Local Optimization of Index Assignments for Multiple Description Coding

Jean Cardinal* Sourour Elloumi† Martine Labbé‡

ABSTRACT

Index assignment problems are central to many joint source-channel coding methods and generally require a lot of computational power. We study the optimization of index assignment matrices for the multiple description coding problem. Index assignment matrices are used at the quantization step and provide a conceptually simple and fast alternative to multiply descriptive transform methods. We describe two novel local optimization algorithms using a bipartite matching procedure to locally minimize the expected mean squared error. We provide experimental results on various codebooks as well as a comparison with a previously published optimization algorithm.

1 Introduction

Multiple description coders are designed to send pairs of redundant descriptions of a signal to combat packet losses in networks or channel impairments in diversity systems. In this paper, we describe new methods for the design of multiple description codes based on unconstrained vector quantizers. We assume the existence of a single-description quantizer codebook in $\mathbb{R}^k$ and an encoder α mapping the input random variable X on the index i of the best (in some rate-distortion sense) reproduction vector in the codebook. We define a transcoding step in which the codevector index is translated into a pair of lower-range indices, or *descriptions*. This step is designed to minimize the distortion under the loss of one of the two descriptions.

More precisely, given an index i in the codebook $\mathcal{C} = \{y_i\}_{i=1}^N \subset \mathbb{R}^k$, we wish to define an injective *index assignment (IA) function*

$$f : \{1, 2, \ldots, N\} \to \{1, 2, \ldots, n_1\} \times \{1, 2, \ldots, n_2\}, \quad (1)$$

with $N \geq n_1 n_2$, that can be used to identify y_i using two communication channels with rates $\log n_1$ and $\log n_2$, respectively. It is convenient to represent the mapping f by an *assignment matrix* containing the indices and whose rows and columns are indiced by the values $f_1(i)$ and $f_2(i)$, respectively. Since the mapping f is injective, it may happen that some description pairs are not used. In that case, a dummy value can be put in the matrix.

The value

$$r = \frac{1}{k} \log \frac{n_1 n_2}{N} \quad (2)$$

is the redundancy of the assignment. It is the number of additional bits per sample that are used to protect the transmission against description losses. Note that we can use entropy coding to reduce the bitrate requirement on the two channels.

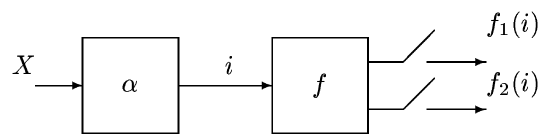


Figure 1: Block diagram of the transmission scheme

We define the objective function to minimize as the mean squared error (MSE) incurred when the codevector y_i is estimated from a single description. The minimum MSE estimation of y_i given, say, $f_1(i)$ is the average of all the codevectors $\{y_j\}$ such that $f_1(j) = f_1(i)$.

Let us denote by $R_l \subseteq \mathcal{C}$ the set of codevectors y_j such that $f_1(j) = l$ and by $C_l \subseteq \mathcal{C}$ the set of codevectors y_j such that $f_2(j) = l$. Then the MSE can be written

$$\text{MSE} = \frac{1}{2kN} \left(\sum_{i=1}^{n_1} \text{var}(R_i) + \sum_{i=1}^{n_2} \text{var}(C_i) \right) \quad (3)$$

where

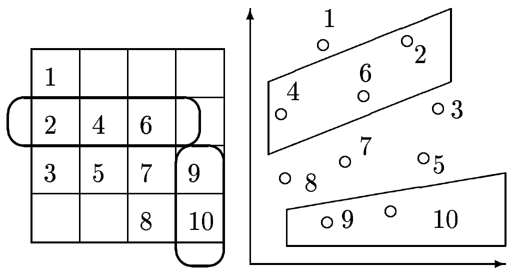
$$\text{var}(S) = \sum_{y \in S} \| y - \frac{1}{|S|} \sum_{x \in S} x \|^2 . \quad (4)$$

An example of IA is shown on Fig. 2(a), together with the corresponding codebook of size 10. The two codevector sets corresponding to $f_1(i) = 2$ and $f_2(i) = 4$ are

* Département d'Informatique, Université Libre de Bruxelles, Bld. du Triomphe CP 212, B-1050 Brussels, Belgium. Email: jcardin@ulb.ac.be

† CEDRIC-CNAM, 292 Rue Saint-Martin, F-75141 Paris cedex 03, France. Email: selloumi@ulb.ac.be

‡ Service d'Optimisation, Institut de Statistique et de Recherche Opérationnelle Université Libre de Bruxelles, Bld. du Triomphe CP 210/01, B-1050 Brussels, Belgium. Email: mlabbe@ulb.ac.be



(a) Index assignment

(b) Codebook

Figure 2: Example of an IA and the subsets R_2 and C_4 in the input space

shown. The IA should minimize the average variance of such sets. Note that we make implicit use of a uniformity assumption over the codevectors. It is rather simple, however, to generalize the above criterion using a probability mass function over the codebook indices. This problem can be shown to be NP-hard by a reduction to the minimum-variance clustering problem.

A great deal of work has already been devoted to the design of good index assignments in the case $k = 1$. The objective function is generally replaced by a simpler one of the form:

$$\min_{f} \max_{i,j: f_1(i)=f_1(j) \vee f_2(i)=f_2(j)} |i - j|. \qquad (5)$$

This is equivalent to the Minimum Graph Bandwidth problem, which is NP-hard in general, for some special graphs defined by the assignment matrix. The Minimum Graph Bandwidth problem consists in ordering the vertices of an arbitrary graph so that the maximal difference between the indices of two adjacent vertices is minimal. In the IA problem, the graph is a twofold cartesian product of cliques or an induced subgraph of it. More precisely, a vertex of this graph is a pair of coordinates in the assignment matrix, and there exists an edge between two vertices if the corresponding coordinates differ on exactly one component. Berger-Wolf [1] studies this problem in the context of coding, and provides useful bounds and a simple systematic algorithm. The optimal such IA function, however, has only been found for $n_1 = n_2$ and $N = n_1^2$. In that case, the maximal index extent is $\binom{\sqrt{N}+1}{2} - 1$. The maximal indices extent with a diagonal arrangement of thickness t was shown to be equal to $t(t-1)/2$. Vaishampayan [8] describes a family of diagonal arrangements that meet this bound. Tian [3] uses a heuristic minimization algorithm and combines it with an efficient image coding scheme.

In Oggier [5], a related problem for $k > 1$ is studied, in which the IA is fixed and the goal is to find good codevectors. Semi-definite programming relaxations are used but no empirical results are given. IA for lattice codebooks have been studied by Vaishampayan, Servetto and Sloane [7]. Koulgi, Regunathan and Rose [4] described a deterministic annealing technique for the joint design of codebooks and IA. To our knowledge, it is the only reference in which the IA problem for $k > 1$ is addressed explicitly.

The next two sections present novel local optimization algorithms for building IA matrices with low MSE.

2 Binary Switching Algorithm

In [2], a simple algorithm that finds a local optimum of the MSE is described. In this method, the neighborhood of an IA matrix M is defined by all the matrices obtained by swapping two elements in M. The algorithm is a simple steepest descent in which the binary switch leading to the maximal objective function decrease is selected at each iteration. We refer to this algorithm as the binary switching algorithm (BSA). The approach is similar to that of Zeger and Gersho for solving the assignation problem in channel-optimized vector quantizers [9] and to the successive refinement IA technique studied by Riskin, Ladner, Wang and Atlas [6].

3 Bipartite Matching Algorithm

We now show that it is possible to design a simple local optimization algorithm in which the neighborhood of a solution is much larger than in the BSA. In this new method two solutions are neighbors of each other if we can switch from one to the other by permutation of the indices in a row or a column. The number of IA matrices that are neighbors of a given matrix M may be as high as $n_1 n_2! + n_2 n_1!$, while in the BSA, the size of the neighborhood is equal to $\binom{N}{2} = N(N-1)/2$.

To obtain a better solution from the current one, we select one row or one column and try to find the permutation of the indices that minimizes the MSE. Let us assume, without loss of generality, that we wish to find the best permutation of the set C_1 of indices that are found in the first column. The MSE in each column is invariant with respect to these permutations, so we are only concerned with the MSE on the rows. Finding the best permutation therefore reduces to a matching problem in a bipartite graph. The two types of nodes are on one hand the indices found in the first column, and on the other hand the rows to which they will be assigned. The solution is a matching between the indices and the lines minimizing the MSE. An illustration of this reduction is given on Fig. 3.

The bipartite matching problem is a classical one for which numerous efficient algorithms exist. It is also straightforward to formulate the problem as a linear program. Let c_{ij} be the MSE cost of assigning the index previously found in the ith row to the jth row and let x_{ij} be an indicator variable such that $x_{ij} = 1$ if this index should actually be assigned to row j. The following lin-

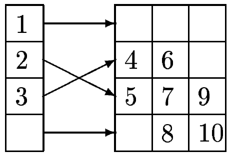


Figure 3: Update of an IA using bipartite matching

ear program is a formulation of the bipartite matching problem:

$$\min \sum_{i=1}^{n_1} \sum_{j=1}^{n_1} c_{ij} x_{ij}, \tag{6}$$

subject to

$$\sum_{j=1}^{n_1} x_{ij} = 1 \ \forall i \in \{1, 2, \ldots, n_1\} \tag{7}$$

$$\sum_{i=1}^{n_1} x_{ij} = 1 \ \forall j \in \{1, 2, \ldots, n_1\} \tag{8}$$

$$x_{ij} \in [0, 1] \ \forall i, j \in \{1, 2, \ldots, n_1\}. \tag{9}$$

It is well known from optimization theory that solutions of this type of problem are integer, that is, $x_{ij} \in \{0, 1\}$.

Let us assume that the index l is found in the assignment matrix M at position $(i, 1)$, that is, l is such that $f_1(l) = i$ and $f_2(l) = 1$. Then the cost c_{ij} of setting $f_1(l) = j$ instead of i is written

$$c_{ij} = \mathrm{var}((R_j \setminus C_1) \cup \{y_l\}). \tag{10}$$

It may happen, however, that no index is assigned to $(i, 1)$. In that case c_{ij} is simply equal to $\mathrm{var}(R_j \setminus C_1)$. In the example of Fig. 3, the cost c_{23} of assigning index 2 found in row 2 to row 3 is equal to $\mathrm{var}(\{y_2, y_5, y_7, y_9\})$. In that case, we have $R_3 = \{y_3, y_5, y_7, y_9\}$, $C_1 = \{y_1, y_2, y_3\}$ and $l = 2$. On the other hand, leaving 2 in the second row yields a cost $c_{22} = \mathrm{var}(\{y_2, y_4, y_6\})$. No index is found in the last row, and the cost of leaving this cell empty is $c_{44} = \mathrm{var}(\{y_8, y_{10}\})$.

When the matching is done, that is when all x_{ij} have their optimal binary value, the MSE of the new IA can be written

$$\mathrm{MSE} = \frac{1}{2kN} \left(\sum_{i=1}^{n_1} \sum_{j=1}^{n_1} c_{ij} x_{ij} + \sum_{i=1}^{n_2} \mathrm{var}(C_i) \right). \tag{11}$$

We propose two algorithms that use bipartite matching as a subroutine for the IA optimization. We will refer to the general idea as BMA (bipartite matching algorithms). In Algorithm 1, each row and each column is optimized in turn in an arbitrary order until no changes are made. The algorithm always finds a local optimum since the objective function is lower-bounded and cannot increase.

Algorithm 1 Systematic BMA

repeat
  for each line (row or column) **do**
    find the best permutation using the bipartite matching algorithm and update M
  end for
until MSE is nondecreasing

In Algorithm 2, we select at each iteration the row or column update that maximizes the MSE decrease. This algorithm takes more time to optimize, since once a row is modified, it has to recompute all the column matchings, and similarly for the columns. On the other hand, it guarantees that at each step the direction of maximal MSE decrease is chosen.

Algorithm 2 Steepest-descent BMA

repeat
  $\Delta_{\max} \leftarrow -\infty$
  for each line (row or column) l **do**
    find the best permutation using the bipartite matching algorithm
    $\Delta \leftarrow$ absolute MSE variation
    if $\Delta > \Delta_{\max}$ **then**
      $\Delta_{\max} \leftarrow \Delta$
      $l_{\max} \leftarrow l$
    end if
  end for
  update row or column $l_{\max}$ in M
until $\Delta_{\max} = 0$

It is not clear which of the two versions should perform best, but both algorithms yield locally optimal IA. It can be used together with a greedy heuristic algorithm that initializes the IA matrix M.

4 Experimental Results

We compared the performance of the three algorithms on codebooks of different sizes designed for a correlated Gaussian distribution with correlation factor 0.9. The vector dimension was set to $k = 4$, and the descriptions have equal rates, that is, $n_1 = n_2$. We plot the results of the three algorithms for codebook sizes $N = 64, 128$ and 256, and for various redundancies r. The assignment matrix size can be deduced from r by the relation $n_1 = n_2 = \sqrt{2^{kr + \log N}}$. In all cases the IA initialization was random. Results are shown on Fig. 4, 5 and 6.

In the three cases, we observe that the BMA yield much lower distortions compared to the BSA. Algorithm 2 is slightly less efficient than Algorithm 1 in most cases. Hence the steepest descent idea does not seem to be the method of choice in that case.

In addition, we remarked that the optimization times were much shorter for the BMA than for the BSA, and the shortest optimization times were obtained with Algorithm 1. This algorithm therefore seems to be the best one so far, both in terms of speed and quality.

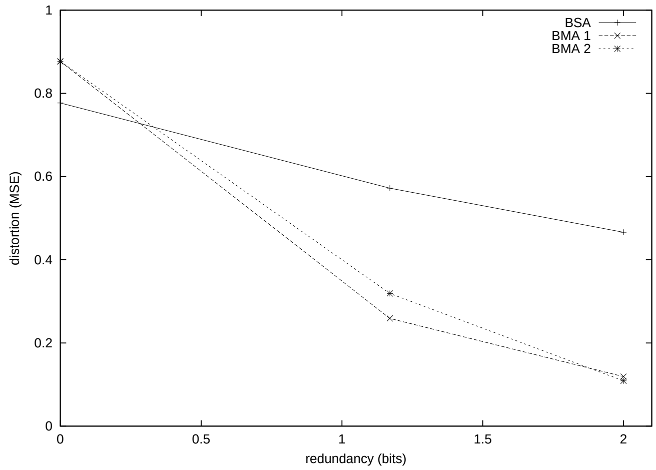


Figure 4: Results for $N = 64$

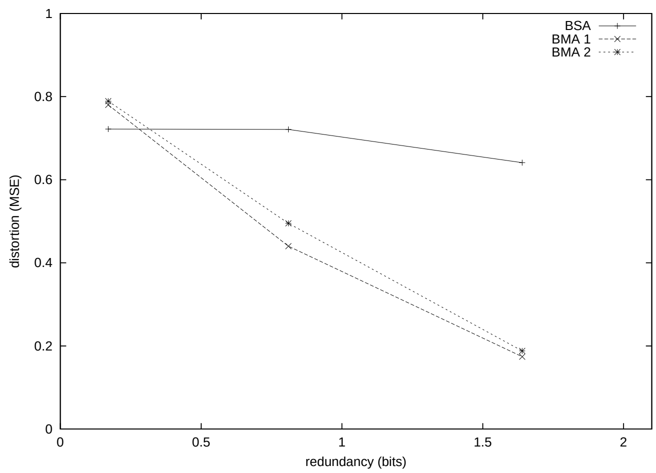


Figure 5: Results for $N = 128$

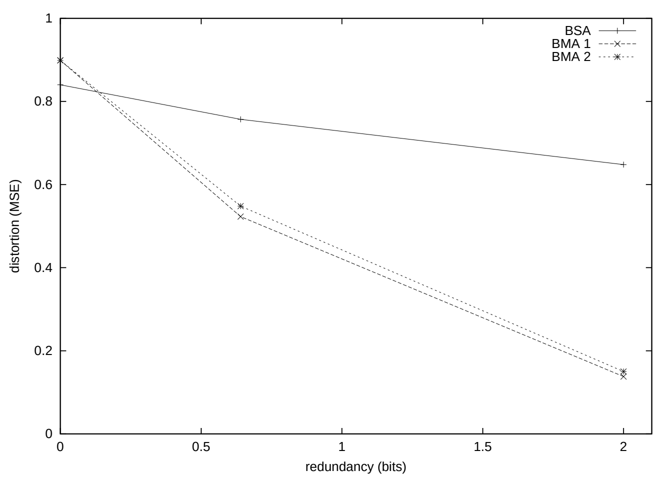


Figure 6: Results for $N = 256$

5 Conclusion

We described two novel local optimization algorithms for the design of index assignments in multiple description vector quantizers. These algorithms clearly improve on a previously published local optimization method both in optimization times and MSE.

Our current work aims at determining exact solutions and lower bounds for the min-max problem in which the maximal subset diameter is minimized, instead of the average variance. It can be shown to be a valuable criterion as well at the high resolution limit.

References

[1] T. Y. Berger-Wolf and E. M. Reingold. Optimal index assignment for multichannel communication. In *ACM-SIAM Symposium on Discrete Algorithms*, January 1999.

[2] J. Cardinal. Vector index assignments for packet communication. In *8th Symposium on Communications and Vehicular Technology in the Benelux*, pages 42–47, October 2001.

[3] K. Zeger G. Sherwood, X. Tian. Efficient image and channel coding for wireless packet networks. In *Proc. IEEE Int. Conf. Image Processing*, 2000.

[4] P. Koulgi, S. L. Regunathan, and K. Rose. Multiple description quantization by deterministic annealing. In *Proc. Int. Symposium on Information Theory*, June 2000.

[5] F. E. Oggier and S. D. Servetto. Semidefinite programs for the design of codes for delay-constrained communication in networks. In *Proc. IEEE Data Compression Conf.*, 2001.

[6] E. A. Riskin, R. E. Ladner, R.-Y. Wang, and L. E. Atlas. Index assignment for progressive transmission of full-search vector quantization. *IEEE Trans. Image Processing*, 3(3):307–312, May 1994.

[7] S. D. Servetto, V. A. Vaishampayan, and N. J. A. Sloane. Multiple description lattice vector quantization. In *Proc. IEEE Data Compression Conf.*, 1999.

[8] V. A. Vaishampayan. Design of multiple description scalar quantizers. *IEEE Trans. Inform. Theory*, 39(3):821–834, May 1993.

[9] K. Zeger and A. Gersho. Pseudo-gray coding. *IEEE Trans. Communications*, 38(12):2147–2158, December 1990.