# Arithmetic Coding of Sub-band Residuals in FDLP Speech/Audio Codec

*Petr Motlicek*[1], *Sriram Ganapathy*[2], *Hynek Hermansky*[2]

[1]Idiap Research Institute, Martigny, Switzerland
[2] ECE Dept., Johns Hopkins University, Baltimore, USA
motlicek@idiap.ch, {sganapa5,hynek}@jhu.edu

## Abstract

A speech/audio codec based on Frequency Domain Linear Prediction (FDLP) exploits auto-regressive modeling to approximate instantaneous energy in critical frequency sub-bands of relatively long input segments. The current version of the FDLP codec operating at 66 kbps has been shown to provide comparable subjective listening quality results to state-of-the-art codecs on similar bit-rates even without employing standard blocks such as entropy coding or simultaneous masking. This paper describes an experimental work to increase compression efficiency of the FDLP codec by employing entropy coding. Unlike conventional Huffman coding employed in current speech/audio coding systems, we describe an efficient way to exploit arithmetic coding to entropy compress quantized spectral magnitudes of the sub-band FDLP residuals. Such an approach provides 11% ($\sim$ 3 kbps) bit-rate reduction compared to the Huffman coding algorithm ($\sim$ 1 kbps).

**Index Terms**: Audio Coding, Frequency Domain Linear Prediction (FDLP), Entropy Coding, Arithmetic Coding, Huffman Coding

## 1. Introduction

Traditionally, a two-step process is carried out to perform source coding of analog audio/visual input signals. First, a lossy transformation of the analog input data into a set of discrete symbols is performed. Second, lossless compression, often referred to as noiseless/entropy coding, is employed to further improve compression efficiencies. In many current audio/video codecs, such a distinction does not exist, or only one step is applied [1].

In conventional speech/audio compression applications, entropy coding is carried out by Huffman coding techniques (e.g., [2, 3]). Either the source symbols are compressed individually, or they are grouped to create symbol strings which are then processed by a vector based entropy coder. Since the entropy of the combined symbols is never higher than the entropy of the elementary symbols (usually it is significantly lower), a high compression can be achieved: nearly $2:1$ in the case of variable-length Huffman coding employing several codebooks [2]. However, a considerable lookahead is required. Therefore, vector based entropy coding is usually exploited for high quality speech/audio coding where an algorithmic delay is available.

Recently, a new speech/audio coding technique based on approximating temporal evolution of the spectral dynamics was proposed [4, 5]. The compression strategy is based on predictability of slowly varying amplitude modulations to encode speech/audio signals. On the encoder side, an input signal is split into frequency sub-bands following critical sub-band decomposition. In each sub-band, the Hilbert envelope is estimated using Frequency Domain Linear Prediction (FDLP), which is an efficient technique for Auto-Regressive (AR) modeling of temporal envelopes of a signal [6]. Sub-band FDLP residuals are processed using the Discrete Fourier Transform (DFT). Magnitude and phase spectral components are quantized using Vector Quantization (VQ) and Scalar Quantization (SQ), respectively. The process of quantization is controlled by a perceptual model simulating temporal masking. The decoder inverts the steps from encoder to reconstruct the signal back.

In this paper, we describe noiseless coding experiments performed to efficiently encode selected codebook indices obtained using VQ, and thus to improve overall compression efficiency of the FDLP speech/audio codec. More particularly, VQ is employed to quantize magnitude spectral components of the sub-band FDLP residuals. Sufficiently low quantization noise as well as acceptable computational load is achieved by split VQ [7]. It provides significantly higher SNRs compared to simple scalar (per-symbol) quantization. However, Huffman coding, successfully applied on scalar quantized spectral coefficients (e.g., in an AAC system), does not bring any additional compression when combined with split VQ in the FDLP codec. This is due to the fact that VQ has already removed most of the redundancy in the encoded data. Therefore, we propose to use another entropy coding technique, arithmetic coding [8], to be employed to operate on top of split VQ indices. Due to a minor time correlation of phase spectral components, only magnitude spectral components of sub-band FDLP residuals are quantized using efficient split VQ and further entropy encoded.

Since arithmetic coding has advantageous properties for small alphabets [9], VQ codebooks are first pruned down (without the significant increase of quantization noise). Input sequences provided by successive VQ indices are then split into two sub-streams (with reduced alphabets) which are then independently entropy compressed.

The compression efficiency of the extended arithmetic coding technique employed in the FDLP codec is emphasized by comparing ratios with the Huffman coding algorithm on challenging speech/audio data.

## 2. Structure of the FDLP codec

The FDLP codec is based on processing relatively long temporal segments. As described in [5], the full-band input signal is decomposed into non-uniform frequency sub-bands. In each sub-band, FDLP is applied and Line Spectral Frequencies (LSFs) approximating the sub-band temporal envelopes are

This work was partially supported by the Swiss National Center of Competence in Research (NCCR) on “Interactive Multi-modal Information Management (IM)2”; managed by the Idiap Research Institute on behalf of the Swiss Federal Authorities.

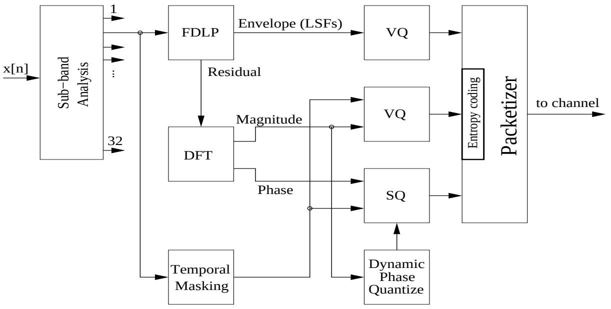

Figure 1: *Scheme of the FDLP encoder with block of entropy coding.*

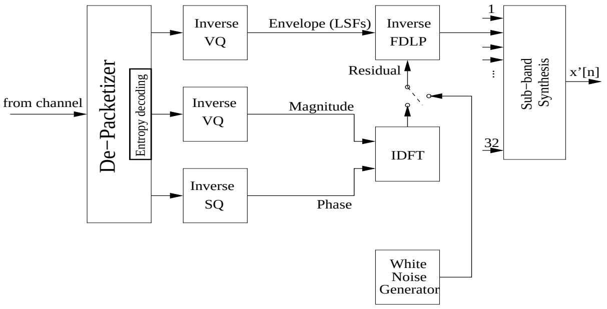

Figure 2: *Scheme of the FDLP decoder with block of entropy decoding.*

quantized. The residuals (sub-band carriers) are obtained by filtering sub-band signals through a corresponding AR model reconstructed from the quantized LSF parameters (quantization noise is introduced using an analysis-by-synthesis approach). Then these sub-band residuals are segmented into sub-segments and processed in the DFT domain. Magnitude and phase spectral components are quantized using VQ and SQ, respectively. A graphical scheme of the FDLP encoder is given in Fig. 1.

In the decoder, shown in Fig. 2, quantized spectral components of the sub-band carriers are reconstructed and transformed into the time-domain using the inverse DFT. The reconstructed FDLP envelopes (from LSF parameters) are used to modulate the corresponding sub-band carriers. Finally, sub-band synthesis is applied to reconstruct the full-band signal. The final version of the FDLP codec operates at 66 kbps.

Among the important blocks of the FDLP codec belong:

- *Non-uniform QMF decomposition*: A perfect reconstruction filter-bank is used to decompose a full-band signal into 32 (critically band-sized) frequency sub-bands.
- *Temporal masking*: A first order forward masking model of the human hearing system is implemented and employed in encoding the sub-band FDLP residuals.
- *Dynamic Phase Quantization (DPQ)*: DPQ represents a special case of magnitude-phase polar quantization, which enables to better control the selection of scalar quantization levels and thus to reduce the bit-rate consumption of phase spectral components.
- *Noise substitution*: FDLP filters in frequency sub-bands above 12 kHz (last 3 sub-bands) are excited by white noise in the decoder. This has shown to have a minimal impact on the quality of reconstructed signal [5].

### 2.1. Quantization of spectral magnitudes and phases in the FDLP codec

Spectral magnitudes together with corresponding phases represent 200 ms long sub-segments of the sub-band FDLP residuals. At the encoder side, spectral magnitudes are quantized using VQ (corresponding codebooks generated using the LBG algorithm).

VQ is a well known technique which provides the best quantization scheme for a given bit-rate. However, a full-search VQ exponentially increases computational and memory requirements of vector quantizers with the bit-rate. Moreover, usually a large amount of training data is required. Therefore, a sub-optimal (split) VQ is employed in the FDLP codec. Each vector of spectral magnitudes is split into a number of sub-vectors and these sub-vectors are quantized separately (using separate VQ). Due to the unequal width of frequency sub-bands introduced by the non-uniform QMF decomposition, the vector lengths of spectral magnitudes differ in each sub-band. Therefore, the number of splits differs, as well. In addition, more precise VQ (more splits) is performed in lower frequency sub-bands where the quantization noise has been shown to be more perceptible than in higher sub-bands.

Finally, codebook pruning is performed in the lower frequency sub-bands (bands $1-26$) in order to reduce their size and to speed up VQ search. Objective quality evaluations proved that $25\%$ codebook reduction (i.e., the least used centroids are removed based on the statistical distribution estimated on training data) has a minimum impact on resulting quality.

The distribution of phase spectral components of the sub-band FDLP residuals was found to be close to uniform, thus their correlation across time is minor. A uniform SQ is performed (controlled by DPQ block) without applying additional entropy coding.

## 3. Arithmetic Coding

Unlike Huffman coding, Arithmetic Coding (AC) was found to significantly increase the compression efficiency of the FDLP speech/audio codec. The main advantage of AC is that it can operate with symbols (to be encoded) with a fractional number of bits [8], as opposed to well-known Huffman coding. In general, AC can be proven to reach the best compression ratio possible introduced by the entropy of the data being encoded. AC is superior to the Huffman method and its performance is optimal without the need for grouping of input data. AC is also simpler to implement since it does not require building a tree structure. A simple probability distribution of input symbols needs to be stored at encoder and decoder sides, which possibly requires dynamic modifications based on input data to increase compression efficiency.

AC processes the whole sequence of input symbols in one time by encoding symbols using fragments of bits. In other words, AC represents an input sequence by an interval of real numbers between 0 and 1. As a sequence becomes longer, the interval needed to represent this sequence becomes smaller. Therefore, the number of bits to specify the given interval grows.

Nowadays, AC is being used in many applications, especially those with small alphabets (or with unevenly distributed probabilities) such as compression standards G3 and G4 used for fax transmission. In these cases, AC is maximally efficient compared to the Huffman coding algorithm. It can be shown that Huffman coding never overcomes a compression ratio of $(0.086 + P_{max})H_M(S)$ for an arbitrary input sequence

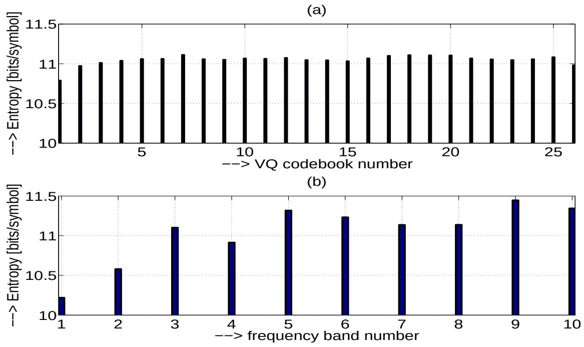

Figure 3: *Mean entropy of VQ indices of the first 10 sub-bands estimated: (a) for each VQ codebook (codebook dependent sequences), (b) for each sub-band (sub-band dependent sequences).*

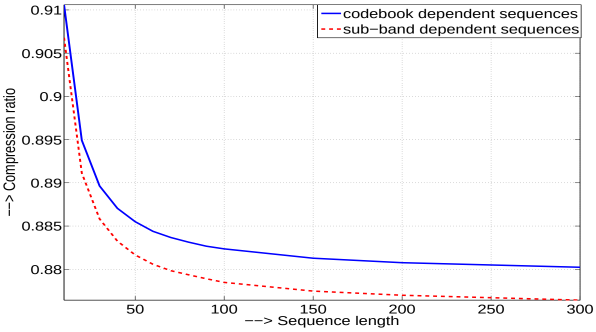

Figure 4: *Compression ratio of arithmetic coder for different lengths of input sequences. Input sequences are generated: (a) for each codebook (codebook dependent sequences), (b) for each sub-band (sub-band dependent sequences).*

$S$ with $P_{max}$ being the largest of all occurring symbol probabilities [10]. $H_M(S)$ denotes the entropy of the sequence $S$ for a model $M$. It is obvious that for large alphabets, where $P_{max}$ reaches relatively small values, the Huffman algorithm achieves better compression efficiencies. Therefore, this justifies such a technique on large alphabets. However, for small alphabet applications, which lead to bigger symbol occurrence probabilities, AC is more efficient.

### 3.1. Experimental data

Entropy coding experiments are performed on audio/speech data sampled at $48$ kHz. In all experiments, fixed model based entropy coding algorithms are used. Unlike the Huffman algorithm, which requires generating a tree structure shared by the encoder and the decoder, AC requires only probabilities of input symbols to be estimated from training data.

In our experiments, the training data consists of $47$ speech/audio recordings ($19.5$ minutes), mainly downloaded from several internet databases. The content consists of speech, music and radio recordings. Test data consists of $28$ recordings ($7.25$ minutes) with mixed signal content from the MPEG database for “explorations in speech and audio coding” [11].

### 3.2. Experimental setup

Entropy coding is applied on spectral magnitudes of the sub-band FDLP residuals in all $32$ sub-bands. The size of VQ codebooks employed in the FDLP codec differs for lower and higher frequency bands. Codebooks in bands 1-26 and 27-32 contain $3096$ and $512$ centroids, respectively. This corresponds to $11.5962$ bits/symbol and $9$ bits/symbol, respectively.

Several experiments are conducted to optimize the performance of AC. In these experiments, VQ indices (symbols) only from the first $10$ sub-bands ($0 \sim 4$ kHz) are used to form the input sequences for AC. Sub-bands $1-10$ utilize $26$ (band independent) VQ codebooks to quantize magnitude spectral components. Since AC operates over sequences of symbols, it matters how these symbol sequences are generated. We experiment with two ways:

- Input sequences comprise symbols generated by the same VQ codebook (codebook dependent sequences): A fixed probability model *for each VQ codebook* is estimated from training data. Mean entropy estimated from training data is shown in Fig. 3 (a). Different lengths of input test sequences are created from test data to be encoded by AC. Compression ratios achieved for different test sequence lengths are shown in Fig. 4.
- Input sequences comprise symbols belonging to the same sub-band (sub-band dependent sequences): A fixed probability model *for each sub-band* is generated from training data. Mean entropy estimated from training data is shown in Fig. 3 (b). Compression ratios achieved for different test sequence lengths created from test data are given in Fig. 4.

The compression ratios given in Fig. 4 clearly show that AC is more efficient in the second mode, i.e., when applied independently in each frequency sub-band. This means that entropy coding can better exploit similarities in the input data distribution generated by that frequency sub-band.

With respect to the theoretical insights of AC mentioned in Sec. 3, we further perform alphabet reduction. It is achieved by splitting each input sequence comprising 12-bit symbols into two independent 6-bit symbol sub-sequences. Training data is used to estimate two independent probability models from 6-bit symbol distributions. During encoding, each input test sequence of 12-bit symbols is split into two 6-bit symbol sub-sequences which are then encoded independently by two ACs employing two different probability models. Finally, the obtained compressed bit-streams are merged to create one bit-stream to be transmitted over the channel. Compression ratios achieved (for the first 10 sub-bands) are given in Fig. 5. This figure compares performances for the case when AC employs the reduced and the full alphabet. As can be seen, the proposed alphabet reduction provided by splitting of 12-bit symbol sequences into two 6-bit symbol sub-sequences significantly increases compression efficiency.

## 4. Experimental results

Sec. 3.2 describes the experimental procedure to exploit AC in the FDLP codec. These experiments were performed with data (VQ indices) coming from the first $10$ frequency sub-bands. The best performance was obtained for the case when AC was applied independently in each frequency sub-band (regardless of VQ codebook assignment). Furthermore, the reduced alphabet provided better compression efficiency in all frequency sub-bands compared to the full alphabet. Next, this configuration is used to test the efficiency of AC applied to encode VQ indices from all $32$ frequency sub-bands. The resulting compression

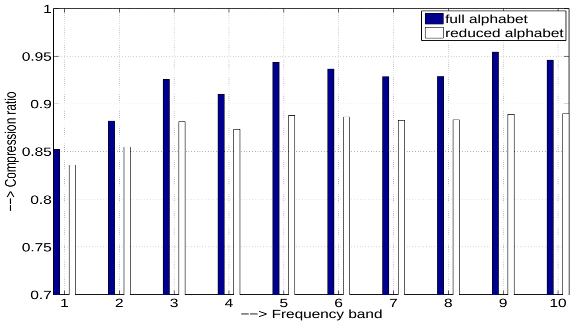

Figure 5: *Compression ratio of arithmetic coder operating on the full and the reduced alphabet.*

ratios are given in Fig. 6. In these experiments, test sequence lengths are chosen to equal $50$ (number of successive VQ indices forming input sequences for AC).

Further, the AC performance is compared to the performance of Huffman coding, traditionally applied in state-of-the-art speech/audio systems. The same training data is used to generate a fixed model provided by a tree structure shared by the Huffman encoder and the decoder. Since better Huffman coding performance is obtained for large alphabets [10], the original $12$-bit alphabet is used. Similar to AC, Huffman coding is also applied independently in each frequency sub-band (the Huffman tree structure is generated for each frequency sub-band). The performance of the Huffman based entropy coder for different frequency sub-bands is also given in Fig. 6.

## 5. Discussions and conclusions

In this paper, an entropy coder based on Arithmetic Coding (AC) algorithm is proposed to be implemented in an FDLP speech/audio codec operating at $66$ kbps. Only VQ codebook indices of magnitude spectral components of the sub-band FDLP residuals from $0$ to $12$ kHz were entropy encoded. Overall bit-rate reduction achieved by AC is $3$ kbps. This corresponds to an $11\%$ bit-rate reduction to compress VQ indices of spectral magnitudes. AC outperforms traditional Huffman coding, which provides $1$ kbps bit-rate reduction. Although AC requires a sequence of symbols to be encoded at the input, it does not increase the computational delay of the whole system. The decoding can start immediately with the first bits transmitted over the channel.

One can see in Fig. 4 that the compression efficiency of AC increases with the length of the input sequence. Due to applying relatively long temporal analysis in the FDLP codec, AC algorithm provides an efficient solution to perform subsequent entropy compression. In our work, AC did not exploit the adaptive probability model, which could significantly increase performance. In this case, AC would be a powerful technique, which would not require complex changes of the structure, as opposed to the Huffman coding.

Objective and subjective listening tests were performed and described in [5] to compare FDLP codec with LAME-MP3 (MPEG $1$ Layer $3$) [12] and MPEG-4 HE-AAC v1 [13], both operating at $64$ kbps. Since AC is a lossless technique, previously achieved audio quality results are valid. Overall, the FDLP speech/audio codec achieves similar subjective qualities to the state-of-the-art codecs on medium bit-rates. Currently, a low-delay version of the FDLP codec is being developed to operate on lower variable ($32 - 64$ kbps) bit-rates. AC based entropy coding will represent the important part in this codec for increasing the compression efficiency.

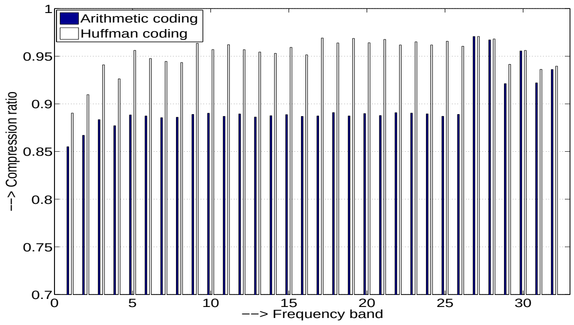

Figure 6: *Compression ratio of arithmetic and Huffman coding for different frequency sub-bands.*

## 6. References

[1] P. A. Chou, T. Lookabaugh, R. M. Gray, "Entropy Constrained Vector Quantization", *in Trans. Acoust. Sp. and Sig. Processing*, 37(1), January 1989.

[2] S. R. Quackenbush, J. D. Johnston, "Noiseless coding of quantized spectral components in MPEG-2Advanced Audio Coding", *in IEEE Workshop on Appl. of Signal Proc. to Audio and Acoustics*, New Paltz, USA, October 1997.

[3] Y. Shoham, "Variable-size vector entropy coding of speech and audio", *in International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 769-772, Salt Lake City, USA, May 2001.

[4] P. Motlicek, H. Hermansky, H. Garudadri, N. Srinivasamurthy, "Speech Coding Based on Spectral Dynamics", Proceedings of TSD 2006, LNCS/LNAI series, Springer-Verlag, Berlin, pp. 471-478, September 2006.

[5] S. Ganapathy, P. Motlicek, H. Hermansky, H. Garudadri, "Autoregressive Modelling of Hilbert Envelopes for Wide-band Audio Coding", *Audio Engineering Society*, 124th Convention, Amsterdam, Netherlands, May 2008.

[6] M. Athineos, D. Ellis, "Frequency-domain linear prediction for temporal features", *Automatic Speech Recognition and Understanding Workshop IEEE ASRU*, pp. 261-266, December 2003.

[7] K. Paliwal, B. Atal, "Efficient Vector Quantization of LPC Parameters at 24 Bits/Frame", *in IEEE Trans. on Speech and Audio Processing*, vol. 1, pp. 3-14, January 1993.

[8] J. Rissanen, G. Langdon, Jr., "Arithmetic Coding", *in IBM Journal of Res. & Dev.*, vol. 23, No. 2., 3/79.

[9] E. Bodden, M. Clasen, and J. Kneis, "Arithmetic Coding revealed - A guided tour from theory to praxis", *Technical report*, Sable Research Group, McGill University, No. 2007-5, 2007.

[10] K. Sayood. "Introduction to data compression", (2nd ed.), Morgan Kaufmann Publishers Inc., 2000.

[11] ISO/IEC JTC1/SC29/WG11: "Framework for Exploration of Speech and Audio Coding", MPEG2007/N9254, Lausanne, Switzerland, July 2007.

[12] LAME MP3 codec: $<$*http://lame.sourceforge.net*$>$

[13] 3GPP TS 26.401: "Enhanced aacPlus General Audio Codec".