A Rate-Distortion Optimal Alternative to Matching Pursuit

Tom Ryen, Guido M. Schuster, and Aggelos K. Katsaggelos, *Fellow, IEEE*

***Abstract*— This paper presents a method to find the operational rate-distortion optimal solution for an overcomplete signal decomposition. The idea of using overcomplete dictionaries, or frames, is to get a sparse representation of the signal. Traditionally, suboptimal algorithms, such as Matching Pursuit (MP), are used for this purpose. When using frames in a lossy compression scheme, the major issue is to find the best possible rate-distortion (RD) tradeoff. Given the frame and the Variable Length Code (VLC) table embedded in the entropy coder, the solution of the problem of establishing the best RD tradeoff has a very high complexity. The proposed approach reduces this complexity significantly by structuring the solution approach such that the dependent quantizer allocation problem reduces into an independent one. In addition, the use of a solution tree further reduces the complexity. It is important to note that this large reduction in complexity is achieved without sacrificing optimality. The optimal rate-distortion solution depends on the selection of the frame and the VLC table embedded in the entropy coder. Thus, frame design and VLC optimization is part of this work. We experimentally demonstrate that the new approach outperforms Rate-Distortion Optimized (RDO) Matching Pursuit, previously proposed in [1].**

***Index Terms*— Overcomplete dictionary, rate-distortion optimization, QR-decomposition, frame design, depth-first-search.**

I. Introduction

TRANSFORM coding is a widely used method in lossy compression. The idea in transform coding is to decorrelate the data and compact the energy of the signal in few coefficients. Low frequency coefficients are subject to fine quantization while high frequency coefficients are subject to course quantization, resulting in a number of zero coefficients. The small number of nonzero coefficients results in a *sparse* representation. An entropy coder is used as the last step in the compression scheme. A sparse representation is preferable as an input, since it can be represented with fewer bits, due to its low entropy.

In recent years, the use of overcomplete dictionaries, or *frames*, has received a lot of attention in lossy compression. A frame is a set of column vectors, just as a transform, but with a larger number of vectors than the number of elements in each vector, thus the name *overcomplete*. The basic idea of using a frame instead of a transform is that we have more vectors to choose from and thus a better chance of finding a small number of vectors whose linear combination match the signal vector well. In algorithms like Matching Pursuit (MP) [2], Orthogonal Matching Pursuit (OMP) [3], and Fast Orthogonal Matching Pursuit (FOMP) [4], the vectors are selected in sequential order. The latter algorithm is an efficient version of Order Recursive Matching Pursuit (ORMP) (and not the mentioned OMP algorithm.) The objective is to minimize the distortion subject to a sparsity constraint, where a maximum number of selected vectors or nonzero coefficients per signal block is given. Good results are achieved by using these algorithms in low bit rate compression [5]. The algorithms are fast, but suboptimal. The complexity of finding the optimal sparse representation of a signal vector is much higher when using frames instead of orthogonal transforms. A drawback with the matching pursuit algorithms is that even if we had an optimal selection of continuous valued weight coefficients, an independent scalar quantization of each coefficient would be suboptimal. This is due to the nonorthogonality between the column vectors in the frame. In addition, from a compression point of view it would be better to have the *bit rate* as the constraint, instead of the sparseness criterion, since in lossy compression the rate-distortion tradeoff is the main objective.

The major issue in *rate-distortion optimization* is to find the best tradeoff between rate and distortion. Rate-Distortion Theory (RDT) [6] has been used in many application, such as in Video compression [7], [8], [9], Shape Coding [10] and Compression of electrocardiogram (ECG) data [11]. The central entity in RDT is the Rate-Distortion Function (RDF), which is the lower bound of the distortion that is obtainable with a given bit rate. When the bit lengths of the code words are known, we can find the *Operational Rate-Distortion Function* (ORDF) [7]. When using frame coding, each combination of coefficients will result in a rate R and a distortion D, which can be viewed as an (R, D)-point in a rate-distortion diagram. An (R, D)-point is a part of the ORDF if there is no other (R, D)-points with a smaller distortion using the same or a smaller rate. A simple example of a rate-distortion diagram is shown in Figure 1. All (R, D)-points are indicated as plus signs. The circled ones are the members of the ORDF, while the solid line represent the ORDF's convex hull. A much used method to find members of the convex hull is the *Lagrangian Multiplier Method* [7], [8]. This method is essential in the way we formulate and solve our problem in the next two sections.

To the authors knowledge, this paper is the first to propose a rate-distortion optimal (RDO) solution for a compression scheme using overcomplete dictionaries. The presented approach is not only optimal in rate-distortion sense, it is also

T. Ryen is with Department of Electrical and Computer Engineering, Stavanger University College, P.O.Box 8002, 4068 Stavanger, Norway. (E-mail: tom.ryen@tn.his.no)

G. M. Schuster is with HSR Hochschule für Technik Rapperswil, Abteilung Elektrotechnik, Oberseestrasse 10, 8640 Rapperswil, Switzerland. (E-mail: guido.schuster@hsr.ch)

A. K. Katsaggelos is with Department of Electrical and Computer Engineering, Northwestern University, Evanston, Illinois 60208-3118, USA. (E-mail: aggk@ece.nwu.edu)

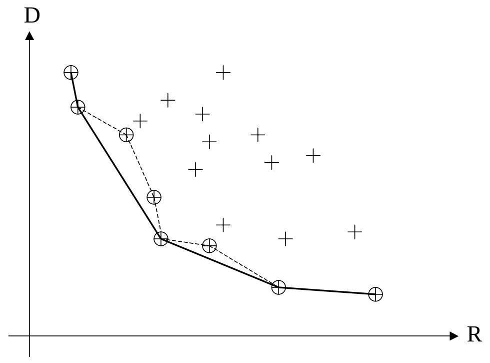


Fig. 1. Operational rate-distortion function (ORDF). All (R, D)-points are indicated as plus signs. Members of ORDF are circled. The solid line is the convex hull for the ORDF.

computationally efficient compared to other optimal algorithms. A formulation of the optimization problem is presented in Section II. The problem is shown to be very hard to solve optimally. In Section III a new problem formulation is presented, which can be solved by an algorithm of much lower complexity. Matching pursuit techniques are greedy algorithms and thus suboptimal. While the problem in Section II leads to the RD optimal solution of the problem that the Basic MP algorithm addresses, the solution of the problem in Section III is the RD optimal solution of the problem that FOMP addresses. Further complexity reduction is obtained in Section IV, first by structuring all possible solutions in a tree and using a depth-first-search strategy to find the optimal solution, followed by the introduction of lower bounds in the nodes during the search. In Section IV-C we reduce the problem by only allowing ordered vector selection. This reduces the size of the tree and consequently the number of solutions. It reduces the complexity and allows Run-length encoding (RLE) that in most cases reduces the overall bit rate. The new problem is solved optimally, now coding the *runs* between selected frame vectors instead of their *indices*. The optimal rate-distortion solution depends on the *Variable Length Code* (VLC) table embedded in the entropy coder and the frame. Thus, iterative training on a signal of the same class as the test signal is important to get a well designed frame and a good VLC table. The training procedure is described in Section V. In Section VI, we make experimental comparisons to the RDO matching pursuit algorithms described in [1]. AR(1) processes are used as test signals in all experiments. In Section VII, we summarize the paper and present our conclusions.

II. Problem Formulation

The main idea behind the proposed approach is to find the minimum distortion representation of a signal, subject to a given bit budget. Consider a one-dimensional signal, $\mathbf{x}$, represented by an $NL \times 1$ vector, divided into L blocks, each consisting of N samples. The l-th signal block, $\mathbf{x}_l$, is a column vector of length N. Consider a frame, $\mathbf{F}$, of dimension $N \times K$, where $K > N$. Given that the column vectors, $\mathbf{f}_k$, $k = 1, \ldots, K$, span the space to which $\mathbf{x}_l$ belongs, $\mathbf{x}_l$ can be written as a combination of these vectors, that is,

$$\mathbf{x}_l = \mathbf{F}\,\mathbf{w}_l = \sum_{k=1}^{K} w_{l,k}\mathbf{f}_k, \tag{1}$$

where $\mathbf{w}_l$ is the continuous valued coefficient vector for the l-th signal block and $w_{l,k}$ is the k-th element in $\mathbf{w}_l$. The frame's column vectors are all of unit length. In a compression scheme, we deal with a sparse and quantized coefficient vector, $\tilde{\mathbf{w}}_l$, which will result in the reconstructed signal vector, $\tilde{\mathbf{x}}_l = \mathbf{F}\,\tilde{\mathbf{w}}_l$. Quantization of the coefficients is necessary to achieve a bit efficient representation of the signal.

We define both the bit rate and the distortion for each block to be independent, i.e., the total bit rate, R, and the total distortion, D, is the sum of the rate and the distortion for each block, respectively. The distortion, D_l, for block l is defined by

$$D_l(\tilde{\mathbf{w}}_l) = \left\|\mathbf{x}_l - \tilde{\mathbf{x}}_l\right\|^2 = \left\|\mathbf{x}_l - \mathbf{F}\,\tilde{\mathbf{w}}_l\right\|^2. \tag{2}$$

Since we expect a large number of the K elements in $\tilde{\mathbf{w}}_l$ to be equal to zero, only the quantized *values* of the nonzero coefficients and their corresponding *indices* are encoded. After the last nonzero coefficient in each block, we use an End Of Block (EOB) symbol to indicate the start of the next block. We use two distinct VLC tables of finite length, one with *value* symbols and one with *index* symbols. The EOB codeword should not be confused with any of the *index* codewords. We can now define the rate, R_l, for block l as

$$R_l(\tilde{\mathbf{w}}_l) = \sum_{k\in nz} (R^{val}_{l,k} + R^{ind}_{l,k}) + R^{EOB}, \tag{3}$$

where nz is the set of indices for the nonzero coefficients, $R^{val}_{l,k}$ and $R^{ind}_{l,k}$ the number of bits used to code the *value* and the *index* for the k-th coefficient, respectively, and R^{EOB} the number of bits needed to transmit the EOB symbol. If there are no vectors selected to represent signal block l, only the EOB symbol needs to be transmitted and $R_l(\tilde{\mathbf{w}}_l) = R^{EOB}$.

Our goal is for a given $\mathbf{x}$ to choose the appropriate vectors $\tilde{\mathbf{w}}_l$ so that the distortion of the reconstruction is minimized subject to a given bit budget, R_{budget}. We can formulate our initial optimization problem as

$$\begin{aligned} \min_{\tilde{\mathbf{w}}_l} \quad & \sum_{l=1}^{L} D_l(\tilde{\mathbf{w}}_l) \\ \text{s.t.} \quad & \sum_{l=1}^{L} R_l(\tilde{\mathbf{w}}_l) = R_{budget}. \end{aligned} \tag{4}$$

This is an integer optimization problem due to the discrete valued $\tilde{\mathbf{w}}_l$ and nonlinear due to (2). We relax the problem using the *Lagrangian Multiplier Method*. That is, the following unconstrained optimization is performed

$$\begin{aligned} &\min_{\tilde{\mathbf{w}}_l} \Big(\sum_{l=1}^{L} D_l(\tilde{\mathbf{w}}_l) + \lambda \sum_{l=1}^{L} R_l(\tilde{\mathbf{w}}_l) \Big) \\ &= \sum_{l=1}^{L} \Big[\min_{\tilde{\mathbf{w}}_l} \Big(D_l(\tilde{\mathbf{w}}_l) + \lambda R_l(\tilde{\mathbf{w}}_l) \Big) \Big], \end{aligned} \tag{5}$$

for $\lambda \in \mathbb{R}^+$. After (5) is solved optimally the appropriate λ needs to be chosen so that the rate constraint is satisfied. By using the formulation in (5) with several different λ-values, we can find segments of the ORDF's convex hull. This is of great value in a quality study of the compression scheme. Note that the signal blocks are independent and hence we were able to move the minimization inside of the summation operator in (5). We would have liked a similar situation within each block, that is, for the rate and distortion to be minimized with respect to each of the coefficients of $\tilde{\mathbf{w}}_l$, independently. Unfortunately, the distortion is dependent on all coefficients of $\tilde{\mathbf{w}}_l$, due to the nonorthogonal decomposition, and this dependency can not be decoupled. Hence the minimization can not be carried out coefficient by coefficient, but needs to be carried out for every combination of coefficient values. The complexity of the minimization in (5) is high, since for every block we need to search among all possible ways to place M nonzero elements in a coefficient vector of length K. $M = 1, 2, \cdots, M_{max}$, where M_{max} is the largest number of nonzero coefficients we choose to use for a single signal block. The number of combinations with M nonzero coefficients is $\binom{K}{M}$. In addition, each nonzero coefficient can take on a given number of different *value* symbols, J. For each combination, the number of solutions is J^M. Thus, the total number of different solutions is $\sum_{M=0}^{M_{max}} \binom{K}{M} J^M$. In all practical cases, $M_{max} << K$, due to the sparse representation idea. In this paper we will use 16×32 frames, i.e., $K = 32$. Assume that all nonzero coefficient can take on $J = 32$ different values. For $M = \{1, 2, 3, 4\}$, $\binom{K}{M} = \{32, 496, 4960, 35960\}$ and $J^M = \{32, 1024, 32768, 1048576\}$. A way to avoid the latter combinatorial explosion is presented in the next section, where we introduce a computationally efficient algorithm that allows us to find the optimal solution. This is achieved by first moving to an orthogonal space, were the distortion becomes the sum of the coefficient distortions, and then reformulating the optimization problem in that space. These two steps allow us to reduce the J^M complexity above to an $JMM!$ complexity. By choosing the right search strategy, the complexity can be reduced from $JMM!$ to $JM!$ for each combination of vectors. Together this is a tremendous reduction, keeping in mind that $M << J$ in all practical cases. For example, in the above case with $M = 4$ this reduction is in the order of 1365.

III. Proposed solution approach

In this section we introduce the core contribution of this work, which is the formulation and solution of the optimization problem. The set of selected frame vectors is orthonormalized before coefficient quantization. Due to this, the computational complexity of finding the rate-distortion optimal solution is radically reduced.

A. Orthonormalization of selected frame vectors

We first choose a particular set of M frame vectors, and find an optimal solution for this particular selection. If the M vectors were orthonormal, the total distortion would be the sum of the coordinate distortions. However, this is not the case since these M vectors are not necessarily orthonormal and therefore the optimization problem still involves dependent quantizers, and is therefore very complex. We now force this orthonormal condition on the problem by using a *QR-decomposition* [12], which results in a new space where the total distortion is simply the sum of the coordinate distortions. Hence in this space, there are no dependencies among the coefficients and therefore the problem is now an independent quantizer allocation problem, which is much faster to solve as the set of optimal quantizers for each coefficient is also the optimal solution to the overall problem.

For a given combination of M nonzero coefficients, the rate for the *index* symbols is known, since we know the VLC table in the entropy coder. Yet, we still do not know the distortion nor the rate for the *value* symbols. It is always the case that $M < N$, due to the sparse representation idea. Let us define a new matrix, $\mathbf{\Phi}_l$, which is formed by the column vectors of $\mathbf{F}$ corresponding to the nonzero coefficients. $\mathbf{\Phi}_l$ will have dimensions $N \times M$ and represent an *undercomplete* set of vectors. Let $\mathbf{v}_l = [v_{l,1}, \cdots, v_{l,M}]^T$ and $\tilde{\mathbf{v}}_l = [\tilde{v}_{l,1}, \cdots, \tilde{v}_{l,M}]^T$ be the nonzero coefficients of $\mathbf{w}_l$ and $\tilde{\mathbf{w}}_l$, respectively. Since $M < N$, the best reconstruction of signal vector $\mathbf{x}_l$ when continuous valued coefficients are used is given by

$$\hat{\mathbf{x}}_l = \mathbf{\Phi}_l \mathbf{v}_l = \mathbf{\Phi}_l (\mathbf{\Phi}_l^T \mathbf{\Phi}_l)^{-1} \mathbf{\Phi}_l^T \mathbf{x}_l, \tag{6}$$

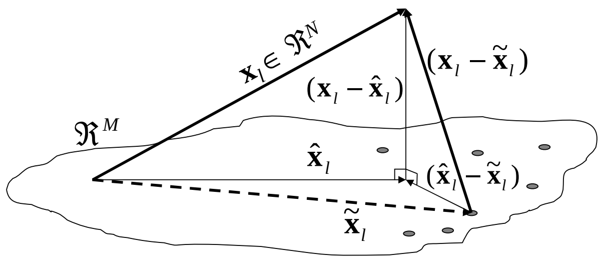
due to the Best Approximation Theorem [12]. The error is orthogonal to any vector spanned by the column vectors in $\mathbf{\Phi}_l$. Using the Pythagorean theorem the distortion in block l can be written as

$$D_l = \|\mathbf{x}_l - \tilde{\mathbf{x}}_l\|^2 = \|\mathbf{x}_l - \hat{\mathbf{x}}_l\|^2 + \|\hat{\mathbf{x}}_l - \tilde{\mathbf{x}}_l\|^2. \tag{7}$$

This is illustrated in Figure 2, where the dots represent possible values the reconstructed vector, $\tilde{\mathbf{x}}_l$, can take. The first term in (7) is a constant due to (6), since both $\mathbf{x}_l$ and $\mathbf{\Phi}_l$ are known when the set of selected vectors is known. We can now focus on the second term of the equation. The column vectors in $\mathbf{\Phi}_l$ are not necessarily orthogonal, and therefore we still have dependencies between the coefficients. By using QR-decomposition, we can get an orthogonal version of $\mathbf{\Phi}_l$. We can write $\mathbf{\Phi}_l = \mathbf{Q}_l \mathbf{R}_l$, where $\mathbf{Q}_l$ is an $N \times M$ matrix with all vectors orthonormal, and $\mathbf{R}_l$ an $M \times M$ upper triangular matrix. Let us define new coefficient vectors related to the orthonormal basis as $\mathbf{v}_l^o = \mathbf{R}_l \mathbf{v}_l$ and $\tilde{\mathbf{v}}_l^o = \mathbf{R}_l \tilde{\mathbf{v}}_l$. We can write

$$\begin{aligned}
\|\hat{\mathbf{x}}_l - \tilde{\mathbf{x}}_l\|^2 &= \|\mathbf{Q}_l \mathbf{R}_l \mathbf{v}_l - \mathbf{Q}_l \mathbf{R}_l \tilde{\mathbf{v}}_l\|^2 \\
&= (\mathbf{v}_l^o - \tilde{\mathbf{v}}_l^o)^T \mathbf{Q}_l^T \mathbf{Q}_l (\mathbf{v}_l^o - \tilde{\mathbf{v}}_l^o) \\
&= \|\mathbf{v}_l^o - \tilde{\mathbf{v}}_l^o\|^2 = \sum_{m=1}^{M} (v_{l,m}^o - \tilde{v}_{l,m}^o)^2.
\end{aligned} \tag{8}$$

Due to the QR-decomposition of $\mathbf{\Phi}_l$ the computation of (6) is simplified. That is, the projection matrix, $\mathbf{\Phi}_l (\mathbf{\Phi}_l^T \mathbf{\Phi}_l)^{-1} \mathbf{\Phi}_l^T$, can be replaced by $\mathbf{Q}_l \mathbf{Q}_l^T$ in the following way:



$\mathbf{x}_l \in \Re^N$
$(\mathbf{x}_l - \tilde{\mathbf{x}}_l)$
$(\mathbf{x}_l - \hat{\mathbf{x}}_l)$
$\Re^M$
$\hat{\mathbf{x}}_l$
$(\hat{\mathbf{x}}_l - \tilde{\mathbf{x}}_l)$
$\tilde{\mathbf{x}}_l$

Fig. 2. A visualization of the orthogonality between the minimum error vector and the subspace spanned by the column vectors in $\mathbf{\Phi}_l$. The dots represent possible values $\tilde{\mathbf{x}}_l$ can take on.

$$\begin{aligned}
& \mathbf{\Phi}_l(\mathbf{\Phi}_l^T\mathbf{\Phi}_l)^{-1}\mathbf{\Phi}_l^T \\
= \;& \mathbf{Q}_l\mathbf{R}_l(\mathbf{R}_l^T\mathbf{Q}_l^T\mathbf{Q}_l\mathbf{R}_l)^{-1}\mathbf{R}_l^T\mathbf{Q}_l^T \\
= \;& \mathbf{Q}_l\mathbf{R}_l(\mathbf{R}_l)^{-1}(\mathbf{R}_l^T)^{-1}\mathbf{R}_l^T\mathbf{Q}_l^T \\
= \;& \mathbf{Q}_l\mathbf{Q}_l^T. \qquad (9)
\end{aligned}$$

It should be noted that we expect the upper triangular matrix, $\mathbf{R}_l$, to be invertible, i.e., all its diagonal elements are nonzero. This is true when all column vectors of $\mathbf{\Phi}_l$ are linearly independent. If the column vectors are linearly dependent, there are redundancies in the representation and one of the dependent vectors should be removed. Thus, $\mathbf{R}_l$ will always be invertible. It should also be noted that if the order of the selected frame vectors is changed, $\mathbf{Q}_l$ and $\mathbf{R}_l$ will be different. Having M vectors, the number of ways to order them is $M!$, and consequently the number of different QR-decompositions is $M!$.

The importance of (8) is that in the orthonormal space, the distortion is simply the sum of the coefficient distortions. More specifically, D_l in (7), can be written as a constant plus the sum of *independent* coefficient distortions.

B. Reformulation of the optimization problem

The original optimization problem in (4) is hard to solve since the distortion cannot be written as the sum of coefficient distortions. In the previous section we have shown an orthogonalization procedure, which results in the distortion being the sum of the coefficient distortions in an orthogonal space. The original optimization problem, however, asks for the quantization and encoding of the original decision vectors, which are not in this orthogonal space. In this section we reformulate the original optimization problem, in that we quantize and encode the orthogonal coefficients, which has the effect that the distortion and the rate become additive with respect to its orthogonal coefficient rates and distortions. I.e., the dependencies between the orthogonal coefficients are decoupled. We will from now on use the coefficient vector $\tilde{\mathbf{v}}_l^o = [\tilde{v}_{l,1}^o, \cdots, \tilde{v}_{l,M}^o]^T$ as the decision variable. For a given combination of M nonzero coefficients and a given order of selected frame vectors, we know $\mathbf{\Phi}_l$. We find $\mathbf{Q}_l$ and $\mathbf{R}_l$, the QR-decomposition of $\mathbf{\Phi}_l$. The new problem, for signal block l, is

$$\begin{aligned}
& \|\mathbf{x}_l - \mathbf{Q}_l\mathbf{Q}_l^T\mathbf{x}_l\|^2 \\
+ \;& \lambda\Big(\sum_{m=1}^{M}(R_{l,m}^{ind}) + R^{EOB}\Big) \\
+ \;& \sum_{m=1}^{M}\min_{\tilde{v}_{l,m}^o}\big((v_{l,m}^o - \tilde{v}_{l,m}^o)^2 + \lambda R_{l,m}^{val}\big), \qquad (10)
\end{aligned}$$

minimized with respect to $\tilde{\mathbf{v}}_l^o$. The first term of (10) is the first term of (7), found by (6) and (9). Note that the rates, $R_{l,m}^{ind}$ and $R_{l,m}^{val}$, now depend on the coefficients of the new decision vector, since these are the coefficients encoded and transmitted. From (10) it becomes clear that the problem is now much faster to solve, since only JM comparisons are necessary, compared to the J^M comparisons needed in our original problem. The continuous valued coefficient vector, $\mathbf{v}_l^o$, is found by

$$\mathbf{v}_l^o = \mathbf{R}_l\mathbf{v}_l = \mathbf{R}_l(\mathbf{\Phi}_l^T\mathbf{\Phi}_l)^{-1}\mathbf{\Phi}_l^T\mathbf{x}_l = \mathbf{Q}_l^T\mathbf{x}_l. \qquad (11)$$

To find the optimal rate-distortion tradeoff, we must solve (10) for all $\binom{K}{M}M!$ combinations for $M = \{1, \ldots, M_{max}\}$, and store the minimum of all solutions. When working with low bit rate compression, the maximum number of nonzero coefficients per block, M_{max}, is low, resulting in a computationally manageable problem.

It is the orthogonal coefficient vector, $\tilde{\mathbf{v}}_l^o$, and the corresponding indices that are entropy encoded and transmitted through the channel. Therefore, a QR-decomposition is required in the decoder in addition to knowledge of the frame and the VLC tables used in the encoder. For every signal block, l, the set of selected frame vectors, $\mathbf{\Phi}_l$, is found by decoding the *index* codewords. $\mathbf{Q}_l$ is found from $\mathbf{\Phi}_l$, and the reconstructed signal vector, $\tilde{\mathbf{x}}_l$, is found by

$$\tilde{\mathbf{x}}_l = \mathbf{Q}_l\tilde{\mathbf{v}}_l^o. \qquad (12)$$

A full search through all combinations of up to M_{max} frame vectors (including all vector ordering permutations) is necessary in order to find the optimal solution. Yet, there is a considerable amount of time savings by choosing the right search strategy. This is the topic of Section IV.

The Fast Orthogonal Matching Pursuit (FOMP) approach [1] follows similar steps to the ones we propose in this paper. In other words, the column vector in the frame, $\mathbf{F}$, is first found that when multiplied by a weight (which needs to be determined) provides the best match to the signal. The remaining column vectors in $\mathbf{F}$ are then orthogonalized with respect to the selected vector, and the selection of a new vector from these remaining ones is repeated, as described in the previous sentence. This current step is then repeated until until a terminating criterion is met. FOMP is clearly based on a heuristic and therefore does not provide an RD optimal solution. In this work, we orthogonalize the new candidate vectors to the previously selected ones, as well, and choose the vector that gives us the best RD tradeoff. This is done for *all* combinations of up to M_{max} vectors. If the value of M_{max} is at least as large as the largest number of coefficients needed per signal block, our new approach finds the *rate-distortion optimal solution of the problem that the FOMP algorithm addresses*, and it does so in an efficient way. While the problem in Section II leads to the RD optimal solution of the problem that the Basic MP algorithm addresses, the solution of the problem in Section III is the RD optimal solution of the problem that FOMP addresses.

IV. Further complexity reduction using a solution tree

In this section we organize all $\sum_{M=0}^{M_{max}} \binom{K}{M} M!$ possible ways of combining up to M_{max} frame vectors. We build a solution tree for each signal block, whose purpose is to make it possible to use techniques that give us additional complexity reduction of coding each signal vector optimally.

Consider a tree where each node represents a unique set of selected vectors from the frame $\mathbf{F}$, i.e., a unique $\mathbf{\Phi}_l$. The root node represents the selection of *zero* vectors ($M = 0$). It has K children nodes. Each of these nodes represents the selection of exactly *one* frame vector ($M = 1$). The edges that connect the root node to its children nodes are named “1”, “2”, ... ,“K”, to indicate the index of the frame vector that is selected. All nodes in level 1 have $K-1$ child nodes each, all nodes in level 2 have $K-2$ child nodes, and so on, until level M_{max} where the tree is bounded and no child nodes exist. For any node in the tree (except for the nodes at level M_{max}), the branches to the child nodes are named “1”, “2”, ... ,“K”, except for the names of the branches that leads from the root node down to the current node. This is why the number of children for each node at level M is equal to $K-M$. In other words, a frame vector can never be selected twice. In Figure 3 an illustration of this tree is shown. It will have M_{max} generations, or levels. Level M will have

$$\prod_{m=0}^{M-1}(K-m) = \frac{K!}{(K-M)!} = \binom{K}{M} M! \tag{13}$$

number of nodes. The entire tree will represent all possible combinations of $0, 1, \ldots, M_{max}$ vectors selected, including all vector ordering permutations. For each node, we find the minimum rate-distortion solution by solving (10) for the given combination. To find the global optimum solution for the signal block, we need to search through all nodes in the tree.

A. Time complexity reduction by depth-first-search

By choosing *depth-first-search* (DFS) as the search strategy in this tree, we can build the QR-decomposition for each node recursively. There are two benefits by using DFS: There are fewer computations in order to find the QR-decomposition, and only one coefficient has to be found in order to find the best set of coefficients for the respective node.

DFS starts in the root node, moving on to the first child, then to the first child of the first child, and so on, until a node in level M_{max} is reached. It then backtrack to level $M_{max}-1$. For any node in the tree, the next step in the DFS strategy is as follows: If there are unvisited children, go to one of them. If not, backtrack to parent node. DFS stops when all nodes in the tree are visited.

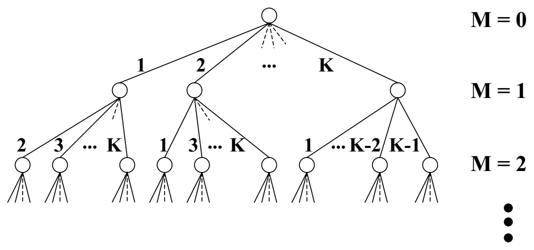


Fig. 3. Graphical representation of the solution tree. Each node represents the minimum rate-distortion solution given a unique set of selected vectors. The root node has *zero* vectors selected, each node in the 1st generation has *one* vector selected, each node in the 2nd generation has *two* vectors selected, and so on. The entire tree represents all possible combinations of selecting up to M_{max} of K frame vectors.

Suppose that we know the minimum cost, $\min C_{l,M-1}$, for a parent node at level $M-1$,

$$\min C_{l,M-1} = \|\mathbf{e}_{l,M-1}\|^2 + \alpha_{l,M-1} + \beta_{l,M-1}, \tag{14}$$

where

$$\begin{aligned}
\mathbf{e}_{l,M-1} &= \mathbf{x}_l - \mathbf{Q}_{l,M-1}\mathbf{Q}_{l,M-1}^T\mathbf{x}_l, \\
\alpha_{l,M-1} &= \lambda\Big(\sum_{m=1}^{M-1}(R_{l,m}^{ind}) + R^{EOB}\Big), \\
\beta_{l,M-1} &= \sum_{m=1}^{M-1}\min_{\tilde{v}_{l,m}^o}\big((v_{l,m}^o - \tilde{v}_{l,m}^o)^2 + \lambda R_{l,m}^{val}\big).
\end{aligned}$$

$\mathbf{Q}_{l,M-1}$ $(N \times (M-1))$ is the orthonormal basis in the QR-decomposition of $\mathbf{\Phi}_{l,M-1}$, the set of selected frame vectors. When going from a parent node to a child node, we use the same set of selected frame vectors, only adding a new frame vector, $\phi_{l,M}$, as the last column in the set of selected frame vectors, $\mathbf{\Phi}_{l,M}$ $(N \times M)$, that is

$$\mathbf{\Phi}_{l,M} = \begin{bmatrix} \mathbf{\Phi}_{l,M-1} & \phi_{l,M} \end{bmatrix}. \tag{15}$$

The new frame vector will always be added to the right end of the matrix. When using the *Gram-Schmidt process* [12] to find the QR-decomposition, we know that column vector no. i in the orthonormal basis is only a function of column vector 1 to i in the original matrix. Thus, the first $M-1$ columns in the orthonormal basis of the QR-decomposition of $\mathbf{\Phi}_{l,M}$, $\mathbf{Q}_{l,M}$, will be equal to its parent's orthonormal basis, $\mathbf{Q}_{l,M-1}$. Since we know $\mathbf{Q}_{l,M-1}$, only the last vector, $\mathbf{q}_{l,M}$, has to be found by Gram-Schmidt to get the entire $\mathbf{Q}_{l,M}$:

$$\mathbf{Q}_{l,M} = \begin{bmatrix} \mathbf{Q}_{l,M-1} & \mathbf{q}_{l,M} \end{bmatrix}, \tag{16}$$

where

$$\mathbf{q}_{l,M} = \frac{\phi_{l,M} - \sum_{m=1}^{M-1}\langle\phi_{l,M}, \mathbf{q}_{l,m}\rangle\mathbf{q}_{l,m}}{\left\|\phi_{l,M} - \sum_{m=1}^{M-1}\langle\phi_{l,M}, \mathbf{q}_{l,m}\rangle\mathbf{q}_{l,m}\right\|}. \tag{17}$$

The minimum solution for a child node at level M is given by

$$\min C_{l,M} = \|\mathbf{e}_{l,M}\|^2 + \alpha_{l,M} + \beta_{l,M}, \tag{18}$$

According to (16), the difference between $\mathbf{x}_l$ and its projection on $span(\mathbf{\Phi}_l)$ can be written as

$$\begin{aligned}\mathbf{e}_{l,M} &= \mathbf{x}_l - \mathbf{Q}_{l,M}\mathbf{Q}_{l,M}^T\mathbf{x}_l \\ &= \mathbf{x}_l - \mathbf{Q}_{l,M-1}\mathbf{Q}_{l,M-1}^T\mathbf{x}_l - \mathbf{q}_{l,M}\mathbf{q}_{l,M}^T\mathbf{x}_l \\ &= \mathbf{e}_{l,M-1} - \mathbf{q}_{l,M}\mathbf{q}_{l,M}^T\mathbf{x}_l. \qquad (19)\end{aligned}$$

The first $M-1$ vectors are not changed, thus the first $M-1$ *index* codewords are the same. A new *index* codeword is added, and $\alpha_{l,M}$ is

$$\begin{aligned}\alpha_{l,M} &= \lambda\Big(\sum_{m=1}^{M}(R_{l,m}^{ind}) + R^{EOB}\Big) \\ &= \lambda\Big(\sum_{m=1}^{M-1}(R_{l,m}^{ind}) + R^{EOB}\Big) + \lambda R_{l,M}^{ind} \\ &= \alpha_{l,M-1} + \lambda R_{l,M}^{ind}. \qquad (20)\end{aligned}$$

Each element in the coefficient vector $\tilde{\mathbf{v}}_l^o$ is found separately according to (10). Due to (11) the first $M-1$ continuous valued coefficients, $[v_{l,1}^o, \ldots, v_{l,M-1}^o]^T$, will be the same when the first $M-1$ vectors in $\mathbf{Q}_{l,M}$ are not changing. The quantized coefficients, $[\tilde{v}_{l,1}^o, \ldots, \tilde{v}_{l,M-1}^o]^T$, will be the same, and only the last added coefficient, $\tilde{v}_{l,M}^o$, needs to be found. Thus, $\beta_{l,M}$ can be written as

$$\begin{aligned}\beta_{l,M} &= \sum_{m=1}^{M}\min_{\tilde{v}_{l,m}^o}\big((v_{l,m}^o - \tilde{v}_{l,m}^o)^2 + \lambda R_{l,m}^{val}\big) \\ &= \sum_{m=1}^{M-1}\min_{\tilde{v}_{l,m}^o}\big((v_{l,m}^o - \tilde{v}_{l,m}^o)^2 + \lambda R_{l,m}^{val}\big) \\ &\quad + \min_{\tilde{v}_{l,M}^o}\big((v_{l,M}^o - \tilde{v}_{l,M}^o)^2 + \lambda R_{l,M}^{val}\big) \\ &= \beta_{l,M-1} + \min_{\tilde{v}_{l,M}^o}\big((v_{l,M}^o - \tilde{v}_{l,M}^o)^2 + \lambda R_{l,M}^{val}\big). \qquad (21)\end{aligned}$$

To find $\mathbf{q}_{l,M}$ in (17) we need $2NM+1$ multiplications and $NM-1$ additions when using Gram-Schmidt. If we would need to use Gram-Schmidt to find the entire $\mathbf{Q}_{l,M}$, the number of multiplications and additions would be equal to $\sum_{m=1}^{M} 2Nm + 1$ and $\sum_{m=1}^{M} Nm - 1$, respectively. In all practical cases, $N \gg 1$, thus the number of multiplications and additions is approximately equal to $2NM(M+1)/2$ and $NM(M+1)/2$, respectively. Computing only $\mathbf{q}_{l,M}$ instead of the entire $\mathbf{Q}_{l,M}$ will result in a reduction in the number of multiplications and additions by a factor equal to $(M+1)/2$. The complexity reduction factor is larger for larger M, i.e., the benefits of using the DFS strategy are greater for larger values of M_{max}.

Due to the knowledge of the parent node's optimal solution, the number of comparisons needed to find the optimal solution of the child node at level M is reduced. The first $M-1$ elements in the coefficient vector is the same as the parent's coefficients. Only the last coefficient must be found. In (21) the number of comparisons is reduced from JM to J.

M_{max}	2	3	4
Original problem	1	320	74411
Orthogonalized problem	0.128	6	240
Orthogonalized problem using DFS	0.066	2.1	62.6
Orthogonalized problem using DFS and LB	0.028	1.26	38.2

TABLE I

TIME UNITS RELATIVE TO CODING ONE SIGNAL BLOCK USING THE ORIGINAL FORMULATION IN (5) WHEN $M_{max} = 2$. $K = 32$ AND $J = 32$ FOR ALL CASES.

B. Time complexity reduction by introduction of lower bounds

In order to find the global optimum solution, all the nodes in the solution tree have to be visited. But, finding the optimal solution in each node is not necessarily a requirement. Let C^* be the *so far* minimum cost. As an initial value, C^* could be set to the root node cost, $C_{l,0} = \|\mathbf{x}_l\|^2 + R^{EOB}$. An alternative is to use a fast heuristic to find a suboptimal solution. C^* is updated every time a node's solution is better than C^*. For a node at level M, we define the *lower bound* (LB) for the minimum node cost as

$$LB = \|\mathbf{e}_{l,M}\|^2 + \alpha_{l,M} + \lambda M R_{min}^{val}, \qquad (22)$$

where R_{min}^{val} is the minimum number of bits for a *value* codeword in the VLC table. If $C^* < LB$ for a particular node, we know that the global optimal solution is not represented by the particular set of frame vectors, and further computation in order to find $\beta_{l,M}$ is not necessary. The DFS algorithm proceeds to the next node. In the worst case $C^* \geq LB$ in all nodes, and a full computation is necessary. But, in an average case many nodes do not need full computation, and the time saved by the introduction of lower bounds is significant. Let us show an example that supports this statement. We use the proposed algorithm to code a Gaussian AR(1) process with $\rho = 0.95$. We use a 16×32 frame trained on another AR(1) signal. The number of different coefficient values is $J = 32$. In Table I we show the time consumption when using 1) the original problem in (5), 2) the orthogonalized version in (10), 3) the orthogonalized version by DFS strategy, and 4) the orthogonalized version by DFS strategy and lower bounds (LB) technique. The experiment is done for $M_{max} = \{2, 3, 4\}$. The time units in the table are relative to coding one signal block using the original formulation when $M_{max} = 2$. The largest time reduction is achieved by the orthogonalization procedure described in Section III, particulary in the case when $M_{max} = 4$. When using the DFS strategy, the time is reduced by an additional factor equal to M_{max}. What is interesting to see is that the time is further reduced by a factor of 2 when lower bounds in each node in the tree are used. This is a significant contribution for speeding up the algorithm without loosing optimality.

C. Ordered vector selection and run-length coding

We add a constraint to our problem according to that the frame vectors are selected and QR-decomposed only in the

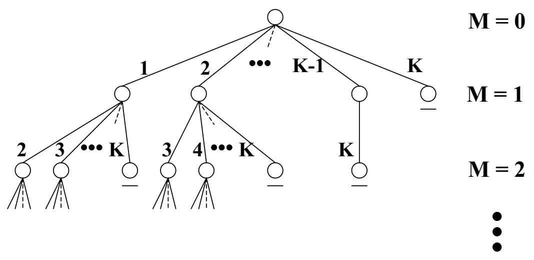


Fig. 4. The reduced solution tree. Each node represents the minimum rate-distortion solution for a set of selected vectors. The entire tree represents all possible combinations of selecting up to M_{max} of K frame vectors in the same order as they appear in $\mathbf{F}$.

		Complexity
a)	Original nonorthogonalized problem	$1+\sum_{M=1}^{M_{max}}\binom{K}{M}J^M$
b)	Orthogonalized problem	$1+\sum_{M=1}^{M_{max}}\binom{K}{M}JMM!$
c)	Orthogonalized problem using DFS	$1+\sum_{M=1}^{M_{max}}\binom{K}{M}JM!$
d)	Orthogonalized problem using DFS and ordered vector selection	$1+\sum_{M=1}^{M_{max}}\binom{K}{M}J$

TABLE II

COMPLEXITY OF OPTIMAL ENCODING OF ONE SIGNAL BLOCK.

order as they appear in the frame $\mathbf{F}$. By doing this we reduce the size of the search in the problem, since all but one of the $M!$ permutations of a given set of M vectors is avoided. In addition, it allows the use of *run-length encoding* (RLE) as part of the entropy encoder, which is a bit saving technique when the number of zero elements in the coefficient vector is large. Instead of encoding the coefficient *indices*, the number of zeros between each nonzero coefficient, *runs*, is encoded. After the last nonzero coefficient in each block, we use an End Of Block (EOB) as we did when coding the *indices* directly. The *run* for the k-th coefficient is the number of zeros *in front* of this coefficient.

The number of solutions is reduced and we can draw a new *reduced* solution tree, shown in Figure 4. This tree is not balanced like the full solution tree in Figure 3. The reduced tree is equal to the full tree at levels 0 and 1, and the edges that connect the root node to its children nodes are named “1”, “2”, ... ,“K”, indicating the vector selected. The node that represents the selection of frame vector no. 1, $\mathbf{f}_1$, has $K-1$ children nodes. These children represent the selection of *two* vectors ($M = 2$), where the first vector of $\mathbf{\Phi}_l$ is $\mathbf{f}_1$ and the second is $\mathbf{f}_i$, where $i = \{2, 3, \ldots, K\}$, respectively. But, the node representing $\mathbf{\Phi}_l = \mathbf{f}_2$ has $K-2$ children, the node with $\mathbf{\Phi}_l = \mathbf{f}_3$ has $K-3$ children, and so on. Thus, the node with $\mathbf{\Phi}_l = \mathbf{f}_{K-1}$ has only one child and the node representing $\mathbf{\Phi}_l = \mathbf{f}_K$ has none. The reduced tree will have M_{max} levels, as the full tree, but the number of nodes is reduced from $\sum_{M=0}^{M_{max}}\binom{K}{M}M!$ to $\sum_{M=0}^{M_{max}}\binom{K}{M}$. This is a considerable reduction in complexity. We do not have the same optimization problem as when coding the *indices*, since the vector selection now is ordered. But, the introduction of RLE can give us a more bit efficient representation and thus a better rate-distortion optimal solution.

An overview of the complexities of the cases described so far is given in Table II. We can see the tremendous reduction in complexity between the original nonorthogonalized problem and the orthogonalized problem using depth-first-search and ordered vector selection. In addition, we have a complexity reduction caused by the lower bound technique presented in Section IV-B. For fixed values of K and J, the complexity reduction is becoming very significant for increasing M_{max}.

So far we have presented a formulation of the optimization problem and an algorithm that finds the optimal solution in an computational efficient way. Now, we will look at the parameters that this optimal solution highly depends on, the frame and the VLC tables. Next section is about how to train these parameters.

V. FRAME DESIGN AND VLC OPTIMIZATION

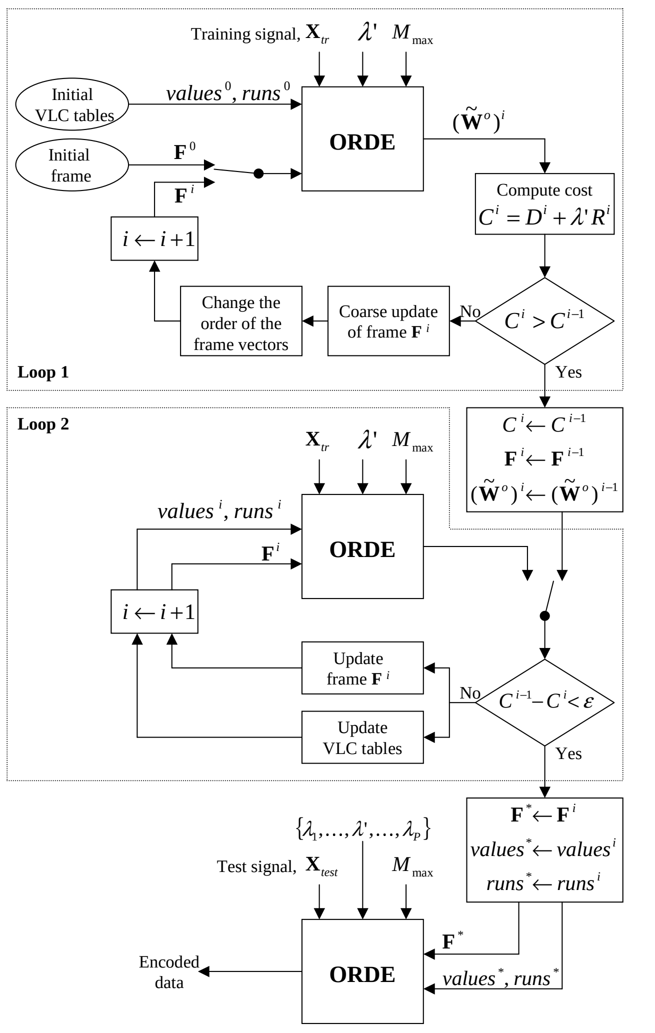
Let us call the algorithm described in the previous sections the *Optimal Rate-Distortion Encoder* (ORDE). To distinguish between the two situations described in Section IV-B and Section IV-C, the algorithms are called ORDE *ind* and ORDE *run*, respectively. From the frame design and the VLC optimization point of view, the use of ordered or not ordered vector selection has no influence, thus the denomination ORDE is used in this section. The optimality of ORDE depends on the given frame and VLC tables. In this section, we describe how we train the VLC tables and the frame for the particular class of signals that we are going to code, in order to get an improved coding result.

Let the sets of signal vectors, $\{\mathbf{x}_l\}$ and $\{\tilde{\mathbf{x}}_l\}$ and their respective sets of coefficient vectors, $\{\mathbf{w}_l\}$ and $\{\tilde{\mathbf{w}}_l\}$ be represented by the matrices $\mathbf{X}$, $\tilde{\mathbf{X}}$, $\mathbf{W}$, and $\tilde{\mathbf{W}}$, respectively. The signal matrices will have dimensions $N \times L$, and the coefficient matrices $K \times L$. When using frames in a compression scheme, it is important that the frame is well designed in order to get a sparse representation with a good quality of the reconstructed signal. The flexibility of the design is one of the benefits of using frames instead of orthogonal transforms. In [13] the *method of optimal directions* (MOD) is introduced. This is a frame training algorithm, where each iteration can be divided into two steps: 1. $\mathbf{F}$ and $\mathbf{X}$ are known, and $\mathbf{W}$ is found by using a vector selection algorithm. 2. $\mathbf{W}$ and $\mathbf{X}$ are known, and the best possible $\mathbf{F}$ is found. The vector selection algorithm used in step 1 is Orthogonal Matching Pursuit, which is a greedy algorithm. In step 2 the frame is found by

$$\mathbf{F} = \mathbf{X}\mathbf{W}^T(\mathbf{W}\mathbf{W}^T)^{-1}, \tag{23}$$

which is optimal in terms of the Mean Square Error (MSE) [13]. We note that quantization is not a part of MOD, nor any QR-decomposition, that is, the coefficient matrix with continuous valued elements, $\mathbf{W}$, is used.

In our work, we use a training scheme consisting of two iterative loops, as shown in Figure 5. The figure shows the run-length coding case, but the scheme is the same for *index* coding, by replacing “*runs*” with “*indices*”. Each loop has two steps, just as in the MOD algorithm. In both loops, step 1 is



Training signal, $\mathbf{X}_{tr}$
λ'
$M_{\max}$
Initial VLC tables
$values^0, runs^0$
ORDE
$(\tilde{\mathbf{W}}^o)^i$
Initial frame
$\mathbf{F}^0$
$\mathbf{F}^i$
Compute cost $C^i = D^i + \lambda' R^i$
$i \leftarrow i+1$
Change the order of the frame vectors
Coarse update of frame $\mathbf{F}^i$
No
$C^i > C^{i-1}$
Yes
Loop 1
Loop 2
$C^i \leftarrow C^{i-1}$
$\mathbf{F}^i \leftarrow \mathbf{F}^{i-1}$
$(\tilde{\mathbf{W}}^o)^i \leftarrow (\tilde{\mathbf{W}}^o)^{i-1}$
$\mathbf{X}_{tr}$
λ'
$M_{\max}$
$values^i, runs^i$
$\mathbf{F}^i$
ORDE
$i \leftarrow i+1$
Update frame $\mathbf{F}^i$
Update VLC tables
No
$C^{i-1} - C^i < \varepsilon$
Yes
$\mathbf{F}^* \leftarrow \mathbf{F}^i$
$values^* \leftarrow values^i$
$runs^* \leftarrow runs^i$
$\{\lambda_1, \ldots, \lambda', \ldots, \lambda_P\}$
Test signal, $\mathbf{X}_{test}$
$M_{\max}$
Encoded data
ORDE
$\mathbf{F}^*$
$values^*, runs^*$

Fig. 5. Optimal Rate-Distortion Encoder (ORDE) and training loops for the run-length coding case. The scheme is the same for *index* coding, by replacing "*runs*" with "*indices*".

the Optimal Rate-Distortion Encoder (ORDE). In this step we find the optimal set of orthogonalized and quantized coefficient vectors, $\tilde{\mathbf{W}}^o$, given the frame $\mathbf{F}$, the VLC tables for *run* (or *index*) and *value* symbols, and a training signal, $\mathbf{X}_{tr}$. In step 2 of Loop 1, there is a coarse update of the frame, only. Step 2 of the second loop, Loop 2, is a fine update of the frame in parallel with an optimization of the VLC tables.

In Loop 1, we use the same technique as MOD to find the new frame, but with the set of quantized coefficients, $\tilde{\mathbf{W}}$, instead of $\mathbf{W}$, thus

$$\mathbf{F} = \mathbf{X}_{tr}\tilde{\mathbf{W}}^T(\tilde{\mathbf{W}}\tilde{\mathbf{W}}^T)^{-1}. \tag{24}$$

The output from ORDE is the set of *orthogonalized* and quantized coefficients, $\tilde{\mathbf{W}}^o$. To generate $\tilde{\mathbf{W}}$ from $\tilde{\mathbf{W}}^o$, the QR-decomposition of each set of selected vectors from $\mathbf{F}$, $\mathbf{\Phi}_l$, has to be found. This is a straight forward operation when using ordered vector selection, since we know the combination of the selected vectors from the nonzero coefficients in $\tilde{\mathbf{W}}^o$. For the not ordered vector selection case, the information about the sequence of selected frame vectors needs to be added. Column vector no. l in $\tilde{\mathbf{W}}$, $\tilde{\mathbf{w}}_l$, is found by

$$\tilde{\mathbf{w}}_l = \mathbf{R}_l^{-1}\tilde{\mathbf{w}}_l^o. \tag{25}$$

In (24), the matrix $(\tilde{\mathbf{W}}\tilde{\mathbf{W}}^T)$ has to be full rank in order to be invertible. In situations where a particular frame vector, $\mathbf{f}_j$, has not been selected for any of the L signal blocks, $(\tilde{\mathbf{W}}\tilde{\mathbf{W}}^T)$ will not be full rank. We solve this by removing the frame vector $\mathbf{f}_j$ from $\mathbf{F}$. Let the remaining frame be named $\mathbf{F}'$. Correspondingly, row j in $\tilde{\mathbf{W}}$ is removed and the remaining $(K-1)\times L$ matrix is called $\tilde{\mathbf{W}}'$. Then we solve

$$\mathbf{F}' = \mathbf{X}_{tr}(\tilde{\mathbf{W}}')^T(\tilde{\mathbf{W}}'(\tilde{\mathbf{W}}')^T)^{-1}. \tag{26}$$

When $\mathbf{F}'$ is found, a new frame vector is added in order to get a frame of dimension $N \times K$. Instead of choosing a random vector, the signal vector $\mathbf{x}_s$ with corresponding coefficient vector, $\tilde{\mathbf{w}}_s^o$, with the highest number of nonzero coefficients is chosen. If there are more than one coefficient vectors in this category, $\tilde{\mathbf{w}}_s^o$ is the vector that in addition has the highest bit rate. By adding $\mathbf{x}_s$ to the frame, we will reduce the bit rate in the next iteration for signal block no. s, since only one frame vector will be chosen to represent $\mathbf{x}_s$. All the vectors in the updated frame are normalized, in order to get frame vectors of unit length.

The last part of step 2 in Loop 1 is a reorganization of the frame vectors. The most frequently chosen vector from the encoding of the training signal in step 1 is the first vector in the new frame. The rest of the vectors is placed in descending frequency order. This is done in order to get a higher density of low valued *runs*, and thereby an overall lower bit rate, after the update of the VLC tables in Loop 2. Due to the reorganization, the *runs* between nonzero coefficients will change dramatically from one iteration to another. Thus, an update of the VLC tables is not part of Loop 1. This reorganization has no meaning when coding *indices* instead of *runs*.

Even though (23) is optimal in the terms of MSE in [13], (24) is not optimal in this work. This is due to the fact that $\tilde{\mathbf{W}}$ is a function of $\mathbf{F}$. The algorithm will iterate as long as the computed cost for iteration i, $C^i = D^i + \lambda' R^i$, is less than C^{i-1}, the cost in the previous iteration. Loop 1 is not guaranteed to converge, nor in MSE or rate-distortion sense. However, test results in the next section show that this scheme work well and produces frames that are well suited for a given class of input data. The loop terminates at iteration i where $C^i \geq C^{i-1}$. The cost, C^i, the frame, $\mathbf{F}^i$, and the set of coefficients, $(\tilde{\mathbf{W}}^o)^i$, is set equal to C^{i-1}, $\mathbf{F}^{i-1}$, and $(\tilde{\mathbf{W}}^o)^{i-1}$, respectively, and i is set to $i-1$. The algorithm proceeds to Loop 2.

Loop 2 is guaranteed to converge to a lower or equal cost, C, for each iteration. As in Loop 1, step 1 is the ORDE, which always gives us the minimum cost. In step 2, the update of the VLC tables and the frame is done in parallel, since they are separable. That is, a change in the VLC tables would only affect the bit rate, and the update of $\mathbf{F}$ will, in this loop, only affect the distortion. The updating of the VLC tables is based on the symbol frequency distributions for the encoded version of $\tilde{\mathbf{W}}^o$. The bit rate will converge to a local optimum [10].

The update of $\mathbf{F}$ is a simple heuristic where a change in $\mathbf{F}$ is made, only if the distortion is decreasing. The overall distortion, D, is given by

$$D = \sum_{l=1}^{L} \|\mathbf{x}_l - \mathbf{Q}_l \tilde{\mathbf{v}}_l^o\|^2, \tag{27}$$

where $\mathbf{x}_l$ and $\tilde{\mathbf{v}}_l^o$ are fixed parameters in this part of the training loop. For a single frame element, f_{nk}, we add a small real valued constant, δ, and normalize the entire frame vector, $\mathbf{f}_k$. Let us call the new frame $\mathbf{F}^+$, its QR-decomposition in block l $\mathbf{Q}_l^+$, and the new overall distortion D^+. If $D^+ > D$, we subtract δ from f_{nk}, instead of adding, and proceed like described. If $D^+ < D$, $\mathbf{F} \leftarrow \mathbf{F}^+$, and the algorithm continues adding/subtracting δ to f_{nk}, normalizing $\mathbf{f}_k$, and so on. The iteration stops when $D^+ > D$. This is done for every single frame element, f_{nk}, $n = 1, \ldots, N$ and $k = 1, \ldots, K$ in sequental order.

The new frame and VLC tables are used as inputs in the next iteration of Loop 2. The iterations terminate when the cost reduction from one iteration to next is less than a predefined small number, ε. The last updated frame and VLC tables are used as parameters in the encoding of the test signal, $\mathbf{X}_{test}$. In the training loops, a predefined λ-value, λ', is used. In the encoding of the test signal, a set of λ-values, $\{\lambda_1, \ldots, \lambda_P\}$, is used in order to find several points on the ORDF's convex hull. λ' is one of the elements in this set. A predefined M_{max} is used for the entire scheme.

VI. Experimental results

In this section we show the performance of the proposed approach experimentally. In all cases, the input signal is 8192 samples of a Gaussian AR(1) process with $\rho = 0.95$, $L = 512$, and $N = 16$. As an initial frame we will use a 16×32 frame designed for the reason of using Orthogonal Matching Pursuit on an AR(1) signal [5]. We call this frame *fr4*, due to the fact that it is originally designed for the purpose of selecting 4 of 32 frame vectors per signal block. The entropy coder used in this work is an arithmetic coder. The *value* and *run* symbols from the run-length encoder are coded separately. When using index coding, the *index* and *value* symbols are also coded separately. When not stated specifically for the experiment, run-length coding is used. In all rate-distortion diagrams we show the rate in bit per sample versus the Signal-to-Noise Ratio (SNR), defined by

$$SNR = 10 \log_{10} \frac{\sum_{l=1}^{L} \|\mathbf{x}_l\|^2}{\sum_{l=1}^{L} D_l}. \tag{28}$$

In Figure 6 a learning curve is shown, where the training scheme described in Section V and Figure 5 is used. *fr4* is the initial frame, $\lambda' = 0.0002$ and $M_{max} = 5$. The circles in the curve represent the minimum cost for each iteration. The curve is monotonically decreasing, due to the described training procedure. Notice the big jump in iteration 15. This is where Loop 2 starts and the first VLC tables update takes place. The initial VLC tables have a relative flat distribution of bits for the finite set of codewords. In the VLC optimization

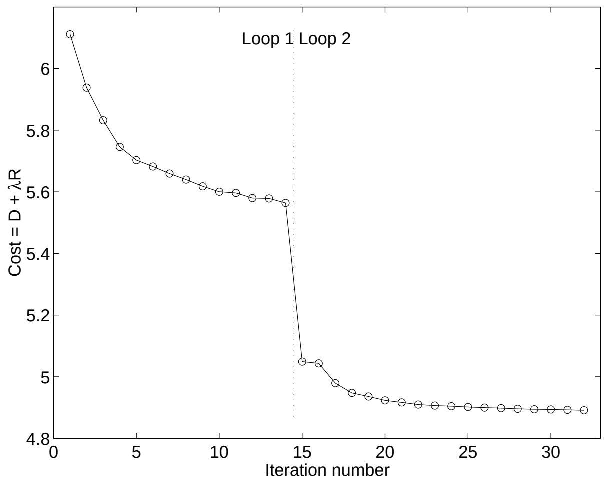


Fig. 6. Learning curve for the AR(1) training signal, for $\lambda' = 0.0002$ and $M_{max} = 5$. The cost is a function of the iteration number. The big jump in iteration no. 15 is due to the first VLC tables update.

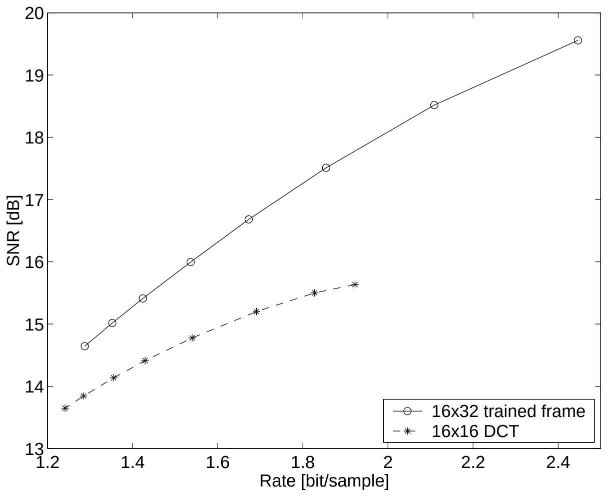
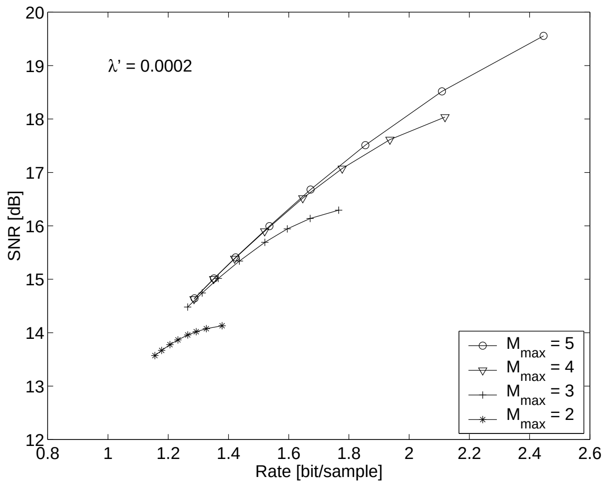


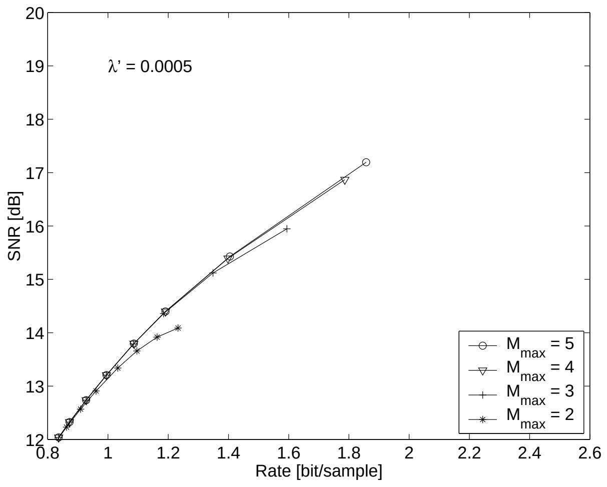
Fig. 7. The results of using the ORDE algorithm on a trained frame and a DCT, for 8 different λ-values in the range 0.0001 - 0.0008.

procedure the total bit rate is reduced, due to the update of the bit distribution that corresponds to the symbol probabilities. In Figure 7 the line with circles is the result of the optimal encoding of another AR(1) signal, where the trained frame and VLC tables are used. For $\lambda = 0.0008$, the lowest point in the curve is obtained, SNR = 14.5 dB at 1.3 bit/sample. For $\lambda = 0.0001$, the highest point in the curve, SNR = 19.5 dB at 2.45 bit/sample. In the same diagram, the results when using a 16×16 Discrete Cosine Transform (DCT) are shown. The VLC tables are optimized for the DCT case, as well. All other parameters are the same as for the trained frame case. We can see that the use of the DCT results in an SNR that is 1-2 dB lower than for the trained frame case at the same bit rate. This shows the strength of frames: They can be trained, and there are more column vectors to choose from, resulting in a better rate-distortion optimal solution.

The next experiment shows the effect of choosing the right value of M_{max}. If M_{max} is large, the time complexity is even larger, but a too small M_{max} would affect the quality



(a)



(b)

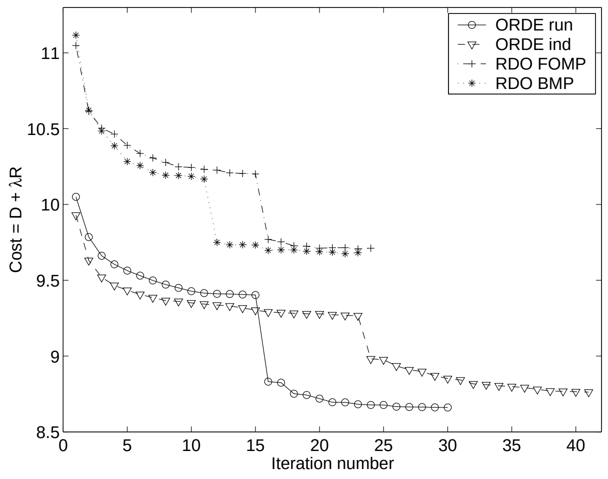
Fig. 8. Rate-Distortion results obtained by the ORDE algorithm with different values of M_{max}. 8 different λ-values were used in the range (a) 0.0001 - 0.0008 and (b) 0.00025 - 0.0020. A trained 16×32 frame is used in both examples.

of the optimal solution. If the input signal has large dynamic variations, e.g., periods where the signal is zero and periods with high energy density, it is preferable to have a skewed bit distribution. In situations like this, M_{max} should be large, to allow the high energy signal blocks to be represented by more nonzero coefficients. In any case, it is the λ-value that determines where the rate-distortion trade-off is placed. A small λ results in more bits, and a low M_{max} would in this case have a greater influence on the optimal solution than if λ is chosen to be larger. This is illustrated in Figure 8. In Figure 8(a) the λ-values are in the range 0.0001-0.0008, while in Figure 8(b) in the range 0.00025-0.0020. When M_{max} is equal to 2 and 3, we can see that the optimal rate-distortion solution is worse in the situation where λ is low.

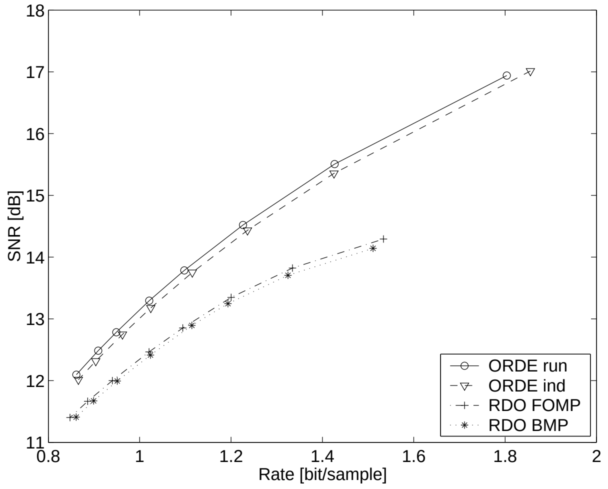
We will now compare our ORDE algorithm with Rate-Distortion Optimal (RDO) Matching Pursuit, proposed in [1]. There are two matching pursuit techniques presented in [1] that we want to compare with; RDO Basic (or standard) Matching Pursuit (RDO BMP) and RDO Fast (or fully-) Orthogonal Matching Pursuit (RDO FOMP). RDO BMP and RDO FOMP are both greedy vector selection algorithms. They are much faster than the ORDE algorithm, but they are suboptimal. In Table II d), Section IV, the algorithm complexity of ORDE is equal to $1 + \sum_{M=1}^{M_{max}} \binom{K}{M} J$. The corresponding worst case complexity for RDO BMP and RDO FOMP is equal to M_{max}. The representing frame vectors are selected one by one. The first vector selected is the one that gives the greatest reduction in the cost, $C_l = D_l + \lambda R_l$. For the RDO BMP algorithm, the residual vector is found. Then the process is repeated for the residual vector. This process stops when the cost function is no longer decreasing. For the RDO FOMP algorithm the first vector is found in the same manner as in RDO BMP. The second vector candidates are all orthogonalized with respect to the first selected one, before choosing one of them. This proceeds, and we have a QR-decomposition, just as in ORDE, but where the indices of the selected frame vectors not necessarily are in an ascending order. In the FOMP algorithm, the situation of choosing a frame vector twice is avoided. FOMP is in general better than BMP from a compression point of view [5]. As in [1] we use an arithmetic coder as entropy coder. In order to get a fair comparison, we have to use the ORDE algorithm with *index* coding (ORDE index) instead of run-length coding. The reason is that the RDO matching pursuit techniques can not include run-length encoding. The frame vectors are selected one by one and in an arbitrary order. This allows a frame vector with a lower index number than the previous selected one to be chosen. Had run-length coding been used, the new coefficient would destroy the previous selected coefficient's *run* data. Thus, index coding is used in RDO BMP and RDO FOMP.

The three algorithms, ORDE index, RDO FOMP, and RDO BMP, are utilized with the training scheme described in Section V and Figure 5. Even though we cannot compare the matching pursuit techniques directly to the ORDE with run-length coding (ORDE run), we show the latter algorithm's performance in the same diagrams. The learning curves are all shown in Figure 9(a), where $\lambda' = 0.0005$, $M_{max} = 5$, and *fr4* is used as initial frame for all cases. In all four curves the big drop in the cost, which is the intersection between training loop 1 and 2, occurs at different iterations. The total number of iteration is varying, as well. As is clearly seen, the ORDE algorithms outperforms the RDO matching pursuit algorithms. ORDE run is actually the best. The situation is the same in Figure 9(b), where the trained frame and VLC tables from the respective algorithms are used in the coding of another AR(1) process. The λ-values are varying from 0.00025 to 0.0020. The ORDE algorithms perform 0.5 dB to 1.5 dB better than the RDO matching pursuit techniques at the same bit rates. This is an improvement of 12-41 %.

With the final experiment we would like to show the differences between the four mentioned algorithms if the same frame is used without training. We will still optimize the VLC tables. The *fr4* frame is used, with $\lambda' = 0.0005$ and $M_{max} =$



(a)



(b)

Fig. 9. (a) Learning curves and (b) test results for ORDE run, ORDE index, RDO FOMP, and RDO BMP algorithm. For all cases, $\lambda' = 0.0005$ and $M_{max} = 4$. The training signal in (a) is different from the test signal in (b).

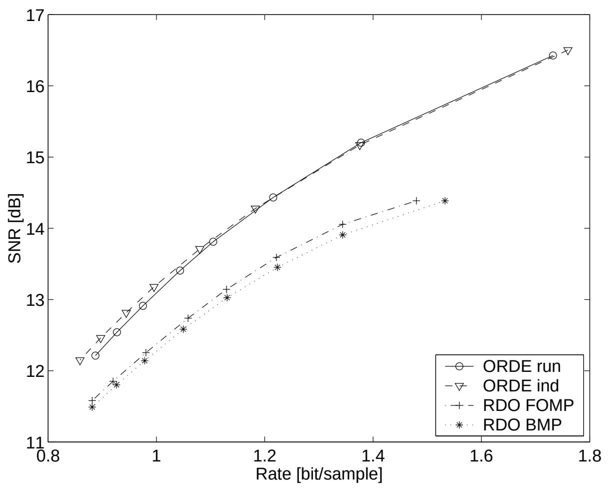


Fig. 10. Test results for ORDE run, ORDE ind, RDO FOMP, and RDO BMP, where no frame training is used. Frame is *fr4*, $\lambda' = 0.0005$, and $M_{max} = 4$.

4. In Figure 10 we see the test results with λ-values in the same range as in the previous figure. The results are approximately the same as in Figure 9(b). The proposed approach is around 15-31 % better than the RDO matching pursuit algorithms. The ORDE algorithm with run-length coding has an RDO solution that is at the same level as the ORDE with index coding, even though only *ordered* vector selection is used.

VII. Conclusions

An efficient method for rate-distortion optimal frame-based coding is presented in this paper. The efficiency is achieved by: a) using a QR-decomposition which results in a new set of independent decision variables; b) by choosing the right search strategy; and c) by using ordered vector selection that allows the use of run-length encoding. The complexity overall is reduced by a factor of J^{M-1}, where J is the number of different *values* the representing coefficients can take on, and M is the number of nonzero coefficients per signal block. This is a tremendous complexity reduction.

Experiments show that this method outperforms Rate-Distortion Optimal (RDO) Basic Matching Pursuit and RDO Fast Orthogonal Matching Pursuit. The proposed approach is 12-41 % better than the matching pursuit techniques. The benefit of using a well designed frame and optimized VLC tables is demonstrated, thus making training of these parameters essential.

References

[1] M. Gharavi-Alkhansari, "A model for entropy coding in matching pursuit," in *IEEE Proc. ICIP '98*, Chicago, USA, Nov. 1998, pp. 778–782.

[2] S. G. Mallat and Z. Zhang, "Matching pursuits with time-frequency dictionaries," *IEEE Trans. Signal Processing*, vol. 41, pp. 3397–3415, Dec. 1993.

[3] G. Davis, *Adaptive Nonlinear Approximations*, Ph.D. thesis, New York University, Sept. 1994.

[4] M. Gharavi-Alkhansari and T. S. Huang, "A fast orthogonal matching pursuit algorithm," in *IEEE Proc. ICASSP '98*, Seattle, USA, May 1998, pp. 1389–1392.

[5] K. Engan, *Frame Based Signal Representation and Compression*, Ph.D. thesis, Norwegian University of Science and Technology/ Stavanger University College, Sept. 2000.

[6] T. Berger, *Rate distortion theory: A mathematical basis for data compression*, Prentice Hall, 1971.

[7] G. M. Schuster and A. K. Katsaggelos, *Rate-Distortion Based Video Compression*, Kluwer Academic Publishers, Boston, 1997.

[8] A. Ortega and K. Ramchandran, "Rate-distortion methods for image and video compression," *IEEE Signal Processing Magazine*, pp. 23–50, Nov 1998.

[9] R. Neff and A. Zakhor, "Matching-pursuit video coding - part ii: Operational models for rate and distortion," *IEEE Trans. Circuits and Systems for Video Technology*, pp. 27–39, Jan 2002.

[10] G. Melnikov, G. M. Schuster, and A. K. Katsaggelos, "Shape coding using temporal correlation and joint VLC optimization," *IEEE Trans. Circuits and Systems for Video Technology*, pp. 744–754, Aug 2000.

[11] R. Nygaard, G. Melnikov, and A. K. Katsaggelos, "A rate distortion optimal ECG coding algorithm," *IEEE Trans. Biomedical Engineering*, pp. 28–40, Jan 2000.

[12] H. Anton, *Elementary Linear Algebra*, John Wiley and Sons, Inc., New York, 7th edition, 1994.

[13] K. Engan, S. O. Aase, and J. H. Husøy, "Multi-frame compression: Theory and design," *Signal Processing*, vol. 80, pp. 2121–2140, Oct. 2000.