Proposal for Compactification of `CaloCell` Objects

S. MENKE
Max-Planck-Institut für Physik
(Werner-Heisenberg-Institut)
Föhringer Ring 6, D-80805 München, Germany
G. USAI
Dipartimento di Fisica, Università di Pisa,
INFN Sezione di Pisa,
Via Buonarroti, 2, I-56127 Pisa, Italy

DRAFT 3.0 October 29, 2004 11:45

Abstract

All ATLAS calorimeters together have 187652 cells. In the reconstruction software of ATLAS for each cell an object of type `CaloCell` with 195 bits of information for energy, time, gain and quality is created. Saving all the calorimeter cells with the LCG POOL persistency service consumes in excess of 1MB (the 1MB is of course dependent on the persistency service used because POOL does a gzip equivalent). In this proposal we are investigating algorithms to compactify the calorimeter data as much as possible without harming the resolution. Using a cubic root compactification method for the energy and a logarithmic compactification method for the time for the $\sim 10\,\%$ of the cells with calculated time the average size per cell could be reduced to 17.6 bits. This note describes the "lossy"[1] algorithms to achieve this goal and their implementation in the ATLAS reconstruction software.

1 Introduction

The ATLAS reconstruction output is split into two primary streams: Event Summary Data (ESD) and Analysis Object Data (AOD). The AOD stream contains minimal information needed for basic physics analysis. The ESD stream contains additional information that allows more fine grained analysis or partial re-reconstruction. The size of AOD and ESD had previously been set by ATLAS to be 10 and 100 kB. Unable to meet this tight requirement, the size of the ESD has been relaxed for the purposes of the Data Challenge 2 exercise. Feedback and experience from DC2 will help in determining the final size of the ESD. Exceeding the ESD size further would have a significant impact on the computing resources.

Typical objects saved in AOD includes electrons, photons, b-jets, muons and such which are the end products of the combined reconstruction and particle identification algorithms. Typical objects saved in ESD would include tracks, calorimeter clusters and other intermediate combined reconstruction objects. The calorimeter group has advocated that it is important to save the cell level information at the ESD level. This will let physicists have greater flexibility during the analysis stage without having to go back to the raw data. The persistency of all Calorimeter Cells would require > 1 MB of disk space

[1] A lossless algorithm does not loose any information in compression (e.g. gzip), a "lossy" algorithm does (e.g. jpeg).

far exceeding the total ESD size limits set by ATLAS. We have therefore chosen to explore various compactification scheme to reduce the total size taken by writing out the calorimeter cells. While a solution will be implemented for DC2 immediately, a final solution will be based on many factors including the impact on any physics that this may have.

2 Compactification Methods

The data contents of an object of type `CaloCell` is:

```
Identifier                  m_ID     ; // an unsigned integer identifying the
                                       // unique channel
double                      m_energy ; // cell energy in MeV in double precision
double                      m_time   ; // cell time in ns in double precision
double                      m_quality; // fit quality for the energy reconstruction
CaloGain::CaloGain          m_gain   ; // hardware gain setting, a signed integer
const CaloDetDescrElement*  m_caloDDE; // a pointer to the detector descreption
                                       // (geometry) of the cell
```

The pointer to `CaloDetDescrElement` is constructible from the `Identifier` and therefore redundant information. Assuming 64 bits for each `double` and 32 bits for each `int` the total data content is 256 bits without the redundant pointer. If all calorimeter cells are going to be persistified the `Identifier` is not needed if the cells are written out in an ordered manner by their `HashIdentifier` numbers which run continuously from 0 to $N_{\text{cell}} - 1$ without holes. The gain has only 3 possible values for LAr calorimeters: `HIGH, MEDIUM, LOW` and 6 possible values for the Tile calorimeter: `HIGHHIGH, ONEHIGH, HIGHLOW, LOWHIGH, ONELOW, LOWLOW`. Without loss of information 195 bits are therefore needed for each `CaloCell`.

In order to further reduce the number of bits needed the precision of each data member has to be limited. Three different types of precision reduction have been considered for each persistable data member:

fixed value: this method assigns 2^n different fixed values for a data member to persistify with n bits. This is the most general method and the other two could be regarded as special cases of this method.

logarithmic: this method uses 1 bits for the sign and $n-1$ bits for the absolute value on a logarithmic scale with bounds $x_0 < |x| < x_1$. The relative loss of precision

$$\frac{\sigma'_x}{x} = \frac{1}{\sqrt{12}} \left(1 - \left(\frac{x_0}{x_1}\right)^{2^{1-n}}\right) \tag{1}$$

is constant for this method.

k^{th} root: this method uses 1 bit for the sign and $n-1$ bits for the k^{th} root of the absolute value with bounds $0 < |x| < x_1$. The relative loss of precision

$$\frac{\sigma'_x}{x} = \frac{k}{\sqrt{12}} \left(\frac{x_1}{x}\right)^{1/k} 2^{1-n} \tag{2}$$

is proportional to $x^{-1/k}$ for this method.

3 Choice of Compactification

3.1 Quality and Gain

A bit (which is called the quality bit) is set if the time is calculated in the RODs. This allows us to locate the time information in the compactified object. The original quality word of the `CaloCell` is ignored at present.

Three gain settings `HIGH, MEDIUM, LOW` are used for the LAr systems and the following mapping of the six Tile gain values to these three fixed values is used: `HIGHHIGH, ONEHIGH` → `HIGH`; `HIGHLOW, LOWHIGH` → `MEDIUM`; `ONELOW, LOWLOW` → `LOW`. This mapping for the Tile is not unambiguously reversible. However, the two energy readings for a `TileCell` are combined to one energy value in a `CaloCell` and even with the more detailed gain information the original two energies are not available from the `CaloCell` anymore.

3.2 Energy

The precision loss for the energy field should not have a significant impact on the accuracy of the measured energy. Since the intrinsic energy resolution satisfies $\sigma_E/E = c/\sqrt{E\ (\mathrm{GeV})}$, with some constant c the requirement on the additional uncertainty introduced by the compactification is $\sigma'_E/E << c/\sqrt{E\ (\mathrm{GeV})}$, with $\sigma'_E = \Delta E/\sqrt{12}$, where ΔE is the width of the energy interval E falls into for a given compactification method. For a given constant d and the desired dynamic range $E_0 < |E| < E_1$ the minimum number of bits needed for encoding $|E|$ is given by the condition $\sigma'_E/E < d/\sqrt{E\ (\mathrm{GeV})}$, for all E. Three cases are explicitly studied: logarithmic, square root and cubic root compactification. The minimum number of bits for the logarithmic method (assuming that 1 bit is needed for the sign) is given by:

$$n > 1 - \frac{1}{\ln 2} \ln \left(\frac{-\ln \left(1 - \frac{d \cdot \sqrt{12}}{\sqrt{E_1\ (\mathrm{GeV})}} \right)}{\ln \left(\frac{E_1}{E_0} \right)} \right). \tag{3}$$

For square root compactification again including 1 bit for the sign the requirement is:

$$n > 1 - \frac{1}{\ln 2} \ln \left(\frac{\sqrt{12}}{2} \frac{d}{\sqrt{E_1\ (\mathrm{GeV})}} \right), \tag{4}$$

and using cubic root one gets:

$$n > 1 - \frac{1}{\ln 2} \ln \left(\frac{\sqrt{12}}{3} \frac{d}{\sqrt{E_1\ (\mathrm{GeV})}} \right). \tag{5}$$

The minimal number of bits will be for square root compactification, since here the precision loss follows the required $1/\sqrt{E}$ form. Cubic root is almost as good in resolution but with a slightly better dynamic range and due to the $1/E^{1/3}$ form with better resolution for smaller energies. The logarithmic compactification needs the most number of bits since it deviates strongly from the $1/\sqrt{E}$ form, with much to good resolution for small energies. Another constraint is the dynamic range. The hardware gain ratio for the LAr systems is 10 between two adjacent ranges. The two gain settings in the Tile (for an individual channel) differ by a factor of 64. Furthermore the energy thresholds at which each subsystem switches the hardware gain are different for all systems and adjustable. Therefore it is not

feasible to use the hardware gain in the compactification procedure to define an upper energy limit. However, no energy larger than 3.2 TeV is expected in a single cell in ATLAS. A lower limit could be set by the requirement that deposits as small as 8 MeV (although the k^{th}-root methods do not depend on the lower threshold) should be compactifiable. For cells which are measured in HIGH gain (MEDIUM gain for the HEC) and below 50 GeV a special compactified gain value will be used indicating a lowered upper threshold of 50 GeV and therefore higher resolution. Upon reconstruction of the energy and gain from the compactified object the special gain value is translated back to HIGH gain (MEDIUM gain for HEC) and the lower threshold of 50 GeV is used to calculate the energy. Should a cell have energy below 50 GeV but a hardware gain different from HIGH (MEDIUM for HEC) the normal resolution range (up to 3.2 TeV) and the original hardware gain are used during compactification. This procedure ensures that the original energies and gains are always reconstructible from the compactified object. We have therefore two internal ranges in the compactification:

$$\text{high resolution}: \quad 8\,\text{MeV} - 50\,\text{GeV}, \tag{6}$$

$$\text{normal resolution}: \quad 512\,\text{MeV} - 3.2\,\text{TeV}. \tag{7}$$

The best intrinsic resolution for electromagnetic showers is expected to be about $10\,\%/\sqrt{E\ (\text{GeV})}$. The constant d should therefore be smaller than 10 %. If we allow for a degradation of the sampling term in the resolution by 1 % to $10.1\,\%/\sqrt{E\ (\text{GeV})}$, we get:

$$d \leq 1.4\,\%. \tag{8}$$

Using $d = 1.4\,\%$ and the normal resolution region (since it will give the worst result) we get the following limits on the number of bits to use:

$$\text{logarithmic}: \quad n > 14.315, \tag{9}$$

$$\text{square root}: \quad n > 12.188, \tag{10}$$

$$\text{cubic root}: \quad n > 12.773. \tag{11}$$

The example shows that with 13 bits and cubic root compactification the worst degradation of intrinsic resolution would be less than 1.0 % assuming an internal resolution of $10\,\%/\sqrt{E\ (\text{GeV})}$ corresponding to a relative compactification precision of better than $1.4\,\%/\sqrt{E\ (\text{GeV})}$. In the high resolution region the degradation would be 0.011 % for the same intrinsic resolution corresponding to a relative compactification precision of better than $0.15\,\%/\sqrt{E\ (\text{GeV})}$. In absolute numbers the resolution at the range boundaries for 13 bits can be found in table 1. For comparison a linear packing with constant absolute resolution would be limited at the lower energy end (i.e. at 512 MeV in normal resolution) and would need $n > 17.493$ bits to reach the same precision as the non-linear methods discussed above.

3.3 Time

With 3 bits for quality and gain and 13 bits for the energy most cells will have just 16 bits since most cells will have no time calculated and therefore the quality `BAD`. For about 10 % of the cells the ROD will calculate a time. Adding another 16 bit word for 10 % of the cells would bring the average cell size to 17.6 bits. With this size per cell the total amount of calorimeter data would be 0.39 MB per event. The time is a number in ns and should be within a 25 ns window for each `GOOD` cell. The accuracy of the time depends however strongly on the energy and is best for the highest energies. In the LAr systems energy and time of a cell are calculated from digital filter weighted sums of E and $E \times t$ running over 5 25 ns samples. While physical times should stay within 25 ns it is possible to get values outside that window due to noise. Any reconstructed time outside the window covered by the 5 samples (or 125 ns)

resolution	E/GeV	σ'_E/MeV	method
normal	3200	1969	logarithmic
normal	3200	451	square root
normal	3200	677	cubic root
normal	0.512	0.315	logarithmic
normal	0.512	5.71	square root
normal	0.512	1.99	cubic root
high	50	30.8	logarithmic
high	50	7.05	square root
high	50	10.6	cubic root
high	0.008	0.005	logarithmic
high	0.008	0.089	square root
high	0.008	0.031	cubic root

Table 1: *Precision loss due to compactification at various energies for three different methods with* 13 *bits*.

is probably not very reliable. For the Tile both channels of the cell will reconstruct a time and their weighted average is stored in the `CaloCell`. Since the precision of the reconstructed time rises with energy and the time value should be centered at 0 ns the logarithmic method seems appropriate. With a lower time threshold of 0.001 ns, and an upper threshold of 1250 ns and 16 bits including sign the relative precision loss for any time $0.001\,\text{ns} < |t| < 1250\,\text{ns}$ according to (1) is: $\sigma'_t/t = 1.24 \cdot 10^{-4}$. The boundaries are rather comfortable, but with the achieved precision of $1.24 \cdot 10^{-4}$ the resolution in the central 25 ns region of ± 12.5 ns is still better than 1.6 ps.

3.4 Additional Information for the Tile Calorimeter

The Tile calorimeter cells are read out by two photo multiplier tubes (PMTs) giving two independent measurements of energy and time per cell. The gap scintillators, mounted on the Tile extended barrel modules, instrumenting the gap between barrel and extended barrel, are on the contrary read-out by a single PMT.

For this reason the `TileCell` class contains different data members with respect to the `CaloCell` class and needs a special treatment in the cell compactification scheme.

In `CaloCell` the cell energy for the Tile is just the sum of the two PMT energies and the time is calculated as the average of the two PMT times. The gain in `CaloCell` is an enumerate type and for the Tile six values are needed to take into account also the gap scintillators (`LOWLOW, LOWHIGH, HIGHLOW,HIGHHIGH,ONELOW,ONEHIGH`).

In `TileCell` there are separate energy, time, quality and gain (`LOW,HIGH`) data members for each PMT.

As an optional compactification scheme the information of the two PMTs can be stored instead of the average `CaloCell` information. The choice is to store each PMT using 16 bits. Two bits are used to flag the PMT gain (`HIGH, LOW`) and a signal overall quality (`GOOD, BAD`). The remaining $13 + 1$ bits are used to store the absolute value of the energy using the cubic root method, and the energy sign.

In this way the detailed information of the Tile cells is saved and the mapping problem of the `TileCell` gain discussed in section 3.1 is solved.

Unlike the default scheme the energy bounds for compactification are coupled to the hardware gain in this scheme and make therefore the use of some safety factor advisable. The upper energy limit for PMTs in high gain is 50 GeV (a factor five larger than the expected maximum energy in this gain). The

low gain upper limit for each PMT is set to 3.2 TeV. Two PMTs at that limit would correspond to a total Tile cell energy of 6.4 TeV. For comparison, we can expect (from the inclusive QCD background spectrum) to have about 10 events per year at high luminosity where the energy deposit in one cell is above 1.2 TeV.

The time information is also stored independently for each PMT with the same compactification scheme as described in section 3.3 in an additional 16 bit word in case the time was calculated. This is indicated by the value of the signal reconstruction quality, defined in the `TileCell` class.

Also the quality is differently treated in the `TileCell` and `CaloCell` classes. In `TileCell` the quality is a 32 bit integer with a 16 bit word for each PMT. In particular one bit flag for each PMT is used to resume an overall "quality" (`GOOD, BAD`) which is the bit written in the compact cell and defining whether or not the time is calculated and stored.

Assuming a 10 % fraction of PMTs with time information this optional compactification scheme requires 22.8 kB per event for the Tile, whereas the default scheme requires about half that size.

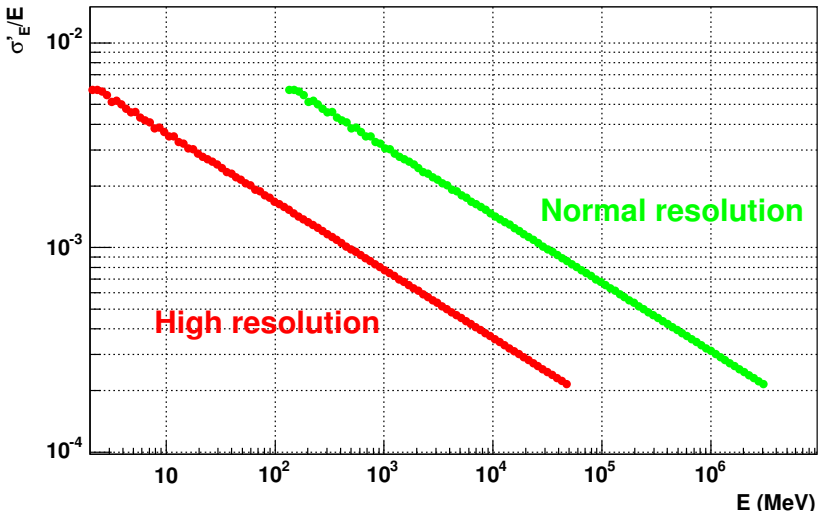

Fig. 1: *Relative loss in precision due to compactification of cell energies for the two different resolution regions. The* 13 *bit cubic root method (see text) is used.*

3.5 Summary

In summary the current choice of compactification consists of 1 bit for whether the time information exists or not, 1 bit for the sign of the energy, 12 bits for the cubic root of the absolute value of the time, 2 bits for the gain, and in case the time information is also available 1 bit for the sign of the time, 15 bits for the logarithm of the absolute value of the time. The total transient memory is therefore reduced to 0.39 MB per event. The total persistent disk space with LCG POOL is 0.25 MB.

4 Tests

In order to test the predicted loss of precision a toy MC has been written that generates energies according to a $1/E$-spectrum in the two resolution ranges, up to 50 GeV, and up to 3.2 TeV. The energy values E_i are compactified with the 13 bit cubic root method mentioned above and a de-compactified energy E_i' is calculated from the compact value. The differences $E_i - E_i'$ are histogramed in bins of equal width in $\log E$ in profile histograms with the 'spread' option, such that the error in each bin will be the

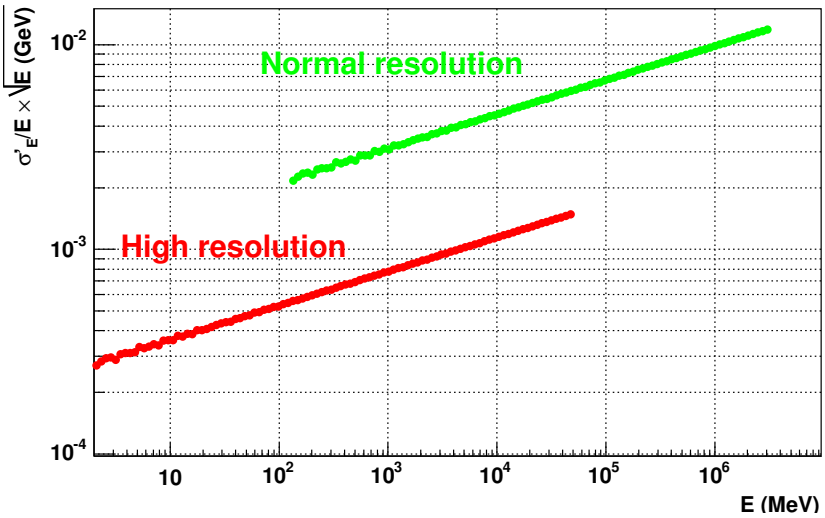

Fig. 2: *'Sampling term' of the relative loss in precision due to compactification of cell energies for the two different resolution regions. The* 13 *bit cubic root method (see text) is used.*

RMS of the deviation of the reconstructed energy from the original one. The relative loss of precision in each bin σ'_E/E is obtained by dividing the above RMS value by the generated energy. Figure 1 shows this relative precision loss for all three gain settings. The direct comparison with the sampling term is best seen by drawing $\sigma'_E/E \times \sqrt{E\ (\mathrm{GeV})}$, which is shown in figure 2. Both plots confirm the numbers stated in the sections before.

Another test was performed with the standard athena reconstruction framework of ATLAS. 100 events of the simulated data /castor/cern.ch/atlas/project/dc2/preprod/g4dig805/dc2.002885.pyt_z_ee.-g4dig805/data/dc2.002885.pyt_z_ee.g4dig805._0001.pool.root have been analyzed with and without compactification. The topological clustering with the following cuts has been used in both cases:

```
CaloTopoClusterMaker.TopoCluster.CaloNoiseTool ="CaloNoiseTool/calonoisetool"
CaloTopoClusterMaker.TopoCluster.UseCaloNoiseTool                = TRUE
CaloTopoClusterMaker.TopoCluster.UsePileUpNoise                  = FALSE
CaloTopoClusterMaker.TopoCluster.NeighborOption                  = "super3D"
CaloTopoClusterMaker.TopoCluster.CellThresholdOnAbsEinSigma      =    0.0
CaloTopoClusterMaker.TopoCluster.NeighborThresholdOnAbsEinSigma  =    3.0
CaloTopoClusterMaker.TopoCluster.SeedThresholdOnEinSigma         =    5.0
CaloTopoClusterMaker.TopoCluster.CellThresholdOnAbsEt            =   -1
CaloTopoClusterMaker.TopoCluster.NeighborThresholdOnAbsEt        =   -1
CaloTopoClusterMaker.TopoCluster.SeedThresholdOnEt               =   -1
CaloTopoClusterMaker.TopoSplitter.NeighborOption                 = "super3D"
CaloTopoClusterMaker.TopoSplitter.NumberOfCellsCut               = 4
CaloTopoClusterMaker.TopoSplitter.EtDensityCut = 500*MeV/(600000*mm3)
```

These settings correspond to the default values in the athena offline release 9.0.0 except for the lowered seed cut and the switched off cuts on $E_\perp$ in order to get more clusters. The plots in figure 3 and figure 4 show a comparison of the energy spectra (actually $E/|E|\log10(|E|\ (\mathrm{MeV}))$) for all 4 calorimeter systems. The figures show that signal and noise can be reconstructed with high precision from the compactified energy information.

Figure 5 shows a comparison of the gain settings. All LAr cells are reconstructed with the correct gain regardless of subsystem and regardless of the internal choice of high or normal resolution. For

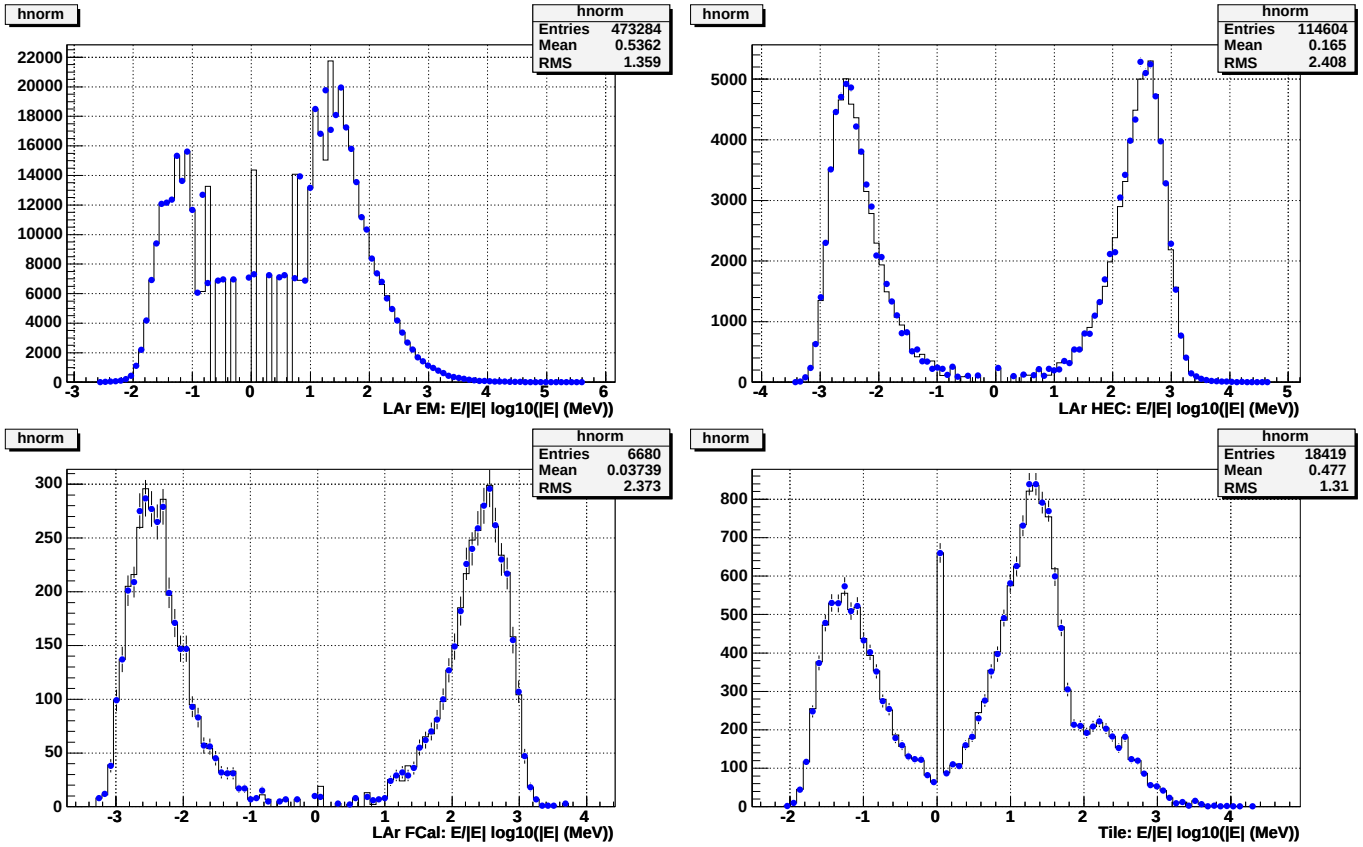

Fig. 3: *Cell energies of cells in topo clusters for 100 events with (dots with error bars) and without (solid line) compactification with version 300. The same histograms as in figure 4 are shown here on a linear y-scale. Clockwise from top left: LAr EM, LAR HEC, Tile, LAr FCal.*

the Tile the original entries at -4 and -3 (`ONELOW`, and `ONEHIGH`) are not reconstructible from the compactified data and therefore moved to -16 (`LOWLOW`) and -11 (`HIGHHIGH`) instead. This could probably be recovered from geometrical information since these cells probably have only one channel instead of two.

The analyzed MC does not contain simulated time values for the Tile calorimeter. For the LAr calorimeters the comparison of the time spectra is shown in figure 6 and figure 7. For the LAr electromagnetic calorimeter 11 % of all cells report a time and only 0.008 % of the cells with time have a time outside a 250 ns window. For the LAr hadronic endcap calorimeter 43 % of all cells report a time and 12 % of the cells with time fall outside the central 250 ns window. Due to the comfortable upper threshold of 1250 ns all time values are reconstructed correctly for this subsystem too. The situation for the LAr forward calorimeter is similar to the hadronic endcap calorimeter: 40 % of all cells report a time and 4 % of those are outside the central region.

While the large percentage of cells with time might be due to the fact that only cells in clusters enter the analysis, it is remarkable that so many time values are located in the unphysical regions. From the energy spectra for the HEC and the FCal with almost identical peaks for negative and positive energies it follows that most of the cells with time are just noise hits. Therefore it is not too surprising that the time measurement is not within the physical 25 ns and often even outside the 250 ns window.

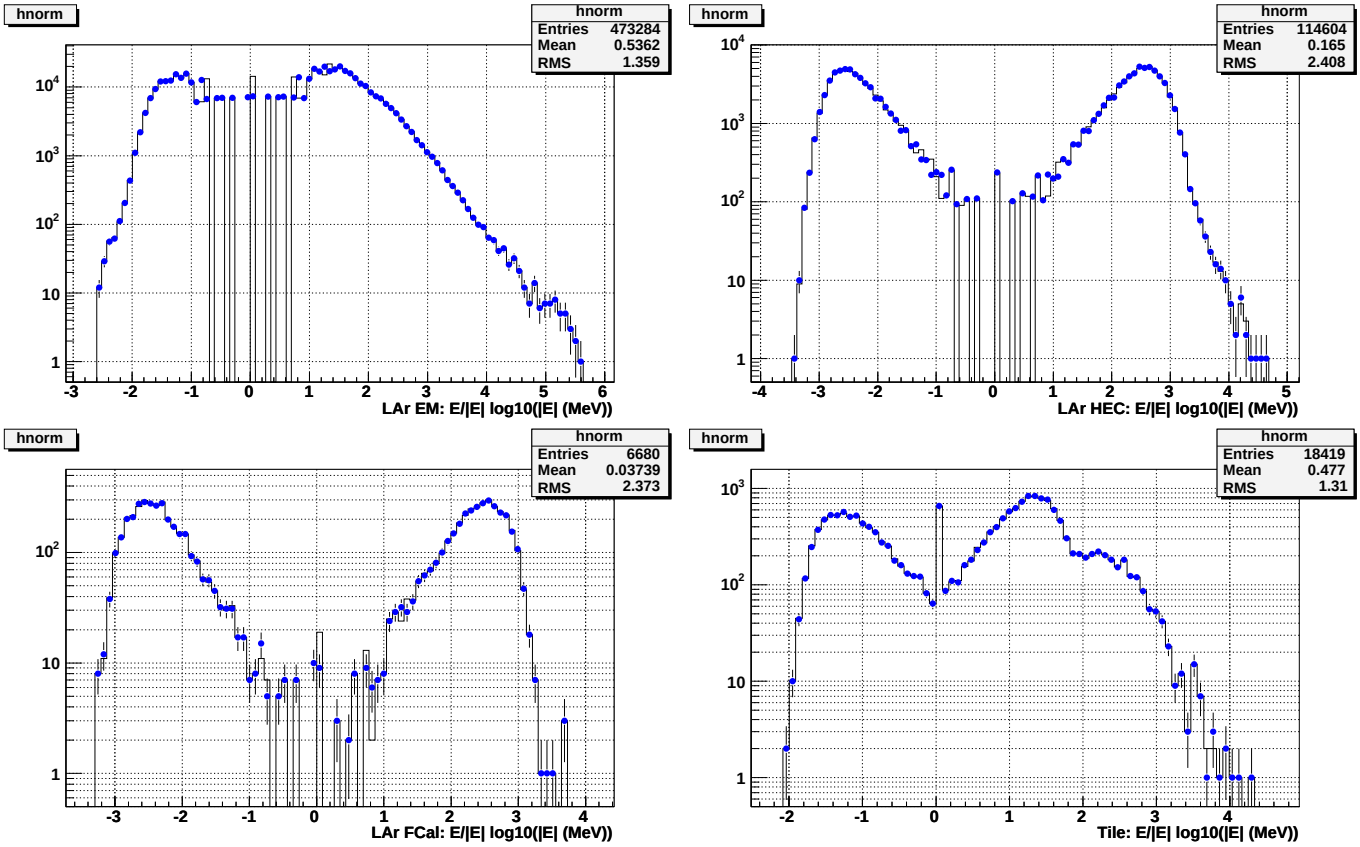

Fig. 4: *Cell energies of cells in topo clusters for 100 events with (dots with error bars) and without (solid line) compactification with version 300. The same histograms as in figure 3 are shown here on a logarithmic y-scale. Clockwise from top left: LAr EM, LAR HEC, Tile, LAr FCal.*

5 Conclusions

Compactification of calorimeter energies can be efficiently performed by transforming the energy to a non-linear scale. Due to the $1/\sqrt{E}$ nature of the intrinsic resolution of sampling calorimeters functions like square root or cubic root are most suitable for this purpose. It has been demonstrated that 13 bit cubic root compactification with 2 resolution modes is sufficient to limit the loss in intrinsic resolution of a calorimeter with resolution $10\,\%/\sqrt{E\ (\mathrm{GeV})}$ to 1.0 % in the worst case for the normal resolution range and to 0.011 % in the worst case for the high resolution range which covers the energies up to 50 GeV if they are taken in high (HEC: medium) gain. The software has been released for use with DC2 and feedback from physics communities would be essential for any further optimization.

Appendix A: Software Implementation

The implementation described here is based on the athena offline release 9.0.0. The `AlgTool` `offline/-Calorimeter/CaloTools/CaloCompactCellTool` is doing the conversion from and to persistent compactified calorimeter data of type `CaloCompactCellContainer` to and from the transient object of type `CaloCellContainer` containing `CaloCell` objects. The `AlgTool` defines the individual bit masks, conversion routines, and boundary values. The number of bits for the compactification of the absolute values by the logarithmic or the cubic root method are calculated from the mask. In order to allow for modifications in the code without compromising the ability to read back persistified data created with older versions a unique header format is defined preceding the persistified data. The header consists of

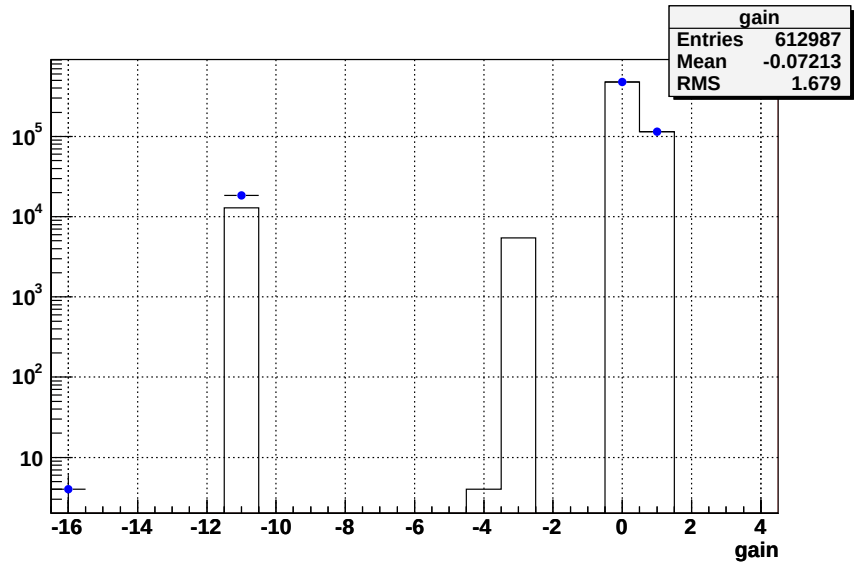

Fig. 5: *Gain settings of cells in topo clusters for 100 events with (dots with error bars) and without (solid line) compactification with version 300. The gain values are: Tile:* −16 *(*`LOWLOW`*),*−15 *(*`LOWHIGH`*),*−12 *(*`HIGHLOW`*),*−11 *(*`HIGHHIGH`*),*−4 *(*`ONELOW`*),*−3 *(*`ONEHIGH`*); LAr:* 0 *(*`HIGHGAIN`*),* 1 *(*`MEDIUMGAIN`*),* 2 *(*`LOWGAIN`*).*

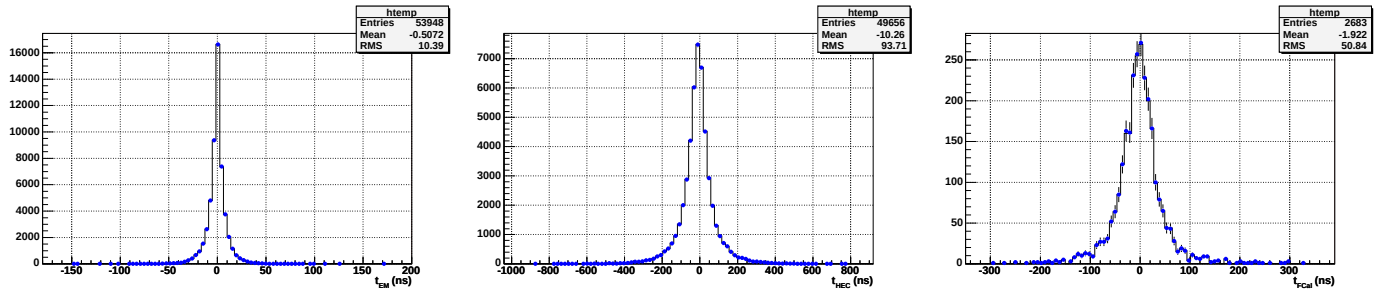

Fig. 6: *Cell times of cells in topo clusters for 100 events with (dots with error bars) and without (solid line) compactification with version 300. The same histograms as in figure 7 are shown here on a linear y-scale. From left to right: LAr EM, LAR HEC, LAr FCal.*

32 bit words and contains the header length in units of 32 bit words as first entry and a version number as second entry. Remaining fields if any are version dependent. The header format for the most recent version implemented looks like this:

```
// header for VERSION_300
  struct header300 {
    int length;
    int version;          // default = 300
    int nCellsLAREM;
    int nCellsLARHEC;
    int nCellsLARFCAL;
    int nCellsTILE;
    int m_QUALY           // default = 0x8000
    int m_EGAIN;          // default = 0x6000
    int m_ESIGN;          // default = 0x1000
    int m_CRTAE;          // default = 0x0FFF
```

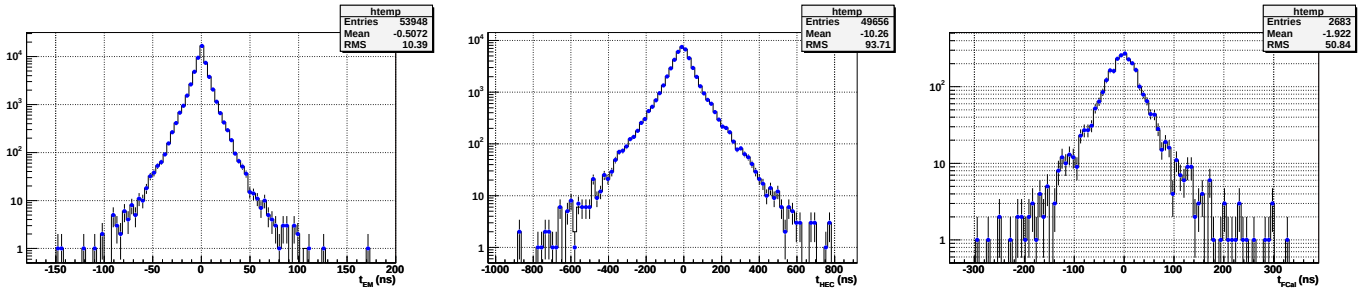

Fig. 7: *Cell times of cells in topo clusters for 100 events with (dots with error bars) and without (solid line) compactification with version 300. The same histograms as in figure 6 are shown here on a logarithmic y-scale. From left to right: LAr EM, LAR HEC, LAr FCal.*

```
    int m_TSIGN;          // default = 0x8000
    int m_LOGAT;          // default = 0x7FFF
    int m_QABAD;          // default = 0; cells without time info
    int m_ENLOW;          // default = 0; normal resolution low gain
    int m_ENMED;          // default = 1; normal resolution medium gain
    int m_ENHIG;          // default = 2; normal resolution high gain
    int m_EHHIG;          // default = 3; high resolution high (HEC: medium) gain
    float m_e1_norm_res   // default = 3.2*TeV; upper eng. limit normal resol.
    float m_e1_high_res   // default = 50*GeV;  upper eng. limit high resolution
    float m_t0;           // default = 0.001*ns; lower time limit
    float m_t1;           // default = 1250*ns ; upper time limit
};
```

The fields in the header listed above are:

`int length:` the total size of the header in 32 bit units.

`int version:` the version number. Compactification algorithms and organization of the storage is version dependent.

`int nCellsLAREM:` the number of persistified `CaloCell` objects from the electromagnetic LAr calorimeter. This number might be larger than the number of `CaloCell` objects from this system in the original `CaloCellContainer`. This happens whenever no `CaloCell` is found for an `IdentifierHash` value in the value range of the system. The special energy -3.2 TeV (actually minus the maximum energy in normal resolution) and quality $= -1$ and gain = LOW is assigned to the persistent object and upon creation of the transient no `CaloCell` object is created, but the `IdentifierHash` value is incremented.

`int nCellsLARHEC:` same for the LAr hadron endcap calorimeter.

`int nCellsLARFCAL:` same for the LAr forward calorimeter.

`int nCellsTILE:` same for the hadronic tile calorimeter.

`int m_QUALY:` bit mask for the quality. By default this is 1 bit, since version 300 can only distinguish two different quality states.

`int m_EGAIN:` bit mask for the gain field. By default this is 2 bits. Version 300 can distinguish three different gain states in normal resolution and one special high resolution mode meaning medium gain for the HEC and high gain for the other systems.

`int m_ESIGN:` bit mask for the sign of the energy. 0 means positive, 1 negative energy.

`int m_CRTAE:` bit mask for the cubic root of the absolute value of the energy. By default this is 12 bits. Version 300 stores quality, gain, and energy in one 16 bit word. The remaining bit masks are for the time which is stored in a second 16 bit word for about 10 % of the cells with quality ≥ 0. The first time word is stored after the last energy word in the persistent container.

`int m_TSIGN:` bit mask for the sign of the time. 0 means positive, 1 negative time.

`int m_LOGAT:` bit mask for the logarithm of the absolute value of the time. By default this is 15 bits.

`int m_QABAD:` specifies which state in the quality bit means quality < 0. The default is 0.

`int m_ENLOW:` specifies the value for low gain. The default is 0.

`int m_ENMED:` specifies the value for medium gain. The default is 1.

`int m_ENHIG:` specifies the value for high gain. The default is 2.

`int m_EHHIG:` specifies the value for high (HEC: medium) gain and high resolution. The default is 3.

`float m_e1_norm_res:` specifies the maximum energy in normal resolution mode in MeV. The default is 3200000 MeV.

`float m_e1_high_res:` specifies the maximum energy in high resolution mode in MeV. The default is 50000 MeV.

`float m_t0:` specifies the lower bound for the logarithmic packing of the absolute value of the time in ns. The default value is 0.001 ns.

`float m_t1:` specifies the upper bound for the logarithmic packing of the absolute value of the time in ns. The default value is 1250 ns.

All of the bit fields and thresholds are adjustable. Since they are part of the header the creation of the transient does not rely on the default values. Intrinsic to a version is only the choice of compactification algorithms and the treatment of the time – version 300 persistifies times for all cells with quality > 0 and expects a time in the persistified object upon reconstruction of the transient whenever the quality bit does not equal `m_QABAD`.

There are some slight differences in the format of the header for the optional scheme (version 301) with detailed Tile information with respect to version 300. In particular the version 301 contains different bit masks for the Tile gain, energy and sign field, whereas the quality bit mask is common for LAr and Tile:

```
// additional header fields for VERSION_301
    int m_EGAIN_TILE;    // default = 0x4000
    int m_ESIGN_TILE;    // default = 0x2000
    int m_CRTAE_TILE;    // default = 0x1FFF
    int m_GHIGH;         // default = TileID::HIGHGAIN,
    int m_GLOW;          // default = TileID::LOWGAIN,
    float m_high_tile;   // default = 50*GeV
    float m_low_tile;    // default = 3.2*TeV
```

The additional fields in the header are:

`int m_EGAIN_TILE`: bit mask for the PMT gain field. The Tile PMT needs only 1 bit in order to distinguish high and low gain.

`int m_ESIGN_TILE`: bit mask for the sign of the PMT energy. 0 means positive, 1 negative energy.

`int m_CRTAE_TILE`: bit mask for the cubic root of the absolute value of the PMT energy. By default this is 13 bits.

`int m_GHIGH`: specifies the value for PMT high gain. The default is `TileID::HIGHGAIN = 1`.

`int m_GLOW`: specifies the value for PMT low gain. The default is `TileID::LOWGAIN = 0`.

`float m_high_tile`: specifies the maximum PMT energy in high gain mode in MeV. The default is 50000 MeV.

`float m_low_tile`: specifies the maximum PMT energy in low gain mode in MeV. The default is 3200000 MeV.

The two methods used for the conversions are:

```
StatusCode CaloCompactCellTool::getPersistent
           (const CaloCellContainer & theCellContainer,
            CaloCompactCellContainer * theCompactContainer,
            const int theVersion);
```

to create the persistent `CaloCompactCellContainer` object from the transient `CaloCellContainer` object and

```
StatusCode CaloCompactCellTool::getTransient
           (const CaloCompactCellContainer & theCompactContainer,
            CaloCellContainer * theCellContainer);
```

to create the transient `CaloCellContainer` object from the persistent `CaloCompactCellContainer` object.