14th European Signal Processing Conference (EUSIPCO 2006), Florence, Italy, September 4-8, 2006, copyright by EURASIP

CONDITIONAL SPLIT LATTICE VECTOR QUANTIZATION FOR SPECTRAL ENCODING OF AUDIO SIGNALS

Adriana Vasilache

Nokia Research Center
Visiokatu 1, 33720, Tampere, Finland
email: adriana.vasilache@nokia.com

ABSTRACT

In this paper we propose a novel quantization method with application to audio coding. Because the lattice truncation based quantizers are finite, not all input points have nearest neighbors within the defined truncations. The proposed conditional split lattice vector quantizer (CSLVQ) allows the possibility of splitting to lower dimensions an input point falling outside the truncation enabling thus the preservation of a low distortion, with only a local payoff in bitrate. Furthermore, the proposed quantization tool is versatile with respect to the dimension of the input data, the same quantization functions being used for different dimensions. The new quantizer has been tested for spectral encoding of real audio samples by encoding each frequency subband of the audio signal using a vector quantizer consisting of a lattice truncated following a generalized Gaussian contour of equiprobability. The results of objective listening tests show similar results to the AAC for high bitrates and clearly better results than the AAC for lower bitrates.

1. INTRODUCTION

The quantization of the modified discrete cosine transformed (MDCT) signal within the advanced audio coder (AAC) is realized using scalar quantization followed by entropy coding of the scale factors and of the scaled spectral coefficients. The entropy coding is performed as differential encoding using eleven possible fixed Huffman trees for the spectral coefficients and one tree for the scale factors. The scale factors for each spectral sub-band are set within two consecutive loops, that make the process rather slow and do not take into account the inter-band correlations affecting the entropy coding. More detailed explanation on the implementation can be found in [1]. Improved results over [1] can be found in [2], where the authors use a trellis search for the constrained optimization of the scales and take into account the inter-band correlation at the entropy coding of the scale factors.

An alternative way to increase the coding efficiency is the use of vector quantization instead of scalar. The clear advantages of the vector quantization over its scalar counterpart have brought it into the attention of researchers, but practical issues like the size of the codebook or the complexity of the encoding algorithms have prevented earlier feasible results. The audio coding using vectorial spectral quantization has been previously successful for lower bitrates [3], but the generalization of the method to higher bitrates was encumbered by increased complexity requirements. Recent work based on the use of 4 dimensional lattice quantization has been presented in [4], but its results have been concentrated on a reduced bitrate domain. We have previously obtained good results for a large range of bitrates in [5] and propose in the following an alternative choice for the quantization method.

The present paper presents a new structure for the quantization of the MDCT spectral coefficients of audio signals within the AAC framework. At each 1024 length frame a lattice vector quantizer is used in each spectral sub-band. The dimension of the quantizer equals the size of the sub-band. Even if the sub-band sizes differ, the same quantization tool is used for all the sub-bands, feature enabled by the use of the conditional split lattice vector quantizer. The bitstream consists of the index of the lattice codevectors and some side information that is entropy encoded. Additionally to the vector quantization, a parameterization of the quantization resolution enables the use of the method for a large domain of bitrates from 128kbits/s down to 16kbits/s.

The present paper introduces first the proposed quantization tool based on lattice truncations, followed by the description of the encoding scheme of the spectral coefficients of audio signals. The last part consists of results on artificial data, followed by listening test results and it is concluded by mentioning the advantages of the proposed method and prospective future research directions.

2. CONDITIONALLY SPLIT LATTICE VECTOR QUANTIZATION

The split quantization is a well-established structured vector quantizer method allowing the reduction of the complexity of the encoding process at the expense of coding performance. The proposed method, named conditional split lattice vector quantizer (CSLVQ) consists in using a split quantizer recursively, and only when demanded by the input data. The main quantizer is a high dimensional lattice. For a given input data, a point from the infinite lattice, closest to the input is chosen and it must be encoded by means of an integer index represented on a number of bits that is sent as side-information. If the chosen lattice point is outside a specified truncation of the lattice, the high dimensional lattice point is split into two lower dimensional lattice points. The use of the split is signaled as a specific character within the bitstream of the entropy encoded side information. The possibility of the split continues recursively until a lowest predefined dimension, where the nearest neighbor of the input data is searched within the corresponding truncated lattice.

To exemplify the proposed method, suppose the input data dimension is n. The lattice Z_n is used for exemplification, but the procedure can be easily extended for use with other lattices. The pre-defined settings of the method are the admissible input space dimensions and the splitting rules for each dimension value. For instance, suppose there are D dimension values $\mathscr{D} = \{d_1, d_2, \ldots, d_D\}$, decreasingly ordered

and the splitting rules for each of them: $d_1 = d_1^{(1)} + d_1^{(2)}$, $d_2 = d_2^{(1)} + d_2^{(2)}, \ldots, d_D = d_D^{(1)} + d_D^{(2)}$ where $d_i^{(1)} \in \mathscr{D}, i = \overline{1,D}$, $d_i^{(2)} \in \mathscr{D}, i = \overline{1,D}$ and at least one of $d_D^{(1)}$ or $d_D^{(2)}$ should equal d_D. The splitting rules should be such that the splitting can be applied recursively down to the smallest dimension value. For each dimension there exist a pre-defined truncated lattice, specified by a given number of leader vectors [6].

The shape of the pre-defined truncation is given by the contour of equiprobability of the input data. For instance, for Gaussian data the truncation norm is the $\|\cdot\|_2$ norm and the shape of the truncation is spherical. The leader vectors [6], or at least their non-zero components, should be stored. Generally, if the truncation norm of the smallest dimension is large enough, the leader vectors for the higher dimensions can be easily inferred from the smallest dimension leader vectors, reducing thus the storage requirements or they can be even generated on the fly [7].

The input n-dimensional data x is first quantized to the nearest neighbor $NN(x)$ in the infinite lattice [8] and then $NN(x)$ is further encoded. If $NN(x)$ belongs to the pre-defined lattice truncation corresponding to the n-dimensional space, an integer index I_n is assigned to $NN(x)$ according to an enumeration technique described in [10].

Information relative to the number of bits needed to encode I_n is entropy encoded. If NN(x) does not belong to the pre-defined truncation then a split operation is performed according to the splitting rule for that dimension and the symbol '-1' is entropy encoded. Since the input dimension is known, as well as the splitting rules, the value of the dimension is easily deduced. The overall recursive encoding function can be summarized by the following pseudo-code:

```
recursive_encode(NN(x), n, x)
{
   if NN(x) is in pre-defined n-dimensional
                    truncation;
      entropy encode the number of bits
             used for the index of NN(x);
      encode NN(x) in index;
   else
      if  n is the smallest dimension;
        look for the NN'(x) in the
             pre-defined truncation;
        entropy encode the number of bits
             used for the index of NN'(x);
        encode NN'(x) in index;
      else
        entropy encode the "split" character;
        recursive_encode(NN1(x), n1, x1);
        recursive_encode(NN2(x), n2, x2);
}
```

where: $n = n1 + n2$ is the split rule for dimension n, NN1(x) and NN2(x) are the first $n1$ components of NN(x) and the last n2 components of NN(x), respectively and $x1$ and $x2$ are the first $n1$ components of x and the last $n2$ components of x, respectively. There is a small number of symbols (integers from -1 up to 22) used to encode the number of bits employed for the codevector indexes, I_n , which makes the entropy coding very fast. The splitting procedure forms a binary tree, which is read as root, left branch, right branch in order to form the bitstream. For instance, if there is no split (zero depth tree) the number of bits for I_n, followed by I_n is encoded, if there is one split (depth 1 tree) the split character (-1) is encoded for the root and then the number of bits for I_{n_1}, followed by I_{n_1} (the right branch) and the number of bits for I_{n_2}, followed by I_{n_2} (left branch) are encoded. If there are supplementary levels of split, the depth of the tree increases and the tree is read following the same rule. An input scaling factor allows controlling the value of the overall bitrate.

3. SPECTRAL QUANTIZATION USING CONDITIONAL SPLIT LATTICE QUANTIZER

3.1 General scenario

The proposed audio coding scheme follows the AAC principle, by MDCT transforming the time domain signal and quantizing the transform coefficients. The transform coefficients are grouped according to the critical bands [9] and encoded such that the quantization error in each band is not larger than an allowed value given by the psychoacoustic model. We replace the scalar quantization of the transform coefficients with lattice vector quantization that incorporates the proposed quantization tool. Contrary to the AAC case, the transform coefficients are no longer power law transformed. A generalized Gaussian distribution with shape factor 0.5 is assumed [5] for the transform coefficients, and the shape of the lattice truncations is chosen in accordance to it. In each sub-band the normalized spectral coefficients are directly divided by a scale factor and the result of the division is input to the CSLVQ having the dimension equal to the size of the sub-band. We denote by sub-band one group of transform coefficients corresponding to one critical band.

For audio files sampled at 44.1kHz the dimension values used within the CSLVQ are: 4, 8, 12, 16, 20, 24, 28, and 32 split as 32=16+16, 28 = 12+16, 24 = 12+12, 20 = 4+16, 16=8+8, 12= 8+4, and 8 = 4+4.

The scale factor of each sub-band is represented by a base value and an exponent, the exponent being encoded. Like in [5], the base value has different values for low and high bitrates, namely 2 for bitrates lower than 48kbits/s and 1.45 for bitrates higher or equal to 48 kbits/s. The information that is encoded for a given sub-band i consists of the exponents of the scale factors $\{s_i\}$, the lattice codevector indexes $\{I_j^{(i)}\}$ and the information related to the number of bits on which the lattice codevector indexes are represented, $\{n_j^{(i)}\}$. The index j enumerates the number of splits per sub-band. The information representing the scale factors and the number of bits is entropy encoded using Shannon-Fano code or arithmetic code.

Information relative to the number of bits needed to encode I_n is encoded using Table 1 as next explained. For a given dimension, if the $\|\cdot\|_{0.5}$ norm of NN(x) is less than K from row 'i' and larger than K from row '$i-1$' then the symbol 'i' will be entropy encoded, using a Shannon-Fano code, to specify the number of bits 'No. bits'. K in Table 1 is actually the squared root of the $\|\cdot\|_{0.5}$ norm, in order to avoid supplementary multiplications.

There are 23 possible values for the exponents $\{s_i\}$, integers from 0 to 22. The number of values for $\{n_j^{(i)}\}$ is 24 (integers from -1 to 22). There is one code for the bit allocation information and one code for the scale information. All sub-band sizes are considered in a single CSLVQ tool.

The bit allocation in sub-bands realized using constrained optimization is controlled by the values of the scaling factor exponents. The choice of the scaling factor exponents is realized through a constrained optimization algorithm which minimizes an overall error measure while the bitrate should be within the available number of bits given by the bit pool mechanism like in AAC.

Two possible implementations of the optimization algorithm are presented in the next section.

3.2 Optimization algorithms for the scale factors

3.2.1 Lagrangian optimization

The constrained optimization is replaced by an unconstrained one with the criterion including both an error measure and a bit rate measure:

$$J = \sum_{i=1}^{N} \left(D_i + \lambda \left(\sum_j B(\{n_j^{(i)}\}) + \sum_j B(\{I_j^{(i)}\}) + B(s_i) \right) \right) \quad (1)$$

where N is the number of sub-bands, D_i is the error ratio signifying the ratio between the sub-band Euclidean distortion and the allowed distortion for the sub-band i, B is a function giving the number of bits used for encoding its argument and λ is the Lagrangian multiplier. The sub-band encoding is done independently and the counters for the entropy coding are updated once per frame, therefore, for a given λ, the scale factor for each sub-band is chosen from the set of possible values, larger than the initial value, such that it minimizes the error ratio per sub-band. The initial value for the scale factor is the highest integer less than $\log_2 AD_i$ minus 3, $\lfloor \log_2 AD_i \rfloor - 3$, where AD_i is the allowed distortion given by the perceptual model for the sub-band i. The multiplier λ is initialized to 0.000001 and progressively increased by 0.0001 until the number of bits per frame is within the allowed domain. The bitstream is formed by the succession of the binary codes for $\{n_j^{(i)}\}$, $\{I_j^{(i)}\}$, and $\{s_i\}$. If, for a given sub-band, there is no split and the number of bits to encode the lattice codevector is zero (corresponding to the all zero vector), then the scale is no longer encoded, because it does not make sense to encode a scale for a null vector.

3.2.2 Search within the distortion-rate space

For each sub-band up to 20 exponent values are selected for evaluation. These exponents comprise the 19 exponent values larger than the initial one, plus the initial one. If there are not 20 exponent values larger than the initial value, then only those available are considered.

For each sub-band and for each considered exponent, a respective pair of bitrate and error ratio can be obtained. This pair is also referred to as rate-distortion point in the rate-distortion space.

For each sub-band the rate-distortion points are sorted such that the bitrate is increasing. Normally, as the bitrate increases, the distortion should decrease. In case this rule is violated, the distortion measure with the higher bitrate is eliminated. This is why not all the sub-bands have the same number of rate-distortion points.

The rate-distortion measures are ordered with increasing value of bitrate along the sub-bands i, $i = \overline{1,N}$, from 1 to $R(i,Ni)$ and consequently decreasing error ratio, $D(i,j)$, $i = \overline{1,N}$, $j = \overline{1,Ni}$. The algorithm is initialized with the rate-distortion measures having a minimum distortion. The initial bitrate is $R = \sum_i^N R(i,Ni)$. For selecting the best rate-distortion measure with index k, the following pseudo code can be applied:

```
For i=1:N       k(i) = Ni
1 If R < Rmax Stop
2 Else
    While (1)
4     For i = 1:n
5       If k(i) > 1
          Grad(i)=(R(i,k(i))-R(i,k(i)-1))/
                (D(i,k(i)-1) - D(i,k(i))));
      End For
8     i_change=arg(max(Grad));
9     R=R-R(i_change, k(i_change))+
                R(i_change,k(i_change)-1)
10    k(i_change)=k(i_change)-1;
11    If R < Rmax Stop, Output k
12  End While
  End
```

The indexes $k(i), i = \overline{1,N}$, enumerate the rate-distortion points, but also to the exponent value that should be chosen for each sub-band i, which is the one that should be used to engender the rate-distortion point.

For high bitrates, e.g. >= 48kbits/s, the algorithm should be modified at line 5 to 'if $k(i) > 2$' such that the sub-band i is not considered at the maximization process if, by reducing its bitrate, all the coefficients are set to zero.

If the total bitrate is too high, it should be decreased, therefore, some of the sub-bands should have a smaller bitrate. If the only rate-distortion measure available for one sub-band is the one with bitrate equal to 1 - which is the smallest possible value for the bitrate of a sub-band, corresponding to all the coefficients in that sub-band being set to zero -, then in that sub-band the bitrate cannot be further decreased. This is the reason for the test 'if $k(i) > 1$'. For each eligible sub-band, the gradient corresponding to the advancement of one pair to the left is calculated, and the one having maximum decrease in bitrate with lowest increase in distortion is selected. Then, the resulting total bitrate is checked, and so on.

4. RESULTS

The structure used as an example in section 3 is used as testing support. The search within the distortion-rate space has been used in the tests for the optimization of the scale factor exponents.

4.1 Method comparison on artificial data

Results and comparison with other methods are presented in Table 2. The quantization SNR values for a generalized Gaussian source with shape factor 0.5 are presented for several methods and the proposed quantization scheme compares favorably especially at lower dimensions. Although the comparison is made against fixed rate methods, the gain of the proposed method is not due uniquely to the variable rate. For higher dimensions, the performance is comparable to other methods, but the complexity of the proposed CSLVQ is smaller both in terms of memory requirements (the possibility of splitting allows for smaller norm lattice truncations) and from the computational point of view (most of the time a single search in the infinite lattice per input vector). Moreover, if the memory requirements allow, higher norm truncations for the higher dimensions help improving the performance for the higher dimensions as well. Also, the proposed method has the advantage of a greater flexibility, allowing the same quantizer to be used for different input dimensions. Additionally, there is a single parameter, the input scaling factor, that must be optimized beforehand. Alternatively, it can be made adaptive and sent over to the decoder.

4.2 Experimental results on real audio data

The proposed method was compared against the quantization procedure from the MPEG4-AAC codec, in a MUlti Stimulus test with Hidden Reference and Anchor (MUSHRA).

D	4		8		12		16		20		24		28		32	
I	K	Nb	K	Nb	K	Nb	K	Nb	K	Nb	K	Nb	K	Nb	K	Nb
0	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
1	1.73	4	1.41	4	1.73	6	1.73	6	1.41	6	1.41	6	1.41	6	1.41	6
2	2.41	6	2.00	5	2.00	5	2.00	5	2.00	7	2.00	7	2.00	7	2.00	7
3	3.15	8	2.41	8	2.73	10	2.41	10	2.41	10	2.73	12	2.41	11	2.41	12
4	4.00	10	3.00	10	3.24	12	3.00	12	3.00	12	3.00	12	3.00	13	3.00	14
5	5.07	12	3.73	12	3.83	14	3.41	13	3.41	14	3.41	15	3.41	16	3.41	16
6	6.24	14	4.41	14	4.41	16	4.00	16	3.83	16	4.00	18	4.00	18	3.83	18
7	7.56	16	5.15	16	5.00	18	4.46	18	4.41	18	4.41	19	4.41	20	4.15	20
8	9.11	18	5.89	18	5.73	20	5.15	20	4.83	20	5.00	22	4.83	22	4.73	22
9	10.90	20	6.81	20	6.41	22	5.73	22	5.41	22	5.46	24	5.41	24	5.15	24
10	13.03	22	7.68	22	7.15	24	6.41	24	5.83	24	6.15	26	5.83	26	5.61	26
11	16.00	24	8.64	24	7.86	26	7.00	26	6.41	26	6.65	28	6.41	28	6.15	28
12			9.65	26	8.65	28	7.66	28	7.07	28	7.18	30	6.83	30	6.56	30
13			10.74	28	9.47	30	8.38	30	7.66	30	7.83	32	7.41	32	7.15	32
14			11.90	30	10.31	32	9.07	32	8.29	32	8.39	34	7.88	34	7.50	34
15			13.16	32	11.21	34	9.80	34	8.97	34	8.98	36	8.48	36	8.15	36
16			14.00	34	12.14	36	10.56	36	9.60	36	9.56	38	9.10	38	8.66	38
17					13.20	38	11.30	38	10.29	38	10.24	40	9.66	40	9.24	40
18							12.12	40	10.98	40	10.88	42	10.24	42	9.83	42
19							13.20	42	11.71	42	11.56	44	10.88	44	10.38	44
20									12.00	43	12.00	46	11.51	46	10.98	46
21													12.00	48	11.56	48
22															12.00	50

Table 1: The number of bits used to encode the lattice codevectors for dimensions 4, 8, 12, 16, 20, 24, 28, and 32 presented as a function of the squared root of the $\|\cdot\|_{0.5}$ norm.

n	Rate [bit/s]	SNR M.-A [6]	SNR M. -B [6]	SNR lat. VQ [12]	SNR CSLVQ
8	1	6.63	6.74	6.17	7.58
8	2	12.63	12.90	11.99	13.32
8	3	na	na	16.71	19.43
16	1	7.67	7.75	na	7.70
24	1	8.09	8.13	na	7.79

Table 2: Method comparison for Generalized Gaussian data with shape factor 0.5.

Name	Description	Name	Description
es01	Vocal (S. Vega)	si01	Harpsichord
es02	German male speech	si02	Castanets
es03	English female speech	si03	Pitch pipe
sc01	Trumpet solo and orch.	sm01	Bagpipes
sc02	Classical orch. music	sm02	Glockenspiel
sc03	Contemp. pop music	sm03	Plucked strings

Table 3: Listening test samples.

A particularity of the employed AAC codec framework was the 11kHz bandwidth considered for quantization for all the bitrates. Three experiments were designed: for high bitrates (128kbits/s, 96kbits/s, 64kbits/s), for moderate bitrates (64kbits/s, 48kbits/s, 32kbits/s), for low bitrates (32kbits/s, 24kbits/s, 16kbits/s). There was also a training experiment for the listeners, covering all the audio quality range. The files used in the tests are listed in Table 3. The files es01 and sm01 were used only in the training experiment and the remaining files were used in each of the three testing experiments. The mono files are sampled at 44.1kHz. The number of expert listeners was 10 (high bitrates), 11 (moderate bitrates) and 9 (low bitrates). The scale factor of CSLVQ is hidden in the sub-band scaling factors and does not need special setting.

Table 4 presents method comparison results issued through the Student T-test from the listening tests. "Lat" stands for the proposed method using CSLVQ, AAC for the AAC coder and the numbers attached to them represent the bitrate in kbits/s. The grade '1' stands for 'first method is statistically better', '0' stands for statistically equal performance and '-1' for 'first method is statistically worse'.

For high bitrates (128, 96, 64 kbits/s) the proposed method gives similar audio quality to the AAC. For bitrates of lower or equal to 48kbits/s the proposed method achieves better performance compared to the quantization method from the original AAC codec. Furthermore, for low bitrates, there is an important quality improvement, equivalent to bitrate savings of 25% at 32kbits/s and of 33% at 24 kbits/s. The improvements are similar to those presented in [4], but their results concentrated only on the 24kbits/s case. Figure 1 illustrates the average grades for the considered methods in the listening tests. "Cmpnd" stands for the method of [5] using companding followed by lattice quantization with rectangular truncation in sub-bands. The results of CSLVQ and of the companding approach are similar. However, larger truncations sizes allow for better performance of the lattice truncation approach relative to the companding approach [13] advantage which pays off on the complexity of indexing. The conditions 'lp7000' and 'lp3500' are the hidden anchors of MUSHRA test, corresponding to low-pass versions of the original signal to 7kHz and 3.5kHz, respectively.

Meth. 1	Meth. 2	Gr.	Meth. 1	Meth. 2	Gr.
Lat_128	AAC_128	0	Lat_24	AAC_32	0
Lat_96	AAC_96	0	Lat_24	AAC_24	+1
Lat_64	AAC_64	0	Lat_24	AAC_16	+1
Lat_48	AAC_48	+1	Lat_16	AAC_32	-1
Lat_32	AAC_32	+1	Lat_16	AAC_24	0
Lat_32	AAC_24	+1	Lat_16	AAC_16	+1
Lat_32	AAC_16	+1			

Table 4: Listening test results based on Student T-distribution with a confidence level of 95%. Grade values signify: '+1' first method is statistically better, '0' the two methods have statistically similar performance.

5. CONCLUSION

In this paper we have proposed a novel flexible quantization tool. Its flexibility is rendered by the ability to be used for different input data dimensions and to avoid overload distortion. Unlike the classical lattice quantization where the points outside the available lattice truncation are quantized with a larger distortion, the proposed method increases locally the bitrate while keeping the distortion obtained from the search in the infinite lattice. The use of the proposed quantization tool within the AAC encoder proved successful for the overall bitrates from 128kbits/s down to 16kbits/s, the improvement over the original AAC being especially visible for lower bitrates. The result is thus a single high quality coder for the entire bitrate domain.

Acknowledgments

The author would like to thank Henri Toukomaa for conducting the listening tests.

REFERENCES

[1] P. ISO/IEC JTC1/SC29/WG11, "Information technology -Coding of audio-visual objects- Part 3: Audio," in *ISO/IEC 14496-3:2001(E).*

[2] A. Aggarwal, S. L. Regunathan, and K. Rose "Near-optimal selection of encoding parameters for audio coding," in *Proc. ICASSP 2001*, Salt Lake City, May 2001, vol. 5, pp. 3269–3272.

[3] N. Iwakami, T. Moriya, and S. Miki, "High quality audio-coding at less than 64 kbit/s by using transform domain weighted interleave vector quantization (TWINVQ)," in *Proc. of ICASSP 1995*, May, 9-12 1995, vol. 5, pp. 3095–3098.

[4] N. Meine and B. Edler, "Improved quantization and lossless coding for subband audio coding," in *the 118th Convention of the Audio Engineering Society, Convention paper 6468*. Barcelona, Spain, May, 28-31 2005.

[5] A. Vasilache, and H. Toukomaa "Vectorial spectral quantization for audio coding," accepted to *ICASSP 2006*, Toulouse, France, May 14-17, 2006.

[6] A. Vasilache, B. Dumitrescu and I. Tăbuş, "Multiple-scale leader-lattice VQ with application to LSF quantization," *Signal Processing*, vol. 82, no. 4, pp. 47–70, 2002.

[7] P. Rault and C. Guillemot, "Indexing algorithms for Z_n, A_n, D_n, and D_n^{++} lattice vector quantizers," *IEEE Trans. on Multimedia*, vol. 3, no. 4, pp. 395–404, Dec. 2001.

[8] J. H. Conway, and N. J. A. Sloane, *Sphere packings, lattices and groups*. Address: Springer, third edition 1999.

[9] M. Bosi, and R. E. Goldberg, *Introduction to digital audio coding and standards*. Address: Kluwer Academic Publishers, second edition 2003.

[10] A. Vasilache, and I. Tăbuş, "Indexing and entropy coding of lattice codevectors," in *Proc. of ICASSP 2001*, vol. 4, pp. 2605–2608, 2001.

[11] J. Pan, "Extension of two-stage vector quantization-lattice vector quantization" *IEEE Trans. on Communications*, vol. 45, no. 12, pp. 1538–1547, Dec. 1997.

[12] F. Chen, and Z. Gao, and J. Villasenor, "Lattice vector quantization of generalized Gaussian sources," in *IEEE Trans. on Information Theory*, vol. 43, no. 1, pp. 92–103, January 1997.

[13] A. Vasilache, M. Vasilache, and I. Tăbuş "Predictive multiple-scale lattice VQ for LSF quantization ," in *Proc. of ICASSP 1999*, Phoenix, Arizona, USA, March 15-19, 1999.

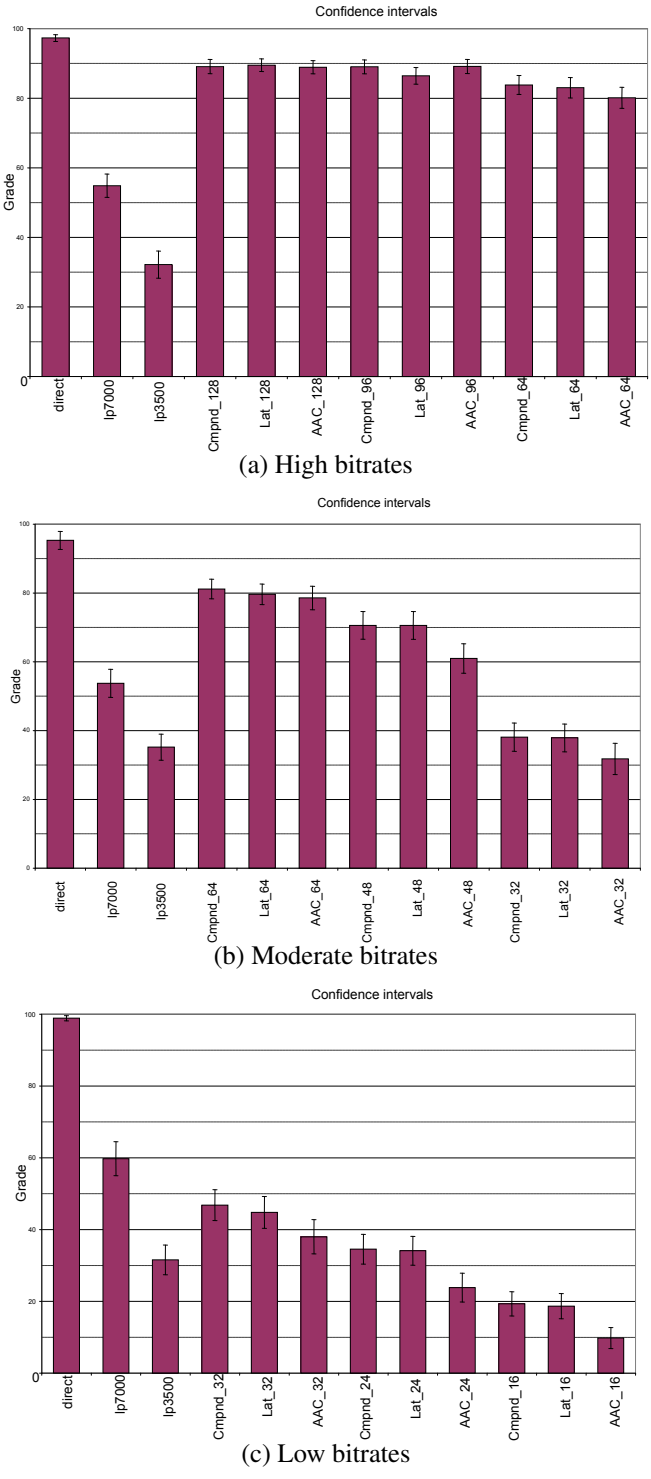

Figure 1: Average of grades from listening tests.