COMPLEXITY SCALING OF AUDIO ALGORITHMS: PARAMETRIZING THE MPEG ADVANCED AUDIO CODING RATE-DISTORTION LOOP

Pablo Delgado, Markus Lohwasser

Fraunhofer IIS
Realtime Audio Processing Group
Erlangen, Germany
pablo.delgado@iis.fraunhofer.de

ABSTRACT

Implementations of audio algorithms on embedded devices are required to consume minimal memory and processing power. Such applications can usually tolerate numerical imprecisions (distortion) as long as the resulting perceived quality is not degraded. By taking advantage of this error-tolerant nature the algorithmic complexity can be reduced greatly. In the context of real-time audio coding, these algorithms can benefit from parametrization to adapt rate-distortion-complexity (R-D-C) trade-offs. We propose a modification to the rate-distortion loop in the quantization and coding stage of a fixed-point implementation of the Advanced Audio Coding (AAC) encoder to include complexity scaling. This parametrization could allow the control of algorithmic complexity through instantaneous workload measurements using the target processor's task scheduler to better assign processing resources. Results show that this framework can be tuned to reduce a significant amount of the additional workload caused by the rate-distortion loop while remaining perceptually equivalent to the full-complexity version. Additionally, the modification allows a graceful degradation when transparency cannot be met due to limited computational capabilities.

1. INTRODUCTION

Development of low-power, fast implementations of perceptual audio codecs is always challenging due to the complex nature of the signal processing involved. In addition, the necessity of operating these algorithms on computationally-restricted architectures for mass production and low delay requirements present an additional constrain on workload. The goal in this case is to optimize execution speed and power consumption while remaining perceptually indistinguishable from a possible reference implementation on a more powerful platform. If this goal cannot be met for complexity reasons, the degradation in quality should be gradual according to perceptual rules.

In order to achieve this, a major task consists on porting a floating-point code to a fixed-point version suited for low-power platforms. It has been already shown that the same overall perceived quality (according to listening tests and objective measurements) of AAC and mp3 codecs can be preserved even with the precision loss associated with porting code from floating-point arithmetic to a 32-bit, fixed-point representation [1]. These results were achieved by using proper scaling of audio signal energies and masking thresholds at various points in the psychoacoustic model, adapting a fractional arithmetic to pure integer processors and using a logarithmic representation of signals in order to transform costly division operations to subtractions, and to attain a suitable mapping of the dynamic range.

On a more general plane, there have been recent discussions on the implementation of multimedia processing algorithms and the role of fixed complexity boundaries on error-tolerant applications [2]. Due to limitations of silicon CMOS technology in providing further increase in speed at acceptable fault-rate and energy-dissipation, current research suggests a closer look at multimedia applications in terms of precision and resilience requirements [3]. Particularly, parametric adaptation of rate-distortion-complexity curves [4] of audio coding algorithms at different stages can greatly help in optimally exploiting the capabilities of each target platform. Moreover, this parametrization also accelerates the process of achieving optimal performance on new processors, independently of hardware optimizations.

The MPEG 2/4 AAC codec is present among the most prominent examples of perceptual audio coding (PAC) technologies. PAC techniques usually convert the time domain input samples into a frequency domain representation in order to remove redundancies and irrelevancies of the signal for efficient transmission and/or storage. In doing so, the coding noise power is adapted to a hearing threshold provided by a human perceptual model [5] in a way that the noise is as less disturbing as possible. Some latest examples include the Unified Speech and Audio Coding (USAC) [6] and the recent MPEG-H Audio [7] standards. Complexity and workload are of particular importance for encoding and decoding audio in real-time due to their application on low-power mobile devices. Versions of the AAC codec specially fit for this task are AAC Low Delay (AAC-LD) [8] and AAC Enhanced Low Delay (AAC-ELD) and AAC-ELD version 2 (AAC-ELDv2) [9] and 3GPP Enhanced Voice Service (EVS) [10]. These variations feature shorter transform lengths in order to accommodate low delay requirements and eliminate or greatly limit bit reservoir techniques for maintaining a constant time delay ([11],[12]), the trade-off being loss of frequency resolution and an increase in workload for a fixed sampling rate.

The AAC encoder carries the most algorithmically complex modules of the codec [1]. Particularly, the psychoacoustic model block including the time-to-frequency transform and the quantization and coding block are the most demanding in terms of computations [13]. Since, for MPEG codecs, only the AAC decoder is standardized, modifications on the AAC encoder to achieve better performance are possible as long as the produced encoded output produces a valid bitstream. This work will focus on the quantization block by studying a possible parametrization of the internal algorithm to scale its complexity according to the available processing power.

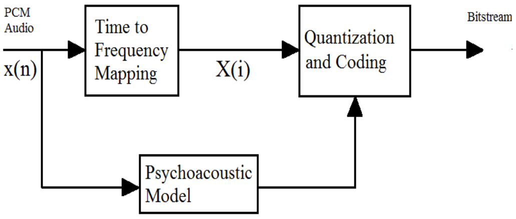

Figure 1: *Simplified block diagram of the AAC Encoder*

1.1. Comparison to previous approaches

Although previous efforts were made in order to identify and optimize the most sensitive steps in terms complexity of the quantization stage ([14], [15]), the proposed algorithms offer efficient implementations with varying compromises in quality, but offer no parametrization of complexity factors except for setting a maximum limit to the number of iterations for rate and distortion loops.

In [14] the authors propose a scheme based on an estimation of the non-uniform quantization stage of the AAC encoder in order to implement a loop-less bit-allocation. This approach results in a significant reduction in complexity (reported 80% to 90%) to the expense of a significant reduction in audio quality. Nevertheless, it is not clear how this approach can benefit from parametrization in cases where the same algorithm is implemented on more powerful architectures and a better audio quality can be allowed.

In this regard, the authors of [15] also propose an hybrid approach where loop-less bit-allocation is used as a starting point for a loop-based method. They claim at least a factor of 10 in complexity reduction with respect to the standard approach to bit-allocation in AAC by decoupling rate and distortion loops. Since only AAC audio decoders -and not encoders- are standardized, the proposed method in the AAC standard is only provided as a reference and is far from meeting the complexity requirements of a low-power and/or real-time implementation. The encoding approach can vary significantly as long as the produced bit-streams are valid. Consequently, many of today's available solutions -including the implementation of AAC used for our work- also shows similar performance with respect to this approach [12], [1].

To the best of our knowledge, the transition between a loop-less approach and a higher quality loop-based bit-allocation is not studied previous works. An equivalent method to that in [14] based on estimation of the quantization noise is already present in our implementation of the AAC encoder in addition to a loop-based bit-allocation method for higher quality. Our work proposes a closer look to the quality-complexity trade-off and a solution for a smoother transition between the two modes of operation.

2. SYSTEM OVERVIEW

The quantization and coding stage of the AAC encoder quantizes the spectral data provided by a Modified Discrete Cosine Transform (MDCT) of a Pulse Code Modulated (PCM) audio signal, in a way that the quantization noise satisfies the demands of the psychoacoustic model [5](Figure 1). On the other hand, the number of bits needed to quantize the spectrum must be below a certain limit, typically the average number of available bits for a block of audio data. The way in which this trade-off is approached is the core of the coding strategy. This strategy is usually carried out in an Analysis-by-Synthesis manner, in which the resulting quantized and coded spectral lines are evaluated and re-quantized until an optimal solution within a range is reached.

The AAC quantization process involves the gain adjustment of groups of spectral values (scale factor bands) which are then processed by a power-law quantizer and a Huffman Coder. The scale factor amplification is used to take advantage of the non-uniform distribution of the coding noise provided by the power-law quantizer to better accommodate perceptual requirements [16]. The amplification factors (scale factors or scaling factors) applied to the scale factor bands are differentially coded also using Huffman Coding. The goal of the quantization stage is to determine the set of scaling factors that better accommodate the psychoacoustic model requirements and the bit availability at a certain rate.

The scale factor amplification and non-uniform quantization is carried out according to

$$\bar{X}(i) = sign(X(i)) \cdot nint\left(\left(\frac{|X(i)|}{\sqrt[4]{2}^{q_k}}\right)^{0.75} - 0.0946\right) \quad (1)$$

where $\bar{X}(i)$ is the value of the quantized *i-th* spectral line, $X(i)$ the *i-th* input MDCT spectral line and q_k the scale factor (amplification factor) associated to the *k-th* scale factor band. The span of $i : 0 < i < I_k - 1$ determines the amount of spectral lines per scale factor band (bandwidth), where I_k ranges from 4 to 52 for each *k-th* scale factor band. Expressions $sign()$ and $nint()$ represent the sign of the argument and the rounding operation to the closest integer respectively [17]. The power, root and division operations present in (1) conform one of the main bottlenecks on the computational load for low power devices [1].

The Fraunhofer AAC Encoder [18] features two different quality modes for the quantization stage: Fast Quality, where an open-loop solution is used to set each scale factor q_k based on *a priori* estimations of the non-uniform quantizer noise and the hearing thresholds provided by the psychoacoustic model, and High Quality, where the scale factors are set by refining the open loop estimation by Analysis-by-Synthesis (AbS) using two stages of iterative search. The AbS method can be seen as an AAC distortion-rate loop [15] and is described below.

2.1. Scale factor band optimization

The optimization method is based on a simple iterative search in the neighborhood of the initial scale factor value, in which a suitable scaling factor q_k that minimizes the quantity

$$D_{sfb}(q_k) = \sum_{\forall i \in \text{sfb}} |X(i) - \tilde{X}(i, q_k)|^2 \quad (2)$$

in each scale factor band (sfb) is chosen. The values of $\tilde{X}(i)$ represent the reconstructed spectral values after quantization using the inverse of (1). The quantity D_{sfb} given by equation (2) is defined as the distortion (or noise power) of each scale factor band caused by the quantization process and is used as a cost function for a series of iterative searches. This minimization is nevertheless restricted to keeping the noise-to-mask ratio (*nmr*) [19] just below a certain limit, and not much lower. If the *nmr* is too low, bits would be wasted in coding a band whose coding noise is already complying with the psychoacoustic rules. The method also considers this case when the starting distortion value is too low and tries to adapt it in order to save bits in coding. After a first calculation of distortion for the first estimation of q_k, the algorithm has two main branches:

First branch (adjust distortion): If the noise-to-mask ratio determined from the distortion is more than 1.25 (estimated experimentally), then try to improve it by searching for a q_k that minimizes D_{sfb} within $\mathbf{q_k} = [q_k - \nu_l, \ldots, q_k - 1, q_k, q_k + 1, \ldots, q_k + \nu_h]$, where ν_l, ν_h are the lower and higher limits of the search vector respectively.

Second Branch (adjust bitrate): The power-law quantizer of equation (1) provides coarser quantization steps for greater scale factors, coarser step sizes will need less bits to be coded. If the calculated *nmr* is less or equal to 1.25, the scale factor used can be increased in order to spare some bits and still comply with quantization noise masking rules. The search vector becomes then $\mathbf{q_k} = [q_k, q_k + 1, \ldots, \phi_h]$, where ϕ_h is the higher limit and the search stops when the resulting *nmr* is greater than 1.25.

Due to the fact that this iterative search requires the repeated re-quantization of the spectral lines $X(i)$ -via (1) and its inverse quantization counterpart [20]- to get $\tilde{X}(i)$ for each value of q_k the choice of ν_l, ν_h and $\phi_h$significantly impacts on the computational complexity.

2.2. Inter-band scale factor assimilation

Once the first set of scale factors has been determined, further iterative searches try to decrease the range of scale factor values across the scale factor bands in order to save bits on differential encoding. Given a set of scale factor bands, their difference in value is reduced and also adjusted to produce the smaller value of (2). Then, for further coding efficiency, additional smoothing along sets of scale factor bands is applied among these previously selected scale factors with the same criteria. In all cases, the same complexity considerations apply and the number of re-quantization operations grow further.

3. PARAMETRIZATION OF THE RATE-DISTORTION LOOP

Prior AAC encoder profiling [18] with the AbS feature switched on showed that the routines of distortion calculation, quantization and inverse quantization were the most cycle-demanding because of the many routine calls to (1) and (2). Distortion calculation and inverse quantization are only performed when the AbS mode is active. The most significant increase in cycle consumption takes place with the switch from no AbS to active AbS at around 40-60% extra cycles depending on the target platform.

3.1. Modification of the distortion calculation

As mentioned above, the evaluation of (2) has many calls throughout the code. It is worth noting that the exact absolute value of D_{sfb} is not important. A certain amount of error is permitted as long as the search algorithm ends at the right value of q_k (i.e. the scale factor that adjusts D_{sfb} to the better trade-off) within the search vector remains guaranteed. Based on this remark, the conditions on the calculation of (2) can be relaxed in favor of lesser computational cost.

3.1.1. Adaptive Threshold

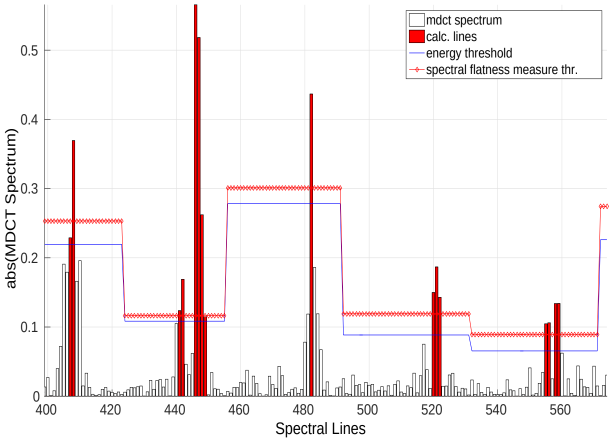

Figure 2: *Extract of the absolute value of the MDCT spectrum of a sample input signal. Lines of the spectrum that are considered for Analysis-by-Synthesis recalculation according to equation (5) ($j \in K_\tau$) and adaptive threshold τ_k (continuous line) for each sfb ($g_h = 0.5$) are coloured in dark red. Threshold based on classical spectral flatness measure [19] for reference (diamond line).*

One approach for reducing the amount of re-quantization can be calculating the distortion for a subset of lines on each scale factor band. The whole spectrum needs to be quantized at the end with the right scaling factor q_k, but for the purpose of the iterative search, only a representative portion of the spectrum can be used for estimating (2). This can greatly reduce the computational burden.

Since the power-law quantizer aims to evenly distribute the SNR across the dynamic range of the signal by applying a companding function to the quantization steps [16], the spectral lines with higher relative energy will contribute more to the round-off error than the lines with lesser energy.

It is well known that tonal signals concentrate the majority of their energy on narrow portions of their frequency spectrum. The lines corresponding to tonal portions (higher relative energy) will be the ones which contribute the most to the calculation of (2) within a scale factor band. It is therefore advantageous to rely on spectral flatness measures or tonality indices per band in order to identify these sections. The commonly used spectral flatness measure is the ratio of the geometric mean μ_g and the arithmetic mean μ_a of the power spectral density of each scale factor band k, $\text{SFM}_k = \frac{\mu_g}{\mu_a}$ [19]. A value of the SFM close to 1 means a flat "non-tonal" spectrum, whereas a value close to 0 means the presence of strong harmonic components corresponding to a tonal signal.

The selection of the subset of lines can be carried out with the aid of an adaptive threshold. This threshold is a fraction of the maximum value of $X(i)$ for a specific spectral band and is based on the estimated tonality for that particular region. The threshold τ_k is defined as:

$$\tau_k = g_h \cdot \max_{i \in \text{sfb}}[X(i)] \cdot (1 - \frac{nl_k}{I_k}) \qquad (3)$$

where I_k is the scale factor band width in spectral lines and nl_k is the number of lines in each scale factor band that will effectively be above some minimum quantization error value, marked as "relevant lines":

$$nl_k = \frac{\sum_{i=0}^{I_k-1} \sqrt{|X(i)|}}{(\frac{e_k}{I_k})^{0.25}} \tag{4}$$

where e_k is the total energy of the spectral band k.

The encoder already calculates nl_k for other purposes and its re-utilization saves processor cycles. For the case of the power-law quantizer of equation (1), an estimation of the quantization error can be made that depends on the quantizer step size q_k and a form factor of the spectral band expressed by the ratio between the geometric mean and the arithmetic mean [5] [14].

The spectral lines with amplitudes below the threshold τ_k will not take part on the distortion calculation. The more tonal the scale factor band, the higher the threshold (Figure 2). This means that only lines corresponding to the strongest harmonics will be calculated. From the figure it can also be seen that, -albeit with different scaling- both classical spectral flatness measures SFM$_k$ and τ_k given by equation (3) are equivalent for this purpose. The proposed tonality measure is used instead of classical approaches because most of the elements of equation (3) are already calculated for other purposes within the encoder. As a consequence some processing power is saved by not estimating tonality again in a different way.

The parameter g_h is an ad-hoc correction gain factor, hand-tuned in a way that the resulted calculated lines after the algorithm approaches the amount predicted by the number of relevant lines nl_k, restricted to the condition that $g_h \cdot (1 - \frac{nl_k}{I_k}) < 1$ so that at least one spectral line is calculated per sfb . It must be noted that other measurements of tonality can be used according to encoder implementation and the available data, and the tuning of the threshold gain g_h can accordingly vary.

3.1.2. Reformulation

The computational cost of calculating (2) for all the spectral lines in narrow bands (few frequency bins) becomes comparable to the one derived from using the threshold implementation. It was determined experimentally that the threshold implementation is more suitable if only used for spectral bands that have more than $I_k = 12$ lines because there is also a cost to implementing the threshold decision within the loop that scans every spectral line. Besides, skipping the calculation of some spectral bands for an already small set can lead to significant errors in (2) that will affect the convergence. Equation (2) then is reformulated as:

$$D_{\text{sfb}}(q_k) = \begin{cases} \sum_{\forall j \in K_\tau} |X(j) - \tilde{X}(j, q_k)|^2 & \text{if } I_k \geq 12 \\ \sum_{\forall i \text{sfb}} |X(i) - \tilde{X}(i, q_k)|^2 & \text{if } I_k < 12 \end{cases} \tag{5}$$

where $K_\tau \in$ sfb is the subset of spectral lines for which $X(i) > \tau_k$.

3.2. Taking advantage of signal stationarity

Another approach for avoiding the excessive re-quantization can be taking advantage of the relative stationarity of audio signals [19]. Once a given set of optimal scaling factors has been determined for an audio time frame, it is possible that the same set of optimal scale factors is already close to optimal for the next frame, given that the signal does not change significantly in that frequency region (Figure 3). Furthermore, this same principle can be applied to stereo signals that do not feature a significant inter-channel difference [21]. For two consecutive audio time or channel frames f and $f+1$, a preset threshold τ_s can be set to the condition:

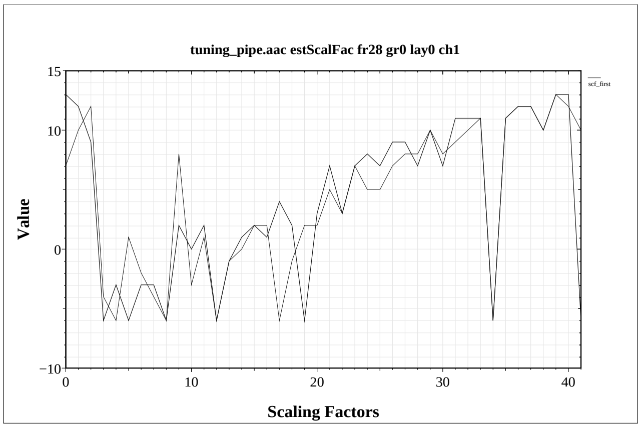

Figure 3: *Sets of scaling factors for coding the spectrum of a tuning pipe recording. Two consecutive audio frames are superimposed. The difference between frames greatly diminishes around the scale factor 28 and up (around 5 kHz and up at a frame size of 1024 and sampling rate of 48 kHz).*

$$|q_k^{(f+1)} - q_k^{(f)}| < \tau_s \tag{6}$$

for q_k factors of the same scale factor band k (frequency region). Only when the scale factor difference between two frames is bigger than the threshold will the AbS procedure described in section 2 take place. Otherwise, the scaling factor is considered close enough to optimal and the already quantized spectrum can be used. This method can be further refined to implement higher order temporal smoothing techniques, to the expense of additional memory usage: increasing the order F of the smoothing filter requires storing the complete set of scaling factors for each of the F previous frames.

4. RESULTS

This section encompasses results regarding trade-offs of the parameter tuning, complexity measurements and quality measurements.

4.1. Parameter Tuning

The parameters g_h and τ_s introduced in Sections 3.1 and 3.2 have limiting values. On one hand, the values can be set very low, so they do not have any influence on the re-quantization simplification and the complexity remains the same. In fact, these parameters can be set to generate a bit-exact stream with the reference version if set to $g_h = 0$, all the lines will then take part in the re-quantization of equation (5). By setting $\tau_s = 0$, the AbS routine takes place even if the scale factors remain the same within two consecutive frames.

On the other hand, if these parameters are set too high, the simplifications on the distortion calculation account for a coarse -and not so frequent- approximation of (2). This can result in audible artefacts with respect to the full AbS version, where all spectral lines are re-quantized in every audio frame. Figure 4 shows the

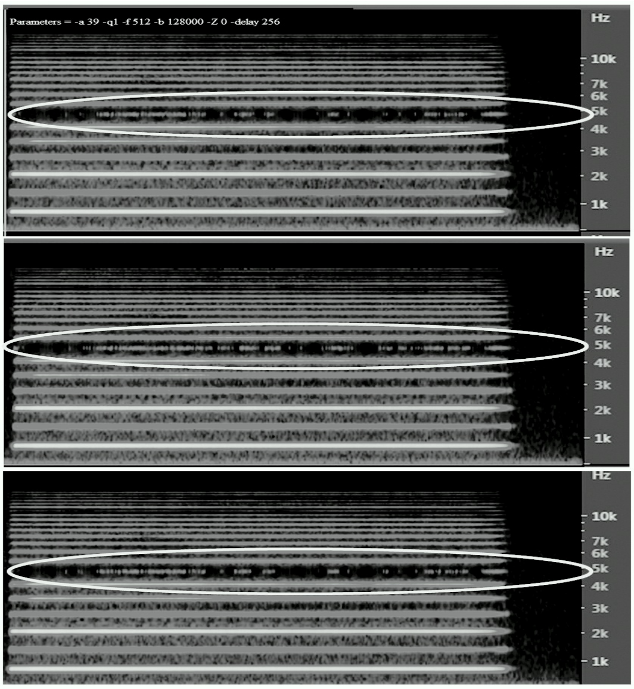

Figure 4: *Spectrogram of a coded tuning pipe mono audio signal at 48 kHz and 48 kbps with AAC-ELD. Above: Unmodified Analysis-by-Synthesis scale factor estimation. Middle: Scale factor smoothing in time (section 3.2) with $\tau_s = 3$. Below: Scale factor smoothing in time with $\tau_s = 1$.*

case where the time smoothing of section 3.2 is implemented with a value of τ_s that is too high (middle spectrogram). In comparison with the full AbS reference (upper spectrogram), more discontinuities can be seen in the spectral line of 5 kHz, which account for audible artefacts. This shows that the stationarity assumption can be overdone and inhibiting the AbS procedure can affect how the encoder works in small details. The lower spectrogram shows an optimal value of $\tau_s = 1$, where a lower complexity is still reached without affecting the sound quality.

Similar assumptions and procedures can be made with the threshold implementation of equation (5), where the optimal value was deemed to be around $g_h = 0.7$

4.2. Complexity

Table 1 shows two representative cases of complexity measurements for the different proposed modifications of section 3. The platforms tested are based on an ARM Cortex-A57 64-bit core and a Texas Instruments C6424 32-bit processor. All MHz values include all memory wait state and cache miss cycles and have been obtained with activated data and program cache. The different encoders do not contain any particular optimization that favors one particular version over the other. The selected encoding variant was AAC-ELD due to the shorter audio processing block length and therefore with tighter complexity requirements [9], [12]. A stereo file at 48 kHz sampling rate was encoded with a frame size (block length) of 512 samples. Encoder tools not relevant to the measurement were turned off in order to minimize the influence of other modules on the measurements. As a reference for a lower bound, workload was also measured for the encoder when no re-quantization routine is used (Condition 5). The frame smoothing procedure of equation (6) was used with a parameter of $\tau_s = 1$ and $g_h = 0.7$ for the adaptive threshold method.

Texas Instruments figures were obtained on a TMS320C6424 EVM evaluation board at a clock speed of 600 MHz.

ARM Cortex-A57 figures were obtained by running the software on a NVIDIA Jetson TX1 module running Linux Ubuntu (GNU/Linux 3.10.67 Kernel aarch64). We considered as the true reproducible value the minimum workload number for processing an audio frame during 10 consecutive encoder executions in order to filter out any influence of the operating system layer. Only one core was active and all power scaling functions were deactivated. The processor core at full performance was running at a CPU frequency of 1.91 GHz.

Indeed, Table 1 shows a general workload reduction for both architectures using one of the two methods, or a combination of both. Relative improvements might change in the presence of architecture-specific optimizations. For example, the significant jump in complexity between condition 5 and the other four conditions for the TI board is due to the poor handling the loops present on the AbS algorithm when no pragma directives are available [22]. As it will be shown in the next section, further reduction in workload can be achieved for mono files, where the encoder does not make use of joint stereo coding techniques [17] that might mitigate the workload reduction impact.

4.2.1. Complexity Scaling

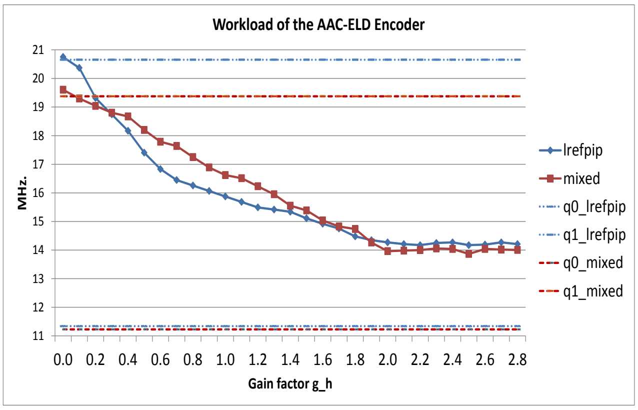

Figure 5: *Parametric workload curves for Cortex-A57 showing the influence of parameter g_h on encoding complexity. Two mono files (lrefpip: tuning pipe, mixed: castanets and guitars) at 48 kHz encoded with a block length of 512 samples. Terms $q0$ and $q1$ used to denote workload figures for no re-quantization at all and full unmodified re-quantization respectively.*

To further illustrate the influence of the proposed modifications on complexity, figure 5 shows parametric workload measurements performed on two different mono signals encoded with the same configurations as before. In contrast to the previous measurements, some compiler optimizations were active (-o3 switch on aarch64-linux-gnu-gcc 4.8.3) in order to provide insight on a real usage case scenario. The signal marked as *lrefpip* corresponds to the recording of a tuning pipe with a predominantly tonal structure, whereas the signal marked as *mixed* contains the recording of

Table 1: *Encoder Workload Measurements (Unoptimized).*

Condition	Encoder variant	Workload (MHz.)	Workload (MHz.)
		ARM Cortex-A57	TI C6424
1	Unmodified Re-quantization	41	74
2	Frame Smoothing	36	67
3	Adaptive Threshold	33	64
4	Fr. Smooth. and Adapt. Thr.	32	62
5	No Re-quantization	30	45

guitars and castanets, showing in this case an hybrid transient-like and tonal structure.

For $g_h = 0$ it can be seen that the workload corresponds to the full re-quantization case where all the lines are calculated. Already the full re-quantization shows varying complexity according to the signal (q1 lines). In the case of the hybrid signal, the content allows some relaxation on the iterative search for finding the minimum distortion, given that part of the coding noise will be masked by the loud transients. Conversely, the tonal structure of *lrefpip* and its relative quietness imposes a stricter requirement for reaching the minimal distortion on each scale factor band, therefore taking more time per frame and increasing workload. There is a small overhead for $g_h = 0$ with respect to the workload of the unmodified encoder on full re-quantization due to the calculation of τ_k.

As g_h increases, the number of spectral lines calculated for each iteration of the AbS algorithm diminishes according to (5). The signal characteristics also influence the way the total complexity is reduced: the encoding of the tonal signal shows a steeper reduction in workload when g_h is increased. This is because the modified re-quantization algorithm presented in section 3.1 only calculates the significant harmonics in each scale factor band. Such harmonics account for a few spectral lines per band that contain the most energy, and therefore convergence to the fitting scaling factor is guaranteed within a few calculations. On the contrary, the selectivity of the algorithm decreases for signals with a greater number of non tonal spectral bands, as is the case of the mixed signal. Most of the lines need to be calculated for non-tonal regions of the spectrum, a smaller workload reduction is achieved when increasing g_h.

The workload reduction reaches a limit for higher values of g_h. According to (5), only scale factor bands that have less than 12 spectral lines will be calculated in its entirety, and the rest of the bands will only be re-quantized with only one line per band in each iteration of the AbS search. Accordingly, this accounts for an offset with respect to the state where no re-quantization at all -at any of the bands- takes place (marked as q0 lines). In addition, even if the AbS iterative search is carried out re-quantizing as few lines as possible in each scale factor band, the whole spectrum needs to be re-quantized with the best suitable scale factors in the end. This final re-quantization of each band with its best scaling factor also contributes to the difference in workload with respect to the q0 operating points of the encoder for each signal.

4.3. Objective and Subjective Quality Grading

In order to evaluate the audio quality of the proposed parametrization, a Perceptual Evaluation of Audio Quality (PEAQ) [23] test automatically rated audio items from a database of 278 entries - music and speech recordings- coded with the different proposed encoder variants ($\tau_s = 1$ and two extreme values for g_h) and encoding at various bit rates and decoded using the same reference decoder.

Table 2: *Total average PEAQ degradation.*

Bitrate	Variant			
/chan	FS	AT ($g_h = 0.2$)	AT ($g_h = 1.9$)	NR (q0)
24000	0.032	0.007	0.014	0.108
32000	0.017	0.026	0.031	0.118
48000	0.180	0.010	0.098	0.344
96000	0.031	0.020	0.050	0.146

Table 3: *Tonal signals average PEAQ degradation.*

Bitrate	Variant			
/chan	FS	AT ($g_h = 0.2$)	AT ($g_h = 1.9$)	NR (q0)
24000	0.067	0.035	0.032	0.177
32000	0.027	0.031	0.093	0.332
48000	0.523	0.038	0.208	0.602
96000	0.092	0.027	0.108	0.291

Table 4: *Audio MUSHRA test items.*

Item	Description
35_short	xylophone
Hanco	jazz music
lrefhrp	harpsichord
lrefpip	tuning pipe
Mahle	orchestra recording
si02	castanets

Tables 2 and 3 show the average differential Objective Difference Grade (ODG) points with respect to the normal unmodified re-quantization operation, corresponding to the q1 operating point described in section 4.2. The encoder versions are, as before with frame smoothing (FS), adaptive threshold (AT) and no re-quantization at all (NR) as a lower quality bound, corresponding

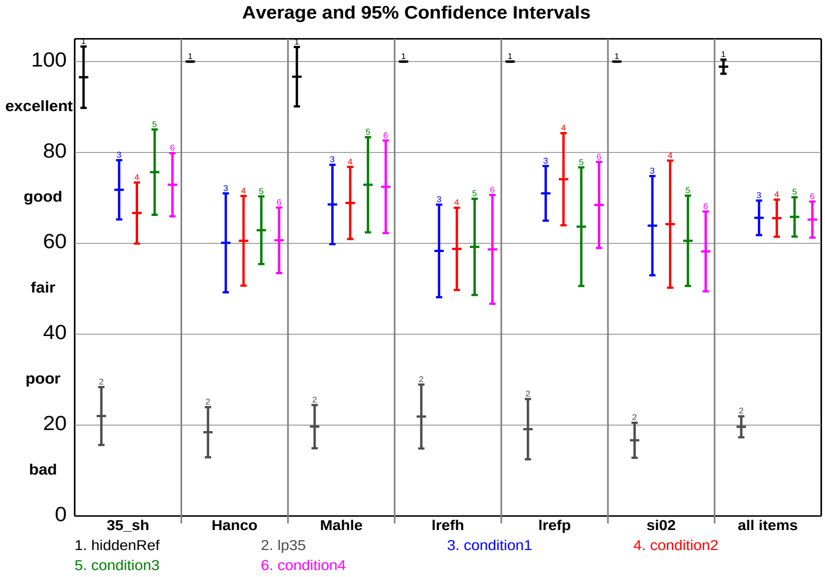

Figure 6: *Listening Test results for the different encoder modifications. Ten test subjects, seven expert listeners. Stereo files encoded with AAC-ELD, 96kbps (48 kbps per channel) at a sampling rate of 48 kHz.*

to the q0 operating point. When comparing Table 2 and 3 it can be seen that ODG differences are greater when only tonal-like signals are considered. The differential ODG values are transcribed without sign for the sake of clarity, but in all cases show a degradation in PEAQ scores with respect to the q1 operating point.

The items which presented the most degradation from the encoder reference were selected for taking part of a MUltiple Stimuli with Hidden Reference and Anchor (MUSHRA) listening test [24]. A short description of the audio items can be found in Table 4. The stereo files were encoded at a bitrate of 48 kbps per channel (96 kbps stereo) again using the AAC-ELD codec.

Figure 6 shows the results of the MUSHRA test for the different encoder variants with re-quantization active. An anchor item that corresponds to a low-pass filtered version of the original reference at 3.5 kHz is marked as "lp35". Condition 5 -no re-quantization- was not included in the listening test since it is considered to operate at a lower quality range, and not meant to be perceptually equivalent to the other versions. There were 10 test subjects, 7 of which are expert listeners. For reproducing the sound, Stax electrostatic headphones and amplifier were used in an acoustically controlled environment. As it can be seen from the test data, from particular items and total, there is no significant perceptual difference between encoder versions.

5. CONCLUSIONS

The error-resilient nature of audio coding algorithms permits a greater headroom for complexity reduction than other signal processing algorithms that do not make use of perceptual rules. In this case, we have shown that the perceived quality of the AbS algorithm in the quantization stage of our AAC-Encoder version [18] does not significantly change, even when the algorithm complexity is notably reduced (up to 20% of the total complexity or 80% of added complexity by AbS on a stereo file, without any hardware optimization). As already discussed in [1] and confirmed here, bit-exactness of the encoder output between versions is not the best figure of merit for conditioning the optimization work. Objective and subjective perceptual evaluation should also be performed in workload reduction strategies aimed to low power implementations. This "perceptually aware" optimization possibility is usually not recognized in later stages of implementation, where all reference algorithms are considered to be optimally tuned, even when computational requirements have not yet been thoroughly assessed.

The implementation stage of algorithms that make use of human perceptual models must be thoroughly evaluated, even during the final stages where usually only architectural optimizations take place. This approach can make a considerable difference when all architecture specific improvements are not enough in order to reach strict workload requirements.

On the case of mono files where joint stereo coding methods are not present, the relative workload reduction is around 30% for the AbS algorithm within perceptual equivalence to the unmodified version. Nevertheless, parametrizing algorithms under complexity-distortion trade-offs can be considered as extra flexibility on top of the work done already if perceptual equivalence does not need to be met. Future work includes trying to design a self-scaling algorithm that implements automated control on these parameters based on instantaneous workload measurements, given experimentally determined limits in section 4.1 and further investigation on parametrizing other modules of the encoder.

6. REFERENCES

[1] Markus Lohwasser, Marc Gayer, and Manfred Lutzky, “Implementing MPEG Advanced Audio Coding and Layer-3 encoders on 32-bit and 16-bit fixed-point processors,” in *Audio Engineering Society Convention 115*, Oct 2003.

[2] Y. Andreopoulos, “Error tolerant multimedia stream processing: There’s plenty of room at the top (of the system stack),”

Multimedia, IEEE Transactions on, vol. 15, no. 2, pp. 291–303, Feb 2013.

[3] M.A. Anam, P.N. Whatmough, and Y. Andreopoulos, “Precision-energy-throughput scaling of generic matrix multiplication and discrete convolution kernels via linear projections,” in *Embedded systems for real-time multimedia (ESTIMedia), 2013 IEEE 11th Symposium on*, Oct 2013, pp. 21–30.

[4] V.K. Goyal and M. Vetterli, “Computation-distortion characteristics of block transform coding,” in *Acoustics, Speech, and Signal Processing, 1997. ICASSP-97., 1997 IEEE International Conference on*, Apr 1997, vol. 4, pp. 2729–2732 vol.4.

[5] M. Bosi and R. Goldberg, *Introduction to digital audio coding and standards*, chapter Psychoacoustic models for audio coding, pp. 179–200, Kluwer Academic Publishers, Dordrecht, The Netherlands, 2003.

[6] Max Neuendorf et al., “MPEG Unified Speech and Audio Coding - The ISO/MPEG standard for high-efficiency audio coding of all content types,” in *Audio Engineering Society Convention 132*, Apr 2012.

[7] Jürgen Herre, Johannes Hilpert, Achim Kuntz, and Jan Plogsties, “MPEG-H audio - the new standard for universal spatial/3D audio coding,” *J. Audio Eng. Soc*, vol. 62, no. 12, pp. 821–830, 2015.

[8] Eric Allamanche, Ralf Geiger, Juergen Herre, and Thomas Sporer, “MPEG-4 low delay audio coding based on the AAC codec,” in *Audio Engineering Society Convention 106*, May 1999.

[9] Manfred Lutzky, María Luis Valero, Markus Schnell, and Johannes Hilpert, “AAC-ELD V2 - the new state of the art in high quality communication audio coding,” in *Audio Engineering Society Convention 131*, Oct 2011.

[10] 3GPP TS 26.445, “Codec for Enhanced Voice Services (EVS); detailed algorithmic description,” Tech. Rep., 3GPP Technical Specification (Release 15), 2015.

[11] Ralf Geiger, Manfred Lutzky, Markus Schmidt, and Markus Schnell, “Structural analysis of low latency audio coding schemes,” in *Audio Engineering Society Convention 119*, Oct 2005.

[12] Johannes Hilpert, Marc Gayer, Manfred Lutzky, Thomas Hirt, Stefan Geyersberger, and Josef Hoepfl, “Real-time implementation of the MPEG-4 Low-Delay Advanced Audio Coding algorithm (AAC-LD) on Motorola’s DSP56300,” in *Audio Engineering Society Convention 108*, Feb 2000.

[13] Xiaopeng Hu, Guiming He, and Xiaoping Zhou, “An efficient low complexity encoder for MPEG Advanced Audio Coding,” in *Advanced Communication Technology, 2006. ICACT 2006. The 8th International Conference*, Feb 2006, vol. 3, pp. 1500–1505.

[14] E. Alexandre, A. Pena, and M. Sobreira, “Low-complexity bit-allocation algorithm for MPEG AAC audio coders,” *Signal Processing Letters, IEEE*, vol. 12, no. 12, pp. 824–826, Dec 2005.

[15] S. Nithin, T. V. Sreenivas, and Kumaraswamy Suresh, “Low complexity bit allocation algorithms for MP3/AAC encoding,” in *Audio Engineering Society Convention 124*, May 2008.

[16] M. Bosi and R. Goldberg, *Introduction to digital audio coding and standards*, chapter Quantization, pp. 13–34, Kluwer Academic Publishers, Dordrecht, The Netherlands, 2003.

[17] M. Bosi and R. Goldberg, *Introduction to digital audio coding and standards*, chapter MPEG2-AAC, pp. 346–350, Kluwer Academic Publishers, Dordrecht, The Netherlands, 2003.

[18] Fraunhofer IIS, “A standalone library of the Fraunhofer FDK AAC code for Android,” Available at https://android.googlesource.com/platform/external/aac/+/master, Accessed February 19, 2016.

[19] T. Painter and A. Spanias, “Perceptual coding of digital audio,” *Proceedings of the IEEE*, vol. 88, no. 4, pp. 451–515, April 2000.

[20] ISO/IEC 14496-3:2009, “Information technology - Coding of audio-visual objects - Part 3: Audio,” Tech. Rep., ISO/IEC, 2009.

[21] C.R. Helmrich, P. Carlsson, S. Disch, B. Edler, J. Hilpert, M. Neusinger, H. Purnhagen, N. Rettelbach, J. Robilliard, and L. Villemoes, “Efficient transform coding of two-channel audio signals by means of complex-valued stereo prediction,” in *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*, May 2011, pp. 497–500.

[22] Texas Instruments, *Introduction to TMS320C6000 DSP Optimization (app. note SPRABF2)*, Oct 2011, Application Report.

[23] ITU-R BS.1387-1, “Method for objective measurements of perceived audio quality,” Tech. Rep., ITU,Geneva, Switzerland, 2001.

[24] ITU-R BS.1116-3, “Methods for the subjective assessment of small impairments in audio systems,” Tech. Rep., ITU,Geneva, Switzerland, 2015.