Combined Source Adaptive and Channel Optimized Matrix Quantization Algorithm

Dr. Vasilios Bozantzis, MIET
Computer Engineering Dept.,School of Applied Technology
TEI of Peloponnesus (ex-TEI of Kalamata)
Sparti, Greece
vbozantzis@teikal.gr

Dr. Panos Philippopoulos, Assistant Professor
Computer Engineering Dept.,School of Applied Technology
TEI of Peloponnesus (ex-TEI of Kalamata)
Sparti, Greece
pfilip@teikal.gr

Abstract*—Matrix Quantization (MQ), a very promising source coding technique, has already been successfully applied for speech signals and noiseless channels. MQ is also shown in the literature to outperform Vector Quantization (VQ) when applied over noisy channels. Considering that most sources of practical interest are non-stationary, this paper introduces a technique which adapts MQ to varying source statistics and optimizes MQ for noisy channels, thus designs a matrix quantizer/decoder that considers both non-stationary source and noisy channel statistics. The resulting algorithm, Combined Source Adaptive and Channel Optimized Matrix Quantization (CSACOMQ) is evaluated for a source modelled as the non-stationary Wiener process and over the memoryless Binary Symmetric Channel (BSC). It is shown that CSACOMQ offers substantial Signal-to-Noise Ratio (SNR) performance improvement compared to the Channel Optimized Matrix Quantization (COMQ).

Keywords*—Matrix Quantization, Channel Optimization, Adaptive Source Coding, Combined Source-Channel Coding.

I. INTRODUCTION

VQ and Adaptive Vector Quantization (AVQ) [1] have been extensively studied and used in practical applications, over the past decade. Applications of VQ and AVQ include modern audio codecs, such as the AMR-WB+ [2], the Broadvoice audio codec [3], the CELT audio codec [4] and the OPUS interactive audio codec [5, 6]. AVQ has been applied, among others, in video compression [7], where the compression scheme is based on adaptive vector quantization of multi-wavelet coefficients, and in image compression [8], where a fuzzy self-adaptive particle swarm optimization algorithm is described extracting a near-optimum VQ codebook. In [9], an AVQ scheme for speech coding is presented, where the VQ of the Line Spectral Frequency (LSF) parameters of speech is optimized for the probability density function (p.d.f.) of the LSF parameters using a mixture of Dirichlet distributions. In [10] a reordering VQ algorithm for temporally structured sources is presented, which outperforms mainstream VQ algorithms, when tested for Markov and speech sources.

In [11,12] Bozantzis et al. presented the Combined Source Adaptive and Channel Optimized Vector Quantization (CSACOVQ) algorithm, which jointly adapts VQ to changing source statistics and optimizes it for the BSC and the Flat Fading Rayleigh Channel (FFRC) respectively. In [13] Bozantzis et al. produce a novel and efficient approach on the Index Assignment (IA) scheme design for VQ. Finally, in [14], Bozantzis et al. present the joint VQ/IA design, which in a combined manner, is adapted to changing source statistics and is optimized for the noisy channels considered in [14].

Additionally to the well-known VQ, MQ has attracted research interest and is shown to be a very promising source coding technique, compared to VQ. In [15] it is shown that MQ can exploit better than VQ the correlation properties between consecutive speech frames with a multi-frame structure. In [16] the COMQ technique is presented, applied for the coding of Line Spectral Pair (LSP) parameters transmitted over noisy channels and is shown to outperform Channel Optimized Vector Quantization (COVQ) [17].

In this paper initially an Adaptive Matrix Quantization (AMQ) technique is presented, which, in accordance to the concept of AVQ, adapts the matrix quantizer/decoder to varying source statistics as the coding process progresses. The technique is based on the move-to-front generalized threshold replenishment (GTR) algorithm for VQ [18]. This AMQ technique alters the matrix quantizer/decoder codebooks for every transmitted matrix, according to a decision met at the transmitter side.

In the sequel, the CSACOMQ algorithm is introduced, which combines the AMQ technique with the COMQ algorithm. CSACOMQ jointly adapts a matrix quantizer/decoder to changing source statistics and optimizes it for a noisy channel, thus acting as a form of combined source-channel coding. In Section II the move-to-front GTR algorithm for MQ is introduced and described providing the principles and preliminaries with the main system components and functionalities. Subsequently, in Section III, the CSACOMQ algorithm is presented, which jointly designs a matrix quantizer/decoder for changing source statistics and the BSC. Finally, in Section IV, simulation results are presented and discussed, and conclusions are drawn in Section V.

II. ADAPTIVE MATRIX QUANTIZATION

The design objective of the move-to-front GTR algorithm for MQ is the adaptation of the matrix quantizer/decoder pair to varying source statistics. It is best described with the following block diagrams. Matrix encoder/decoder pair is depicted in Fig.1, while vector/matrix encoder structure is depicted in Fig.2.

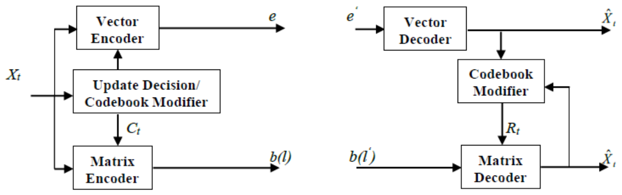

Figure 1. Structure of the Matrix Encoder/Decoder

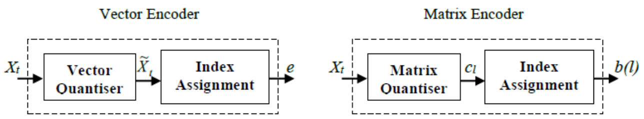

Figure 2. Vector Encoder and Matrix Encoder

At each time instance t, only one of the vector encoder/decoder or matrix encoder/decoder lines is active, according to a decision made by the Update Decision/Codebook Modifier block. More specifically, the source is a non-stationary, continuous amplitude and discrete-time process. The input matrix $X_t \in \Re^{N\times M}$, at time instance t, $X_t = (x_{t1},...,x_{tM})$, $x_{tm} \in \Re^N$, $m=1,...,M$, contains $N\times M$ successive samples from this process and is the input to the vector/matrix encoder and Update Decision/Codebook Modifier. The functionality of each block is described in the sequel.

Vector encoder: The vector encoder consists of a vector quantizer with codebook $V \in \Re^{N\times K}$, $V=(v_1,...,v_K)$, $v_g \in \Re^N$, $g=1,2,...,K$ and an index assignment scheme. The vector quantizer outputs, for each column vector x_{tm} of X_t, the codevector $\tilde{x}_{tm} \in V$ according to the following distortion-based only quantization rule:

$$\tilde{x}_{tm} = \arg\min_{v_g \in V} d(x_{tm}, v_g) \qquad (1)$$

where $d(x_m,v_g)=\|x_m-v_g\|^2$ is the squared Euclidean distance between vectors $x_m, v_g \in \Re^N$. Therefore, the total output of the vector quantizer for input matrix X_t is matrix $\tilde{X}_t = (\tilde{x}_{t1},...,\tilde{x}_{tM})$. The index assignment scheme assigns to its input $\tilde{x}_{tm}$ the binary representation $b(e_m)$ of the index e_m of $\tilde{x}_{tm}$ in codebook V. As a result, the total produced binary sequence at time instance t is $e=b(e_1)b(e_2)...b(e_M)$. The index assignment mapping rule is given by:

$$b(e_m) = (e_m-1)_2 \qquad (2)$$

Vector decoder: The decoder receives the binary sequence $e'= b(e_1')b(e_2')... b(e_M')$. It maps each received $b(e_m')$ to the e_m'-th element of V, $v_{e'_m} \in V$, based on the mapping rule (2). Thus decoder output at time t is matrix $\hat{X}_t = (v_{e'_1},...,v_{e'_M})$.

Matrix encoder: The codebook of the matrix quantizer is $C_t \in \Re^{N\times M\times L}$, where L is the number of codematrices of dimensions $N\times M$ in C_t. At time instance t, it quantizes X_t into codematrix $c_l \in C_t$ according to the following rate-distortion quantization rule:

$$c_l = \arg\min_{c_i \in C_{t-1}} [D(X_t, c_i)+\lambda\cdot(-\log_2 p_{t-1}(i))],\ 1\le i \le L \qquad (3)$$

The distortion measure $D(X,Y)$ between any two matrices $X=(x_1,...,x_M)$ and $Y=(y_1,...,y_M)$, $X,Y \in \Re^{N\times M}$, used also in the generalized Linde-Buzo-Gray (LBG) algorithm [1], is defined as:

$$D(X,Y) = \frac{1}{M}\sum_{m=1}^{M} d(x_m, y_m) \qquad (4)$$

where $d(x_m,y_m)=\|x_m-y_m\|^2$ is the squared Euclidean distance between vectors $x_m,y_m \in \Re^N$. The weighing factor λ decides which of the components (rate or distortion), is important to the designer and $p_{t-1}(i)$ is the probability of occurrence of codematrix c_i at the output of the matrix quantizer at time instance $t-1$. These probabilities are unequal and most importantly time-variant; it is thus necessary to include the rate term in the quantization rule in (3). The index assignment scheme assigns to its input, $c_l \in C_t$, the binary representation $b(l)$ of its index l, $1\le l \le L$:

$$b(l) = (l-1)_2 \qquad (5)$$

Matrix decoder: The codebook of the matrix decoder is $R_t \in \Re^{N\times M\times L}$, where L is the number of reconstruction matrices of dimensions $N\times M$ in R_t. The matrix decoder follows the mapping rule of (5) and operates in exactly the reverse way as the index assignment scheme, mapping the received binary sequence $b(l')$ to the reproduction matrix $r_{l'}$ of R_t.Therefore the decoder output at time instance t is matrix $\hat{X}_t = r_{l'}$.

Update Decision/Codebook Modifier (transmitter side): This is the key element that decides if a codebook update is necessary, according to a rate-decision criterion described in the sequel. If a codebook update is decided at time t, then the secondary transmission line (vector encoder/decoder pair) is activated, and the matrix quantizer codebook C_{t-1} is updated. Otherwise, the main transmission line (matrix encoder/decoder pair) is activated, and the matrix quantizer codebook C_{t-1} is only restructured as described in the following algorithm. In both cases, the block feeds the matrix quantizer at time instance t with its new codebook C_t.

Codebook Modifier (receiver side): The codebook modifier at the receiver side, alters the codebook of the matrix decoder according to the decision met at the transmitter side. If a codebook update is decided there at time instance t, then codebook R_{t-1} is accordingly updated, otherwise it is simply reconstructed, as described in the algorithm. In both cases, the block feeds the matrix decoder at time instance t with its new codebook R_t. It should be noted, that a specific binary codeword, or flag, is normally required to alert the receiver, each time a codebook update has occurred at the transmitter side. Since this flag is sent only when a codebook update occurs, its transmission does not cumber significantly the overall transmission rate.

The critical parameters of the move-to-front GTR algorithm for MQ are the already defined rate-distortion parameter λ and the windowing parameter ω, defined as the number of time instances (or codebook update decisions) prior to the current time instance t, considered in the calculation of the codematrices probabilities $p_t(i)$. The algorithm consists of the following steps:

<u>*Step 1:*</u> The initial time instance $t=1$ and initial codebooks C_0 and R_0 are set.

<u>*Step 1a:*</u> The input matrix X_t is quantized to matrix c_l of the codebook C_{t-1} according to the quantization rule described in (3), where the time subscript states that the codebook was formed by the codebook modifier during the previous time instance $t-1$. This metric minimisation includes the distortion measure $D(X_t,c_l)$ and the probability of occurrence $p_{t-1}(i)$ of the codematrix c_i in the codebook C_{t-1}.

<u>*Step 1b:*</u> The decision criterion ΔJ is calculated, which incorporates the distortion reduction $\Delta D=(X_t, \tilde{X}_t)-D(X_t,c_l)$ and the rate increase $\Delta R=M\cdot log_2K$ bits per input matrix resulting from a codebook update. Thus the advantages and disadvantages of both decisions are merged into a criterion in order to conclude which is the best option:

$$\Delta J = D(X_t, \tilde{X}_t) - D(X_t,c_l)+\lambda\cdot M\cdot log_2K \tag{6}$$

If $\Delta J<0$ algorithm proceeds to Step 1c, otherwise Step 1d.

<u>*Step 1c (Codebook Update):*</u> The input matrix X_t replaces the last codematrix c_L of the codebook C_{t-1} and all elements of C_{t-1} are shifted one position to the right, such that X_t becomes the first element in the row, thus forming C_t. Binary sequence e is transmitted through the channel and $\hat{X}_t=(v_{e'_1},...,v_{e'_M})$ is the output. Matrix $\hat{X}_t$ replaces the last codematrix r_L of the codebook R_{t-1}, while all elements of R_{t-1} are shifted one place to the right such that $\hat{X}_t$ becomes the first element in the row, thus forming R_t. The codebook modifier at the transmitter side 'splits' the probability of codematrix c_l between c_l and X_t (or otherwise c_L, since X_t substitutes c_L). Therefore the number of times each codematrix is selected at the output of the matrix quantizer at time instance t is:

$$n_t(c_i)=\begin{cases}\omega\cdot p_{t-1}(c_i), \text{if } c_i\neq c_l, c_L\\ \omega\cdot p_{t-1}(c_i)/2, \text{if } c_i=c_l, c_L\end{cases} \tag{7}$$

The new codematrix probabilities $p_t(i)$, $1\leq i\leq L$ are given by:

$$p_t(c_i)=\frac{n_t(c_i)}{\sum_{j=1}^{L} n_t(c_j)} \tag{8}$$

After completion, the algorithm proceeds to Step 1e.

<u>*Step 1d (No Codebook Update):*</u> Codematrix c_l is moved to the front of C_{t-1}, index l is transmitted through the noisy channel, and reproduction matrix $r_{l'}$ is moved to the front of R_{t-1}. The new codematrix probabilities $p_t(i)$, $1\leq i\leq L$ at time instance t are then given by:

$$p_t(c_i)=\begin{cases}\omega\cdot p_{t-1}(c_i)/(\omega+1), \text{if } c_i\neq c_l\\ (\omega\cdot p_{t-1}(c_i)+1)/(\omega+1), \text{if } c_i=c_l\end{cases} \tag{9}$$

After completion, the algorithm proceeds to Step 1e.

<u>*Step 1e:*</u> It is set $C_t=C_{t-1}$, $R_t=R_{t-1}$. If $t<I$ (I is the length of training sequence) then $t=t+1$ and the algorithm returns to Step 1a.

The main difference of the concept of AMQ from the concept of the AVQ algorithm [18] is that the indices produced by the matrix encoder have a fixed length because the transmission channel is assumed to be fixed-rate.

III. CSACOMQ ALGORITHM

The CSACOMQ algorithm combines the AMQ algorithm presented in section 2 with the COMQ algorithm [16]. The following steps provide its definition:

<u>*Step 1:*</u> The stopping threshold of the algorithm ε, and the initial codebooks $C^{(0)}$ and $R^{(0)}$ are selected. An initial value $D^{(0)}$ for the overall mean square error (MSE) between the source-input matrix and the sink-output matrix is set. The initial set $\{p_0(i),\ 1\leq i\leq L\}$ of probabilities of occurrence of codematrices $c_i\in C^{(0)}$, the rate-distortion parameter λ, and the windowing parameter ω are also selected. The parameter λ is selected according to the weighing of rate, or distortion in the algorithm, and ω is selected according to the adaptation requirement of the matrix encoder/decoder. The iteration parameter k is set $k=1$.

<u>*Step 2:*</u> Initial codebooks of the move-to-front GTR algorithm are set $C_0=C^{(k-1)}$, and $R_0=R^{(k-1)}$.

<u>*Step 3:*</u> The steps of the move-to-front GTR algorithm of the previous section are executed and generate codebooks C_t and R_t.

<u>*Step 4:*</u> The iteration parameter n is set $n=1$ and initial codebooks are set $C^{(0)}= C_t$ and $R^{(0)}= R_t$.

<u>*Step 4a:*</u> From the set $R^{(n-1)}=\{r_1^{(n-1)}, r_2^{(n-1)},..., r_L^{(n-1)}\}$ the optimal partition $\{S_i^{(n)},\ 1\leq i\leq L\}$ is calculated according to the following partition rule:

$$S_i^{(n)}=\{X_t : \sum_{j=1}^{L} p_{j/i}\cdot D(X_t, r_j^{(n-1)}) \leq \sum_{j=1}^{L} p_{j/u}\cdot D(X_t, r_j^{(n-1)}),\ u=1,2,..,L\} \tag{10}$$

where $p_{j/i}$ is the *a-posteriori* probability that index j is received when index i is transmitted. The above partitioning minimises the distortion measure between source input matrix X_t and sink output matrix $\hat{X}_t$.

Step 4b: The optimal set of reconstruction matrices $R^{(n)}$ is calculated according to:

$$r_j^{(n)} = \frac{\sum_{i=1}^{L} p_{j/i} \cdot E(X_t \mid X_t \in S_i^{(n)})}{\sum_{i=1}^{L} p_{j/i} \cdot P_i^{(n)}}, j=1,...,L \tag{11}$$

where $P_i^{(n)}$ is the probability that input matrix X_t is an element of the partition $S_i^{(n)}$.

Step 4c: The overall MSE is computed:

$$D^{(n)} = \sum_{i=1}^{L} \sum_{j=1}^{L} p_{j/i} \cdot E[D(X_t, r_j^{(n)}) \mid X_t \in S_i^{(n)}] \tag{12}$$

If $(D^{(n-1)}-D^{(n)})/D^{(n)}<\varepsilon$ then,

(i) the codebook $R^{(k)}$ is set $R^{(k)}=R^{(n)}$,

(ii) the codebook $C^{(k)}$ is set as the current set of centroids of $S_i^{(n)}$, $c_i^{(k)}=E(X_t|X_t \in S_i^{(n)})$, $i=1,...L$,

(iii) the overall MSE $D^{(k)}$ is set $D^{(k)}=D^{(n)}$,

and the algorithm proceeds to Step 5.

Otherwise the iteration parameter n is set $n=n+1$ and the algorithm returns to Step 4a.

Step 5: If $(D^{(k-1)}-D^{(k)})/D^{(k)} <\varepsilon$, then the algorithm has reached a solution and stops. Otherwise iteration parameter k is set $k=k+1$ and the algorithm returns to the Step 2.

IV. Simulation Results

Simulation results presented in this section document the performance improvement of the CSACOMQ algorithm over the COMQ algorithm, in terms of Signal-to-Noise Ratio (SNR) gain. The non-stationary source applied, was the Wiener process with $\sigma=1$, which generates sequences of $500000 \cdot N \cdot M$ samples for the training and testing of both algorithms. For comparison purposes, both algorithms were trained with the same training samples sequence, and were then tested using various sample sequences. The stopping threshold ε in the convergence criteria is set to 0.001. The windowing parameter is chosen ω=1000, which provides a sufficient frame of past input matrices. The size of the auxiliary vector encoder/decoder codebook K=512, which provides a sufficient resolution for quantizing the input matrix in case of a Codebook Update. The initial values of the set of probabilities of occurrence $\{p_0(i), 1\leq i\leq L\}$ of the codematrices at the output of the matrix quantizer, were assumed to be equal. Results were produced for rate-distortion parameter values ranging from λ=1800 to λ=4500. In this range, a reasonable trade-off between rate and distortion is observed. If p is the crossover probability of the Binary Symmetric Channel, then transition probability $p_{j/i}$ is:

$$p_{j/i} = p^x (1-p)^{\log_2 L - x} \tag{13}$$

where x is the Hamming distance between the binary representations $b(i)$, $b(j)$ of indices i and j respectively.

The reference COMQ algorithm was simulated as described in [16]. The initial codebooks $R^{(0)}$ and $C^{(0)}$ for both algorithms were generated using the Generalized LBG algorithm [1]. It was assumed that the information about a codebook update is passed to the decoder side without distortion. Any possible matrix encoder/decoder mis-synchronisation due to disruption of this information, is cancelled at the next iteration of the algorithm.

Simulation results for crossover probabilities $p=0.001$, 0.005, 0.01, and 0.05 were generated, and the SNR performance gain of CSACOMQ over COMQ algorithm was calculated ranging from 0.29 to 1.57 dB, as per Table 1. The chosen design parameters were $N=2,3$, $M=4,6$ and $L=256,512,1024,2048$. The SNR is defined as:

$$SNR=10 \cdot \log_{10}[E(X_t^2)/E(D(X_t, \hat{X}_t))]\text{dB} \tag{14}$$

TABLE I. Performance Comparison Between Csacomq And Comq

SNR gain of CSACOMQ over COMQ (dB)	Crossover Probability p of the BSC			
	p=0.001	*p=0.005*	*p=0.01*	*p=0.05*
N=2, M=4, L=256	1.32	1.31	0.83	0.54
N=2, M=4, L=512	1.45	1.41	0.89	0.56
N=2, M=4, L=1024	1.53	1.49	0.92	0.58
N=2, M=4, L=2048	1.57	1.50	1.04	0.59
N=2, M=6, L=256	1.18	1.16	0.72	0.42
N=2, M=6, L=512	1.22	1.20	0.76	0.46
N=2, M=6, L=1024	1.27	1.25	0.79	0.47
N=2, M=6, L=2048	1.30	1.26	0.80	0.50
N=3, M=4, L=256	1.16	1.13	0.68	0.39
N=3, M=4, L=512	1.21	1.19	0.71	0.43
N=3, M=4, L=1024	1.23	1.20	0.76	0.44
N=3, M=4, L=2048	1.27	1.24	0.79	0.47
N=3, M=6, L=256	1.02	0.99	0.59	0.29
N=3, M=6, L=512	1.06	1.02	0.61	0.29
N=3, M=6, L=1024	1.11	1.08	0.63	0.30
N=3, M=6, L=2048	1.14	1.11	0.67	0.31

The mean square value of the input matrix X_t is defined as:

$$E(X_t^2) = \frac{1}{M} \sum_{m=1}^{M} E(x_{tm}^2) \tag{15}$$

where $E(x_{tm}^2) = \|x_m\|^2$ is the squared Euclidean norm of x_m. With $E(D(X_t, \hat{X}_t))$, the mean value of the distortion between source input matrix X_t and sink output matrix $\hat{X}_t$ is denoted.

The matrix quantization rate (source coding rate) is defined as:

$$R_{MQ} = (\log_2 L)/(N*M) \text{ bits/source symbol} \tag{16}$$

From the results demonstrated at Table 1, it can be concluded that (i) the SNR gain increases as the matrix quantization rate increases (ii) for identical matrix quantization rate, the SNR gain increases as the number of rows (N) of each element matrix of the codebook decreases and (iii) the SNR gain decreases as the channel becomes noisier (or as crossover probability increases). The CSACOMQ and COMQ algorithm convergence time have been measured for a large number of simulation cases. At all cases, the convergence time of the CSACOMQ algorithm exceeds the one of the COMQ algorithm by no more than 12%. On the other hand, CSACOMQ algorithm has exactly the same on-line computational requirements as COMQ algorithm, since the CSACOMQ algorithm designs the vector quantizer/decoder off-line at frequent periodical time intervals.

V. Conclusions And Future Work

In this paper a novel move-to-front GTR algorithm for Matrix Quantization is initially presented. Subsequently, the novel CSACOMQ algorithm is introduced, which designs a matrix quantizer/decoder pair taking into account both the non-stationary nature of the source and the noisy nature of the channel. The CSACOMQ algorithm is compared to the reference COMQ algorithm for various values of the design parameters of the matrix quantizer/decoder pair and various values of the crossover probability of the channel. Simulation results show gains of up to 1.57 dB for source coding rate ranging from 4/9 to 11/8 bits per source symbol. The SNR gain is achieved at the expense of minimal off-line additional computational complexity, while no additional on-line computational complexity is required.

The CSACOMQ algorithm can be utilised by modern audio and speech codec standards [2-6,9,10], image compression techniques [8] and video compression techniques [7] where common ground is the need for low bit-rate transmission, quick and robust adaptation to varying source statistics and optimization to fast changing noisy channels. The concept of CSACOMQ can be generalised to combine other AMQ techniques with the COMQ algorithm, and can also be applied to more specialised channels models, such as the Flat (i.e., non-selective in frequency) Fading Rayleigh Channel, for mobile/wireless systems. These issues are currently under study by the authors.

References

[1] A. Gersho, and R. Gray, Vector Quantization and Signal Compression, Springer, 1992.

[2] J. Makinen, B. Bessette, S. Bruhn, P. Ojala, and R. Salami, “AMR-WB+: A new audio coding standard for 3rd generation mobile audio services”, Proceeding of 2005 IEEE International Conference on Acoustics, Speech and Signal Processing, Vol.2, pp. 1109-1112, 2005.

[3] J.H. Chen, and J. Thyssen, “The Broadvoice Speech Coding Algorithm”, Proceeding of 2007 IEEE International Conference on Acoustics, Speech and Signal Processing, Vol. 4, pp. 537-540, 2007.

[4] J.M. Valin, T.B. Terriberry, C. Montgomery, and G. Maxwell, “A High-Quality Speech and Audio Codec With Less Than 10 ms delay”, IEEE Transactions on Audio, Speech and Language Processing, Vol. 18, No. 1, pp. 58-67, January 2010.

[5] J.M. Valin, G. Maxwell, T.B. Terriberry, and K. Vos, “High-Quality, Low-Delay Music Coding in the Opus Codec”, Paper No. 8942, Proceedings of the 135th Convention of the Audio Engineering Society (AES-13), New York, NY, US, October 2013.

[6] K. Vos, K.V. Sorensen, S.S. Jensen, and J.M. Valin, “Voice Coding with Opus”, Paper No. 8941, Proceedings of the 135th Convention of the Audio Engineering Society (AES-13), New York, NY, US, October 2013.

[7] S. Esakkirajan, T. Veerakumar, and P. Navaneethan, “Adaptive Vector Quantization based Video Compression Scheme”, International Conference on Multimedia, Signal Processing and Communication Technologies, pp. 40-43, 2009.

[8] M. Kumar, R. Kapoor, and T. Goel “Vector Quantization based on Self-Adaptive Particle Swarm Optimization”, International Journal of Nonlinear Sciences, Vol. 9, No.3, pp. 311-319, June 2010.

[9] M. Zhanyu, A. Leijon, and W.B. Kleijn, “Vector quantization of LSF parameters with a mixture of dirichlet distributions”, IEEE Transactions on Audio, Speech and Langugage Processing, Vol. 21, No.9, pp. 1777-1790, September 2013.

[10] M.C. Chu, and D.V. Anderson, “Likelihood codebook reordering vector quantization”, International Conference on Acoustics, Speech and Signal Processing, pp. 5114-5117, 2013.

[11] V. Bozantzis, and Ali, “Combined Source Adaptive and Channel Optimized Vector Quantization Algorithm”, IEE Electronics Letters, Vol.35, No.17, pp. 1455-1456, August 1999.

[12] V. Bozantzis, and Ali, “Combined Source Adaptive and Channel Optimized Vector Quantization (CSACOVQ) for a Flat Fading Rayleigh Channel”, Proceedings of the IEEE Symposium on Communications and Vehicular Technology, pp. 201-208, IMEC, Leuven (BE), October 2000.

[13] V. Bozantzis, and Ali, “Combined Vector Quantization and Index Assignment with Embedded Redundancy for Noisy Channels”, IEE Electronics Letters, Vol. 36, No.20, pp 1711-1713, September 2000.

[14] V. Bozantzis, F. H. Ali, and E. Stipidis, “Joint Source Adaptive Vector Quantization and Index Assignment with Embedded Redundancy”, IET Proceedings on Communications, Vol.1, No.5, pp 1023-1030, October 2007.

[15] C. Xydeas, and C. Papanastasiou, “Efficient Coding of LSP Parameters Using Split Matrix Quantization”, IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 740-743, 1995.

[16] J.L. Perez-Cordoba, A. Rubio, J.M. Lopez-Soler, and V.E. Sanchez, “Channel optimized matrix quantization (COMQ) of LSP parameters over waveform channels”, IEEE International Conference on Acoustics, Speech and Signal Processing, Vol.2, pp. 725-728, 2001.

[17] N. Farvardin, and V. Vaishampayan, “On the performance and complexity of Channel-Optimized Vector Quantizers”, IEEE Transactions on Information Theory, Vol.37, No.1, pp.155-160, February 1991.

[18] J.E. Fowler, “Generalized Threshold Replenishment: An Adaptive Vector Quantization Algorithm for the Coding of Non-stationary Sources”, IEEE Transactions on Image Processing, Vol. 7, Issue 10, pp. 1410-1424, October 1998.