A FAST ROBUST STOCHASTIC ALGORITHM FOR VECTOR QUANTIZER DESIGN FOR NONSTATIONARY CHANNELS

B. Kövesi* #, S. Saoudi*, JM. Boucher* and Z. Reguly#

* ENSTBr, Dept. SC., Technopôle de Brest Iroise, BP 832, 29285 Brest Cedex, France
\# Technical University of Budapest, Dept. MMT., Mûegyetem rkp. 9, 1521 Budapest, Hungary

ABSTRACT

In this paper we present the development of the RGSKAε, a new algorithm for designing vector quantizers. The main features of this algorithm are the following:
- Due to its stochastic nature it avoids being trapped in poor local minima;
- Initial codebook is not needed; the codevectors move away from the gravity centre of the training vectors towards their final position;
- Source coding and channel coding are jointly optimized to obtain a codebook robust against different levels of the transmission noise;
- The resulted codebook always performs as well or even better than existing codebooks designed for noisy or noiseless channels;
- The computational complexity is only slightly higher than that of the most widely used K-means algorithm;
- The Bootstrap sampling technique can be successfully applied in case of a large training set;
- The method is suitable for parallel implementation.

1. INTRODUCTION

As a means of data compression, vector quantization has been widely used in various speech and image coding problems in the past decade. In practical situations, as signals are transmitted through noisy channels, the sent binary index may change during the transmission. Considering that binary indeces are generally randomly associated with the codewords, vector quantization is very sensitive to the transmission noise. In this paper we propose a new algorithm for designing vector quantizers that solves most of the questions met in this area. The objective is to design a vector quantizer that introduces as little distortion as possible under various transmission conditions.
The algorithm results in a codebook where the order of the codewords is chosen to make the codebook intrinsically robust against channel error probability changes. The mean distortion caused by the transmission noise is decreased significantly. At the same time the codebook performs well even in the absence of channel noise. That is why we recommend this method for nonstationary channels. Choosing proper codewords and finding the right order are generally problems for nonconvex optimization. The proposed algorithm solves them in a much more simplified way than the similar existing algorithms.

2. DESIGN OF VECTOR QUANTIZERS

Designing an M level vector quantizer means a partition P of the vector space into M regions (S_i) and the choice of the representative codewords (c_i) for each partition. These M codewords form the codebook C. An optimal quantizer fulfils at the same time two necessary conditions [1]:

a1) The partition P is optimal for the codebook C if (1.):

$$S_i = \left\{ x \in R^k / d(x,c_i) \le d(x,c_j); \forall j \ne i \right\}; \quad i = 1,\dots,M$$

b1) Respectively the codebook C is optimal for the partition P if (2.):

$$\int_{S_i} d(x,c_i)p(x)dx = \inf_{u \in R^k} \int_{S_i} d(x,u)p(x)dx; \quad i = 1,\dots,M$$

where p(x) is the probability density function of the input signal. If the Euclidean distance is chosen, we call these conditions the nearest neighbour and the centroid conditions, and *b1)* simplifies as follows (3.):

$$c_i = \frac{\int_{S_i} x \cdot p(x)dx}{\int_{S_i} p(x)dx}$$

In this paper we assume that d is the Euclidean distance.

2.1. The K-means Algorithm

The KMA (K-means algorithm) is a deterministic iterative descent algorithm that updates the codebook, according by *a1)* and *b1)* [2]. The data source is defined by a finite training set and all training vectors are taken into account at each iteration. The distortion function decreases monotonically, it is however often trapped in a poor local minimum far from the global one. The result is determined by the initial codebook and it is impossible to choose an initial codebook that results in the global minimum. In spite of its drawbacks, the KMA is the most widely used technique because of its simplicity and quick convergence.

2.2. Simulated Annealing Algorithm

In case of nonconvex optimization the simulated annealing algorithm (SA) [3] may be used to reach the global optimum independently of the initial conditions. Unfortunately this method is very time consuming.

2.3. The Stochastic K-means Algorithm (SKA)

The first proposed algorithm is the combination of the KMA and the SA retaining the advantages of both. Its main difference from the KMA is how the training set is partitioned. For a training vector we do not assign the region whose former representative codeword is the nearest neighbour of the training vector, but we make a random decision to choose a region. The probability of assigning the jth region to the ith training vector x_i is a function of the inverse temperature β and the distance between x_i and the jth region's actual representative codeword c_{pj} (4.):

0-7803-2431-5/95 $4.00 © 1995 IEEE

$$P(i,j,\beta)=\frac{e^{-d(x_i,c_{pj})\beta}}{\sum_{n=1}^{M} e^{-d(x_i,c_{pn})\beta}}$$

Of course the nearest neighbour has the greatest probability of being chosen. The inverse temperature increases at each iteration step (system cools down). When the system is warm, (β is small), the probability of choosing a region not corresponding to the nearest neighbour is greater. As the system cools down the algorithm becomes more and more deterministic. It was found that the most suitable solution is to increase β as a linear function of time. In the *x*th iteration (5.): $\beta = x/a + b$. The algorithm is quite insensitive to the choice of the parameters *a* and *b*. For example, during the simulations *a*=1, *b*=0 gave almost the same good result as *a*=16, *b*=16.

2.4 Comparison of the KMA and SKA

Due to its stochastic nature, the SKA cannot be trapped in poor local minima, it cannot reach the global optimum either. Anyway, the result is very close to the global optimum, and the SKA consistently gives better codebooks than the KMA. The SKA is independent from the choice of the initial codebook. In an initial codebook consisting of a set of nullvectors all initial probabilities are equal. After the first iteration, all codevectors, being means of randomly selected training vectors, are closed to the gravity centre of the training set, and than they are progressively moved towards their final position. This ensures a better-balanced codebook.

In KMA the distortion decreases monotonically, and the iterations stop when the distortion cannot be improved any more, thus a local minimum is found. As the distortion function of the SKA does not decrease monotonically, the exit condition is reached when the algorithm can not diminish the distortion during some (e.g.20) iterations. The complexity of the SKA is slightly greater than that of the KMA due to the computation of the probabilities for each distance. The convergence speed depends on the temperature schedule during iterations. Generally the SKA needs app. 50% more computation time than the KMA.

3. VECTOR QUANTIZERS FOR NOISY CHANNELS

Under real conditions, due to the channel noise, the transmitted binary indeces may change at the receiver site. For the simulation a binary symmetric channel with crossover probability ε was assumed The probability of receiving the index *j* when index *i* was sent is (6.):

$$P_I(j/i) = (1-\varepsilon)^{n-d_h(i,j)} \cdot \varepsilon^{d_h(i,j)}$$

where *n* is the length of the binary indeces, $d_h(i,j)$ is the Hamming distance of the indeces *i* and *j*.

In real applications one bit fault per index is the most likely error. In other words, in case of transmission error the Hamming distance of the transmitted and received binary index is usually equals to 1. Considering that in a traditional vector quantizer the codewords and the binary indeces are assigned randomly, one bit fault will cause on an average exactly the same distortion as any other more serious fault. Therefore vector quantization shows considerable sensitivity to the change of binary indeces. There are two main ways to increase the robustness against the channel noise. The first is to use errorcorrection codes. Unfortunately, this increases the bit rate. The second, which is chosen in the present paper, is to construct an ordered codebook. In such a codebook, indeces with small Hamming distance belong to codewords on the average close to each other in space. This is the reason why the distortion caused by the most likely faults decreases.

3.1. Generalized K-means Algorithm

In [4] Farvardin proposed an extension of the KMA for noisy binary symmetric channels (generalized K-means algorithm, GKMA). While the KMA tries to minimize the distortion between the training vectors and their representative codewords, the GKMA does this between the training vectors and their codeword restored by the receiver. In this case the two necessary conditions will be the following [4]:

a2) The partition *P* is optimal for the codebook *C* if (7.):

$$S_i = \left\{ x \in R^k / \sum_{j=1}^{M} P_I(j/i) \cdot d(x,c_j) \le \sum_{j=1}^{M} P_I(j/k) \cdot d(x,c_j); \forall k \ne i \right\}$$

i=1,...,M

b2) The codebook *C* is optimal for the partition *P* (supposing the Euclidean distance) if (8.):

$$c_i = \frac{\sum_{j=1}^{M} P(i/j) \int_{S_j} x \cdot p(x)dx}{\sum_{j=1}^{M} P(i/j) \int_{S_j} p(x)dx}; \quad i = 1,\ldots,M$$

We can see that the computation complexity becomes considerably greater than that of the KMA. If the length of the training set is *k* times longer than the size of the codebook, *a2)* increases the computation time *k*/2 times more than *b2)*. The value of *k* is typically equal to 100.

3.2. Generalized Stochastic K-means Algorithm

We can easily generalize the method to the SKA and obtain the generalized stochastic K-means algorithm (GSKA). As the codewords move away from the gravity centre, their correct distribution is not influenced by the distribution of the initial codebook. Due to that, and to other advantages mentioned above for the SKA, we obtain a better performance than with the GKMA. We could see previously that the increase of the computation complexity derives mainly from *a2)*, while it seems less important to generalize this condition. To reduce complexity, we define a random decision of the membership of the training vectors in the same way as in the SKA. The statistical properties of the channel are taken into account only for the construction of the new codevectors. Like in the GKMA or GSKA, each new codevector is the weighted centroid of all training vectors. If a training vector is associated with the *i*th region according to the random decision, its weight, in the calculation of the *j*th region's codevector, will be the probability of receiving the index *j* when the index *i* was sent. In the *p*th iteration (9.):

$$c_{p+1,i} = \frac{\sum_{j=1}^{M} P_I(i/j) \sum_{x \in S_{pj}} x}{\sum_{j=1}^{M} P_I(i/j) \cdot \|S_{pj}\|}$$

where $\|S_{pj}\|$ is the cardinality of S_{pj}. As we update the codebook only once per iteration the computation time remains almost the same as the SKA. Simulations show that the reduced complexity version (RGSKA) performs as well or even better than the original one (GSKA).

3.3. Evaluation of the GKMA, GSKA and RGSKA

To use the GKMA, GSKA or RGSKA one should know the statistical properties of the transition channel in advance. The joint design of the codebook and the binary codeword assignment results in considerable performance improvement when applied to a given channel. In case of a channel mismatch however the performance can be even worse than for the KMA. Let us consider the case of a binary symmetric channel with crossover probability ε. If a small value of ε' is given to the algorithm, its influence is not enough to correctly assign binary codewords to the codevectors. If a higher value of ε' is given, the codebook will be better organized but the average distortion in the absence of channel noise will be worse. The reason is that because of the greater influence of the channel error the codewords of the codebook will be more compressed, and their average distance will be smaller. This is why these codebooks cannot be used for an unknown or nonstationary communication channel, which is the case in many practical situations (e.g. mobile communication, underwater telephone).

3.4. Simulated Annealing to reorder a codebook

We have seen above that the codebook is well organized when the average distortion between those codevectors, whose index differs from one another by one bit, is smaller than for any other ordering of the same codewords. An alternative construction of a vector quantizer for noisy channels is to reorder the codebook given by the KMA or the SKA using an algorithm based on simulated annealing (RSA) [5] to minimize distortion. The codebook obtained by the RSA keeps its performance in the noiseless case (as it contains the same codewords only in different order) and generally gives better protection against transmission errors. However the construction of this codebook is computationally expensive.

3.5. The RGSKAε

To attain this performance with the RGSKA a high value of ε was used at the beginning of the algorithm, which exponentially decreased later towards a small value of γ during subsequent iterations.

In the *p*th iteration (10.): $\varepsilon(p) = \varepsilon_0 \cdot \alpha^p + \gamma$

It is important that ε must decrease to reach the value of γ when the algorithm stops. In simulations described in the 6th chapter $\varepsilon_0=0.5$, $\alpha=0.94$, $\gamma=0.001$ were used. γ has no influence on the noiseless distortion, but it helps to keep the order of the codebook better during the final iterations.

During the first half of the iterations a well-organized codebook is produced with a relatively high distortion in noiseless cases. In the second half the distortion diminishes without significantly changing the order of the codevectors as the temperature is already cold enough. The codebook constructed in this way (RGSKAε) performs as well as the codebook obtained by the RSA both in noiseless and noisy cases. The computation time of the algorithm is only about two times longer than the KMA.

4. BOOTSTRAP SAMPLING TECHNIQUE

The computation time of all these algorithms based on a training set is proportional to its size. The number of the training vectors must be at least 10-20 times longer than the size of the codebook, but a longer training set produces better results during applications. If a very long training set is available, it is recommended to use it all to construct a codebook. To reduce computational time we successfully applied the Bootstrap sampling technique with the RGSKAε. In each iteration only a certain percentage of the training set is processed, chosen randomly. After each iteration the used sub-training set is refreshed by randomly redrawing a certain part (10-20%) of it from the entire training set. In this way the information of the whole training set was used and the convergence of the algorithm was assured at the same time.

5. PARALLEL IMPLEMENTATION

The RGSKAε can be easily implemented on parallel hardware if needed. Both the processing of the training set and the calculation of the new codewords can be done in parallel. The computing processes however should be synchronized before and after the calculation of the new codewords.

6. EXPERIMENTAL RESULTS

The experimental results were obtained in the application of vector quantization to 10 Line Spectrum Pairs (LSP) coefficients for the CELP coder. In the following are given some results for the codebooks of 256 codevectors. Fig.1.,2. illustrate the distortion of different vector quantizers as function of the crossover probability ε of a binary symmetric channel. By distortion of a codebook we mean the mean distance between a test vector and its codeword reconstructed by the receiver. All codebooks were obtained by using the same 25600 training vectors. In Fig.1. the testing sequence was identical to the training sequence while in Fig.2. we used 8000 testing vectors separated from the training vectors. The RGSKAε has the smallest distortion for ε<1%, which is the case for real applications. We could not trace the distortion of the reordered codebook of the KMA with RSA as it was identical with that of the codebook obtained by the RGSKAε. However the RGSKAε used 13 times less computation time even for this long training set.

Fig.3. illustrates the point that by using the Bootstrap sampling technique we can also reduce computational time. The dashed curve gives the distortion of the codebook obtained by RGSKAε using 66000 training vectors. During the construction of another codebook we used only the first 13200 training vectors. This is represented by the dotted curve. For the third codebook (solid curve) in each iteration only 20% of the available

66000 training vectors (13200 vectors as well) chosen by Bootstrap sampling technique were used. We refreshed 20% of the 13200 vectors in each iteration. The distortion of this codebook is only slightly higher than the distortion of the codebook obtained using all the training vectors in each iteration. On the other hand, this codebook is much better than one constructed without the Bootstrap technique, though the same number of training vectors was used in a simple iteration and the computational time was identical as well. The same sequence of 8000 testing vectors as in Fig.2. permits the comparison the two figures as well.

Fig.1 Comparison for identical testing set

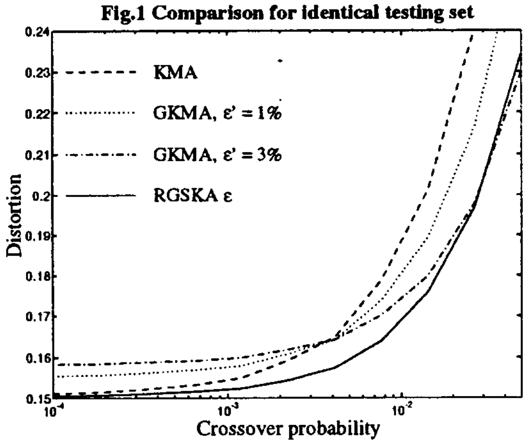


Fig.2 Comparison for different testing set

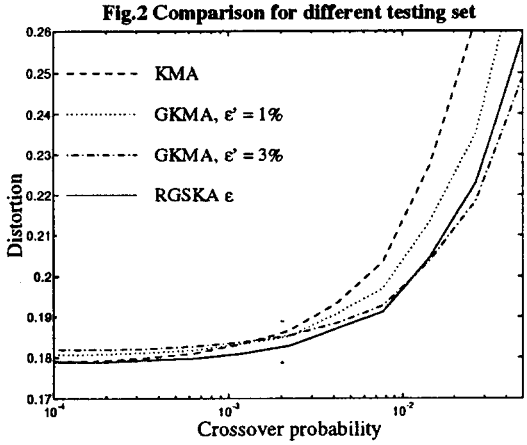


Fig.3 Using Bootstrap sampling technique

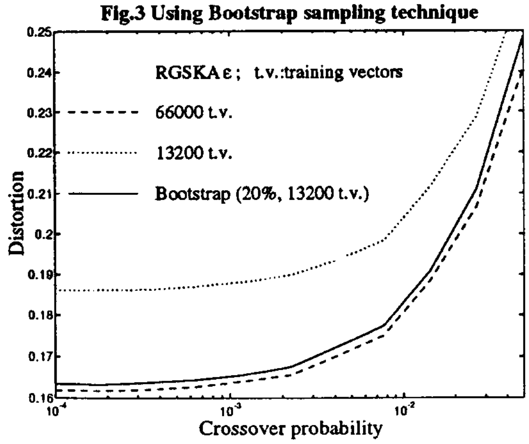
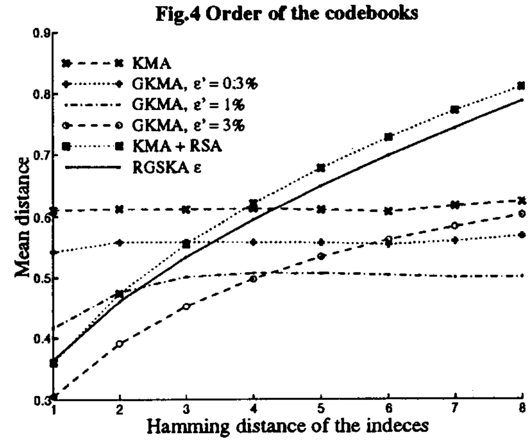


Fig.4. illustrates the mean distance between the codevectors as a function of the Hamming distance of their indeces. For a randomly assigned codebook (KMA) it results a horizontal line. For the codebooks of the GKMA we can observe that even for ε'=0.3% the codebook is not ordered. As the value of ε' given to the algorithm is increased, the codebook becomes better ordered, but at the same time more compressed (smaller mean distance). This is why its performance is worse in the noiseless case.

Fig.4 Order of the codebooks



7. CONCLUSIONS

We have presented a new simple and fast stochastic algorithm for vector quantizer design for nonstationary channels. We can recommend the RGSKAε in all applications as it produces a codebook that gives little average distortion in the absence of channel noise and a high degree of robustness independently of channel error probability changes and initial conditions.

Acknowledgments:
The authors thank Alison Gourves Hayward for her kind assistance with the English of this paper.

8. REFERENCES:

[1] E. Yair, K. Zeger, and A. Gersho, "Competitive Learning and Soft Competition for Vector Quantizer Design", IEEE Trans., vol. SP-40, No. 2, Feb. 1992, pp. 294-309.

[2] Y. Linde, A. Buzo, and R.M. Gray, "An Algorithm for Vector Quantizer Design", IEEE Trans., vol. COM-28, Jan. 1980, pp. 84-95.

[3] S. Geman and D. Geman, "Stochastic Relaxation, Gibbs Distribution and Bayesian Restoration of Images", IEEE Trans., vol. PAMI-11, No. 6, 1984, pp. 721-741.

[4] N. Farvardin, "A Study of Vector Quantization for Noisy Channels", IEEE Trans., vol IT-36, No. 4, July 1990, pp. 799-809.

[5] G. Le Hen, "Quantification vectorielle des coefficients de prédiction linéaire. Application dans le codeur vectoriel a prédiction adaptative. Etude en présence d'erreurs de transmission isolées", These de Doctorat, Université de Rennes I, Juin 1987.