \tag{2.24}$$

where z^{-1} is a unit delay corresponding to the sample taken one sampling period before the current one. The output sample value d_i, obtained at time $i\cdot T_{smp}$, can be expressed from 2.24 as:

$$d_i = v_i - 2e^{-\frac{T_{smp}}{\tau_{\mathrm{sh}}}}v_{i-1} + e^{-\frac{2T_{smp}}{\tau_{\mathrm{sh}}}}v_{i-2}\ , \tag{2.25}$$

where v_i is the shaper output sample taken at time $i\cdot T_{smp}$, i.e.:

$$v_i = V_{sh}(i\cdot T_{smp})\ . \tag{2.26}$$

From the comparison of formulas 2.19 and 2.25 it can be seen that the deconvolution filter is a second order FIR filter with the weights given as:

$$w_0 = 1\ , \qquad w_1 = -2e^{-\frac{T_{smp}}{\tau_{\mathrm{sh}}}}\ , \qquad w_2 = e^{-\frac{2T_{smp}}{\tau_{\mathrm{sh}}}}\ . \tag{2.27}$$

For the chosen $CR-RC$ shaping the deconvolution filter requires only two multiplications and two additions. The weights depend only on the ratio between sampling period T_{smp} and shaper time constant τ_{sh}, which for a given readout is constant. Therefore, the deconvolution can be performed

online as a part of the DAQ system, either by the on-board FPGA or even by the System-on-Chip (SoC) readout ASIC.

Based on formula 2.25, the deconvolution filter response to the front-end pulse given by equation 2.18 can be calculated. In order to simplify the equations, the pulse pedestal b can be set to zero. Since the pedestal removal procedure is performed as an initial data processing, this assumption is well-motivated and does not result in loss of generality. One has to note, that, in general, the pulse starting time t_0 has to be located within $[0, T_{smp})$ range. Since the whole sampling-filtering process is periodical, the pulse starting time lower than zero, or $\geq T_{smp}$, corresponds to the case of the pulse beginning one sample earlier or later than the considered one, respectively. In other words, the pulse starting time equal to $t_0 + k \cdot T_{smp}$ can be treated as a pulse beginning at t_0 in reference to the sample k. Based on these assumptions, the consecutive shaper output samples can be expressed as:

$$\begin{aligned}
v_{-1} &= 0 \\
v_0 &= 0 \\
v_1 &= Ae\frac{T_{smp}-t_0}{\tau_{\mathrm{sh}}}e^{-\frac{T_{smp}-t_0}{\tau_{\mathrm{sh}}}} \\
v_2 &= Ae\frac{2T_{smp}-t_0}{\tau_{\mathrm{sh}}}e^{-\frac{2T_{smp}-t_0}{\tau_{\mathrm{sh}}}} \\
&\ldots \\
v_k &= Ae\frac{kT_{smp}-t_0}{\tau_{\mathrm{sh}}}e^{-\frac{kT_{smp}-t_0}{\tau_{\mathrm{sh}}}} \\
&\ldots ,
\end{aligned} \tag{2.28}$$

resulting in the consecutive deconvolution filter samples given by:

$$\begin{aligned}
d_{-1} &= w_{-1}v_{-1} + w_{-2}v_{-2} + w_{-3}v_{-3} = 0 \\
d_0 &= w_0v_0 + w_{-1}v_{-1} + w_{-2}v_{-2} = 0 \\
d_1 &= w_1v_1 + w_0v_0 + w_{-1}v_{-1} = Ae\frac{T_{smp}-t_0}{\tau_{\mathrm{sh}}}e^{-\frac{T_{smp}-t_0}{\tau_{\mathrm{sh}}}} \\
d_2 &= w_2v_2 + w_1v_1 + w_0v_0 = Ae\frac{2T_{smp}-t_0}{\tau_{\mathrm{sh}}}e^{-\frac{2T_{smp}-t_0}{\tau_{\mathrm{sh}}}} - \\
&- \left[2e^{-\frac{T_{smp}}{\tau_{\mathrm{sh}}}}\right]\cdot\left[Ae\frac{T_{smp}-t_0}{\tau_{\mathrm{sh}}}e^{-\frac{T_{smp}-t_0}{\tau_{\mathrm{sh}}}}\right] = Ae\frac{t_0}{\tau_{\mathrm{sh}}}e^{-\frac{2T_{smp}-t_0}{\tau_{\mathrm{sh}}}} \\
d_k\Big|_{k>2} &= w_kv_k + w_{k-1}v_{k-1} + w_{k-2}v_{k-2} = Ae\frac{kT_{smp}-t_0}{\tau_{\mathrm{sh}}}e^{-\frac{kT_{smp}-t_0}{\tau_{\mathrm{sh}}}} - \\
&- \left[2e^{-\frac{T_{smp}}{\tau_{\mathrm{sh}}}}\right]\cdot\left[Ae\frac{(k-1)T_{smp}-t_0}{\tau_{\mathrm{sh}}}e^{-\frac{(k-1)T_{smp}-t_0}{\tau_{\mathrm{sh}}}}\right] + \\
&+ \left[e^{-\frac{2T_{smp}}{\tau_{\mathrm{sh}}}}\right]\cdot\left[Ae\frac{(k-2)T_{smp}-t_0}{\tau_{\mathrm{sh}}}e^{-\frac{(k-2)T_{smp}-t_0}{\tau_{\mathrm{sh}}}}\right] = \\
&= Ae\frac{1}{\tau_{\mathrm{sh}}}e^{-\frac{kT_{smp}-t_0}{\tau_{\mathrm{sh}}}}\left[kT_{smp}-t_0-2(k-1)T_{smp}+2t_0+(k-2)T_{smp}-t_0\right] \equiv 0 .
\end{aligned} \tag{2.29}$$

As can be seen from equations 2.29, the deconvolution filter produces only two non-zero samples d_k for asynchronous sampling of the $CR-RC$ pulse. If all three input samples v_i are located on zeroed baseline, or on the pulse, the filter output sample d_k is always equal to zero. The non-zero

samples d_k are produced only in case where at least one input sample v_i is located on baseline and the remaining one or two are located on the pulse. In the exceptional case of a synchronous sampling only one non-zero output sample d_k is produced. This can be derived from equations 2.29 by setting the $t_0 = 0$ what corresponds to the input pulse synchronized to the sampling clock. In this case, the output samples d_k are given as:

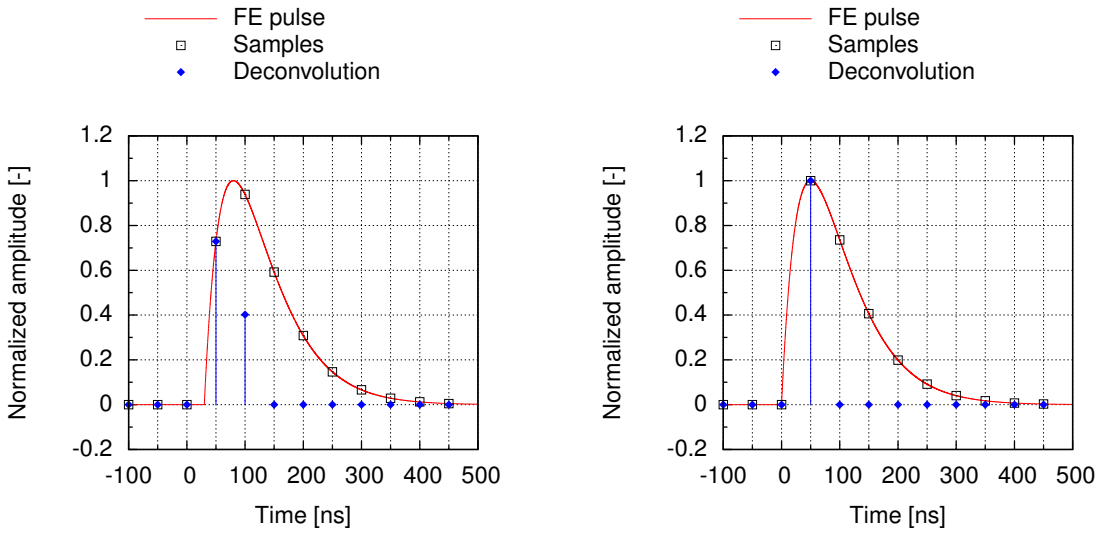
$$\begin{aligned} d_k\Big|_{k<1} &= 0 \\ d_1 &= Ae\frac{T_{smp}-0}{\tau_{\mathrm{sh}}}e^{-\frac{T_{smp}-0}{\tau_{\mathrm{sh}}}} = Ae\frac{T_{smp}}{\tau_{\mathrm{sh}}}e^{-\frac{T_{smp}}{\tau_{\mathrm{sh}}}} \\ d_k\Big|_{k>1} &= 0\,. \end{aligned} \tag{2.30}$$

Some additional simplification can be done in the synchronous case by setting the sampling period exactly equal to the shaper time constant, i.e. $T_{smp} = \tau_{\mathrm{sh}}$. In such case, the one non-zero sample d_1 directly represents the pulse amplitude since:

$$d_1 = Ae\frac{\tau_{\mathrm{sh}}}{\tau_{\mathrm{sh}}}e^{-\frac{\tau_{\mathrm{sh}}}{\tau_{\mathrm{sh}}}} = Ae\cdot e^{-1} = A\,. \tag{2.31}$$

In a more general case of $T_{smp} \neq \tau_{\mathrm{sh}}$ the pulse amplitude can be calculated from equation 2.30 as:

$$A = d_1 \cdot \frac{\tau_{\mathrm{sh}}}{T_{smp}}e^{\frac{T_{smp}-\tau_{\mathrm{sh}}}{\tau_{\mathrm{sh}}}}\,. \tag{2.32}$$



a) Example of asynchronous sampling with two non-zero filter output samples at $t_0 = 30$ ns

b) Example of synchronous sampling with only one non-zero filter output sample at $t_0 = 0$ ns

Figure 2.29: Examples of deconvolution filter output at $T_{smp} = \tau_{\mathrm{sh}} = 50$ ns.

The example responses of the deconvolution filter (blue diamonds) to the sampled (black squares) front-end pulse response (red line) are shown in Figure 2.29 for $T_{smp} = \tau_{\mathrm{sh}} = 50$ ns. In the asynchronous case, with $t_0 = 30$ ns (Figure 2.29a), the two non-zero samples are produced. In the case of synchronous sampling (Figure 2.29b) only one non-zero sample, directly representing the pulse maximum, is produced as described by equations 2.30 and 2.31.

In the asynchronous sampling case, described by equations 2.29, the ratio of the two samples is given by:

$$\frac{d_2}{d_1} = \frac{t_0}{T_{smp} - t_0} e^{-\frac{T_{smp}}{\tau_{\mathrm{sh}}}} , \tag{2.33}$$

enabling the reconstruction of the pulse starting time t_0 for the given T_{smp} and τ_{sh}. From equation 2.33, the reconstructed pulse starting time t_0 can be expressed as:

$$t_0 = \frac{\frac{d_2}{d_1} T_{smp}}{\frac{d_2}{d_1} + e^{-\frac{T_{smp}}{\tau_{\mathrm{sh}}}}} . \tag{2.34}$$

In order to obtain the equation enabling the reconstruction of the pulse amplitude, the sum of two non-zero samples d_1, d_2 can be calculated as follows:

$$d_1 + d_2 = \frac{A}{\tau_{\mathrm{sh}}} e^{-\frac{T_{smp} - t_0 - \tau_{\mathrm{sh}}}{\tau_{\mathrm{sh}}}} \left[T_{smp} - t_0 \left(1 - e^{-\frac{T_{smp}}{\tau_{\mathrm{sh}}}} \right) \right] . \tag{2.35}$$

The pulse amplitude A can be obtained from equation 2.35 as:

$$\begin{aligned} A &= (d_1 + d_2) \frac{\tau_{\mathrm{sh}} e^{\frac{T_{smp} - t_0 - \tau_{\mathrm{sh}}}{\tau_{\mathrm{sh}}}}}{T_{smp} - t_0 \left(1 - e^{-\frac{T_{smp}}{\tau_{\mathrm{sh}}}} \right)} = \\ &= (d_1 + d_2) \left[\frac{\tau_{\mathrm{sh}}}{T_{smp}} e^{\frac{T_{smp} - \tau_{\mathrm{sh}}}{\tau_{\mathrm{sh}}}} \right] \frac{e^{\frac{-t_0}{\tau_{\mathrm{sh}}}}}{1 - \frac{t_0}{T_{smp}} \left(1 - e^{-\frac{T_{smp}}{\tau_{\mathrm{sh}}}} \right)} . \end{aligned} \tag{2.36}$$

As can be seen from the above equation, the pulse amplitude is given by the sum of two samples multiplied by the time dependent correction factor. Apart from sampling period T_{smp} and shaper time constant τ_{sh}, this factor depends on the reconstructed pulse starting time t_0. One should note that in the synchronous sampling case the second output sample $d_2 \equiv 0$ and equation 2.34 provides a properly reconstructed $t_0 = 0$, and, equation 2.36 reduces to 2.32. This shows, that the synchronous sampling does not have to be considered as a separate, special case, since the equations 2.34 and 2.36, derived for the general asynchronous sampling provide a proper results for any t_0 in range $[0, T_{smp})$, covering at the same time the synchronous sampling case $t_0 = 0$.

The time dependent correction factor has to be used in order to enable the precise pulse amplitude reconstruction, as can be seen from equation 2.36. However, the amplitude can be roughly estimated using only the normalized sum of two non-zero samples $\alpha \cdot (d_1 + d_2)$. The normalization coefficient α, resulting from the sampling period to the shaper time constant ratio $T_{smp}/\tau_{\mathrm{sh}}$, is given as:

$$\alpha = \frac{\tau_{\mathrm{sh}}}{T_{smp}} e^{\frac{T_{smp} - \tau_{\mathrm{sh}}}{\tau_{\mathrm{sh}}}} . \tag{2.37}$$

The normalized sum of two non-zero samples $\alpha \cdot (d_1 + d_2)$ (without time dependent correction factor) for the normalized pulse amplitude as a function of normalized pulse starting time t_0/T_{smp}, is shown in Figure 2.30 for various $T_{smp}/\tau_{\mathrm{sh}}$ ratios. For sampling periods shorter than the shaper time constant the maximum discrepancy between the normalized sum of samples and the pulse amplitude A does not exceed few percents. In the particular case of $T_{smp} = \tau_{\mathrm{sh}}$, the maximum deviation slightly exceeds 13 %, and grows with the increasing $T_{smp}/\tau_{\mathrm{sh}}$ ratio. The calculated values of maximum deviation are shown in Table 2.4. The LumiCal readout utilizes the front-end with effective peaking

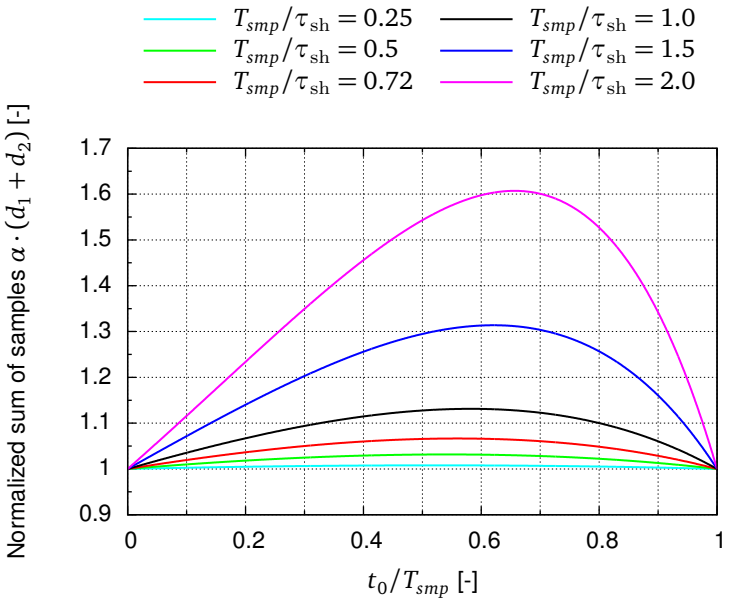


Figure 2.30: Normalized sum of two non-zero samples $\alpha(d_1 + d_2)$ (without time dependent correction factor) for different sampling periods T_{smp}.

T_{smp}/τ_{sh}	Maximum deviation [%]	t_0/T_{smp} of the maximum deviation
0.25	0.78	0.52
0.50	3.16	0.54
0.72	**6.65**	**0.56**
1.00	13.12	0.58
1.50	31.37	0.62
2.00	60.72	0.66

Table 2.4: Maximum deviations between normalized sum of two non-zero samples and the pulse amplitude for different sampling periods.

time slightly below 70 ns and the ADC sampling period of 50 ns. This results in T_{smp}/τ_{sh} ratio of around 0.72, corresponding to the maximum deviation below 7 %.

The above considerations become important when a real pulse, comprising noise and common mode disturbances, is being deconvoluted. In the ideal case, shown in Figure 2.29, all baseline samples are equal, since the noise contribution was not considered. Assuming, for simplicity, that the noise introduced by the front-end electronics has an infinite bandwidth, the pulse starting time t_0, reconstructed using equation 2.34, can obtain any value in range $(-\infty, +\infty)$. For the band-limited noise, together with finite front-end and ADC dynamic ranges, the possible pulse starting time t_0 range, obtained for the noise samples, is also limited, although much wider than the theoretical one ($t_0 \in [0, T_{smp}]$). If only the amplitude of front-end pulse is of interest, the pulse starting time t_0, calculated for any pair of two samples, can stand as initial criterion for the front-end pulse discrimination (i.e. only the pairs which result in $t_0 \in [0, T_{smp}]$ are further processed). However, in order to investigate the SNR after deconvolution process, the noise level has to be somehow estimated. Since the time dependent correction factor (equation 2.36) cannot be used for $t_0 \notin [0, T_{smp}]$, the normalized sum of samples $\alpha(d_1 + d_2)$ can be taken in order to approximate the noise. Since for the given LumiCal readout the condition $T_{smp} = 0.72\tau_{sh}$ is met, the maximum approximation error does not exceed 7 %, which is acceptable value for the noise estimation. Moreover, the normalized sum

of samples is greater (or at least equal) than the real value (as can be seen in Figure 2.30), so the noise level is overestimated, and never underestimated.

The example of the deconvolution procedure for a real LumiCal pulse, after pedestal removal and CMS procedures (channel C3 signal from Figure 2.23b), is presented in Figure 2.31. The top graph presents the sampled input pulse (red squares) with the deconvolution filter output samples (blue dots). The front-end pulse is represented after deconvolution by two non-zero samples, as expected from theory. However, the remaining samples are not exactly zero, due to the noise and common

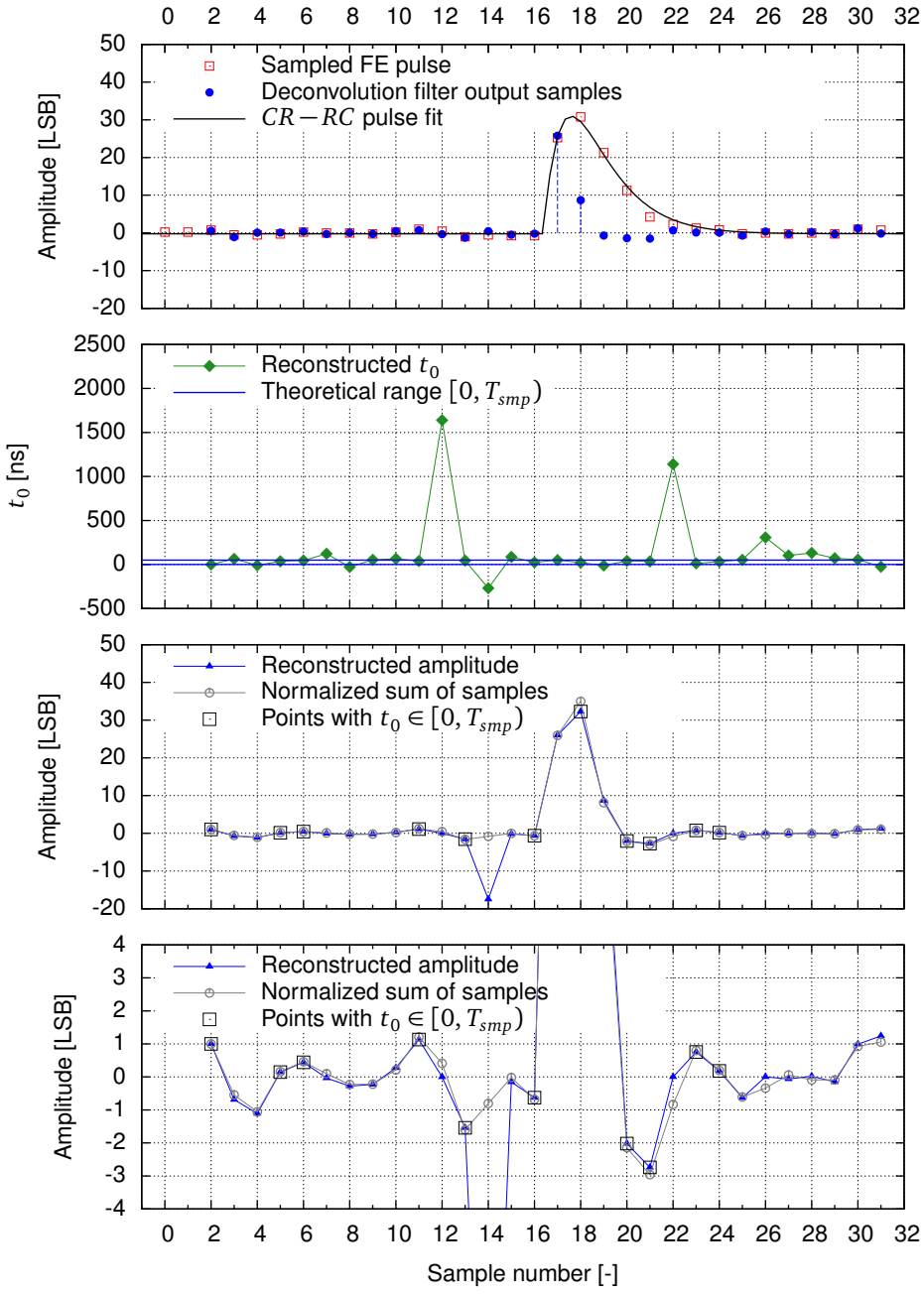


Figure 2.31: Example of the deconvolution results for real LumiCal pulse.

mode disturbances. This affects the reconstructed pulse starting time t_0, shown in the middle graph as green diamonds. The theoretical range of $[0, 50)$ ns, shown as blue lines, is exceeded by an order of magnitude for some samples. The reconstructed amplitude (with time dependent correction factor) is presented on the bottom plots as blue triangles while the normalized sum of samples – as gray circles. The points with $t_0 \in [0, T_{smp})$ are marked using black rectangles.

Due to negative sign of t_0 in the exponent of the time dependent correction factor (equation 2.36), a high positive value of t_0 makes the correction factor extremely small. Therefore, the amplitude reconstructed for a very high value of $t_0 \approx 1600$ (12th sample) is almost zeroed. Contrary to this, a negative $t_0 \approx -400$ (14th sample) increases significantly the correction factor, and the amplitude reconstructed for this sample is significantly lower than the average pedestal. In this case, the normalized sum of samples should be taken as a proper value of the 14th sample.

The theoretical $CR{-}RC$ pulse, given by equation 2.18, was fitted to the sampled front-end pulse. This fit is presented on top graph in Figure 2.31 as a black solid line. The pulse amplitude obtained from the fit is equal 31.82 ± 0.6 LSB, while the one reconstructed by the deconvolution process – 32.27 LSB. The pulse starting time t_0, achieved in both methods is also very similar, 20.5 ± 1.5 ns from the fit and 21.81 ns from the deconvolution. The convergence of the two results indicates that the deconvolution can be used for the amplitude reconstruction instead of the slow, sensitive to initial conditions, theoretical pulse fitting.

In addition to the benefits in single pulse amplitude reconstruction, the deconvolution improves the pile-up resolving capability, since the digital filter shortens effectively the pulse length [66]. However, due to low particle rate in the testbeam, the total number of pile-up-ed events was negligible. Therefore, since this problem is marginal for the performed data analysis, the deconvolution pile-up resolving capability is not discussed in this dissertation.

2.3 Testbeam results

All of the steps, discussed in Section 2.2, form a complete signal processing chain, providing the results enabling the physical data analysis. Apart from the telescope track reconstruction, described already in details, the LumiCal data processing consist of pedestal removal, CMS, deconvolution and final signal extraction procedure. The above steps were implemented in a dedicated software as a chain described in details below.

2.3.1 Initial data processing and noise performance

Initial data processing chain

The average pedestal is calculated for each channel in each event independently from the first fourteen samples, together with the corresponding standard deviation σ_{raw}. The average pedestal is then subtracted from all 32 samples of each channel in each event. The calculated standard deviation σ_{raw} is used in order to discriminate the channels containing the real sensor signal from the CMS procedure. As can be seen from Figure 2.23, common mode disturbances usually contain a single sample very far from the average pedestal, while the signal contains a few consecutive samples above the pedestal. Therefore, to exclude the real signal from the CMS procedure, without detecting mistakenly the common disturbance as a signal, the discrimination criterion was taken as follow: if at least two consecutive samples exceeds the $3\sigma_{raw}$ level, the channel is treated as containing