UNIVERSAL MULTISCALE MATCHING PURSUITS

Murilo B. de Carvalho
Depto. de Eng. de Telecomunicações
Universidade Federal Fluminense
R. Passos da Pátria, 156
24210-240, Niterói, RJ
murilo@lps.ufrj.br

Eduardo A. B. da Silva
PEE/COPPE/DEL/EE
Universidade Federal do Rio de Janeiro
Caixa Postal 68504,
21945-970, Rio de Janeiro, RJ
eduardo@lps.ufrj.br

Weiler Alves Finamore
CETUC - PUC-Rio
Rio de Janeiro, RJ
weiler@cetuc.puc-rio.br

Abstract - In this paper we describe the UMMP (Universal Multiscale Matching Pursuits), a practical universal algorithm for multi-dimensional data lossy compression. The method is based on *approximate multiscale matching of recurrent patterns*. It uses a dictionary of *basis functions* in which the data is decomposed in the spirit of Mallat's Matching Pursuits (MP). Unlike MP however, UMMP builds its own dictionary while encoding the data, instead of using a previously defined one. Also, all basis functions can be *contracted* or *dilated* to better match the input data during the expansion. This allows the algorithm to perform arbitrarily close to the source's $D(R)$ function when the rate goes to zero. Simulation results show that it has good coding performance for a large class of data.

Resumo - Neste artigo nós descrevemos o UMMP (Universal Multiscale Matching Pursuits), um algoritmo prático universal para compressão com perdas de dados multidimensionais. O método é baseado em *casamento aproximado multiescala de padrões recorrentes*. Ele usa um dicionário de *funções base* nas quais o dado é decomposto segundo o algoritmo Matching Pursuits (MP) proposto por Mallat. Contudo, ao contrário do MP, o UMMP constrói seu dicionário enquanto codifica o dado, ao invés de usar um dicionário pré definido. Além disso, todas as funções no dicionário podem ser *contraídas* ou *dilatadas* para que se adaptem melhor ao sinal de entrada durante a expansão. Isto permite ao algoritmo um desempenho arbitrariamente próximo da função $D(R)$ da fonte quando a taxa tende a zero. Resultados de simulações mostram que o UMMP apresenta bom desempenho com uma ampla classe de dados.

Keywords: Universal Lossy Compression, Multiresolution, Matching Pursuits.

1. THE COMPRESSION PROBLEM

When we face the problem of digital signal compression, we are lead to question how much one can compress a particular signal. In other words, we want to know how many bits are necessary to describe a source of data within a given reproduction quality. This problem is studied in the rate-distortion theory, a branch of the information theory [2, 3] introduced by Shannon. In this theory a source is characterized by its statistical properties. Over the last 50 years, rate-distortion theory studies have focused to a large extent on the derivation of performance bounds for the trade-off between coding rate and achievable distortion for a given source. This trade-off is called $R(D)$ function of the source, where R is the *average* rate required to describe any output produced by the source with *average* distortion at most D. One can show that the $R(D)$ function of any source is a decreasing and convex function defined over the interval $[0, D_{max}]$ [4]. D_{max} is the smallest average distortion that can be obtained using zero bits to describe the source (at zero rate we use always the same pattern to approximate the actual output produced by the source). In other words, if we want more quality we have to pay the cost of using more bits. The $R(D)$ function defines the achievable performance of any compression code, that is, the rate of any compression code must satisfy $R \geq R(D)$.

2. CLASSICAL SOLUTIONS TO THE COMPRESSION PROBLEM

There are several methods to compress a discrete source with zero distortion at rates arbitrarily close to $R(0)$, the entropy of the source. A compression with zero distortion is called *lossless*. Huffman coding, arithmetic coding and Lempel-Ziv [4, 5] coding are examples of such lossless methods. The rate of these codes approaches the source entropy, at least when the number of source symbols processed is large. Huffman and arithmetic coders require the knowledge of a statistic model for the source in order to be implemented. This means that they are matched to a particular source and only achieve optimum performance for that source. Adaptive versions of them start with an initial model that is updated as the encoding proceeds. The Lempel-Ziv algorithm on the other hand is *universal* in the sense that no source model is required and the code rate equals the source's entropy rate when the number of source symbols tends to infinity.

2.1. LOSSY COMPRESSION: THE TWO-STEP APPROACH

A compression scheme with distortion different from zero is called *lossy*. Unlike the lossless compression case, a universal method for lossy compression whose performance achieves $R(D)$ for any source is unknown. A widely used solution to the problem of lossy compression is a *two-step* approach: In the first step, we create a version $\hat{x}$ of the source x with smaller entropy, $H(\hat{x}) < H(x)$. This step is called *quantization* and is done by a non-invertible mapping $\hat{x} = Q(x)$ (If the mapping was invertible, then the entropy $H(\hat{x})$ would necessarily equal $H(x)$). In the second step, the entropy coding step, we apply any lossless compression method to $\hat{x}$.

Two basic types of quantizers are the *scalar* and the *vector* quantizers . A scalar quantizer encodes the source in a sample-by-sample basis, while a vector quantizer (VQ) encodes blocks of N samples, or vectors, at once. The VQ divides the N dimensional space into regions and associates one reproduction vector to each region. The theory of scalar and vector quantization is well developed an tells us how to chose the intervals/regions and reproduction values/vectors to maximize different performance criteria. For example there are the Loyd-Max quantizers [6] optimized for minimum distortion when used with a given source. If we are willing to skip the entropy coding step in order to reduce the implementation complexity, this can be the best approach. However, if we wish a performance closer to $R(D)$, we can use the entropy-constrained quantizers [7], optimized to minimize distortion while keeping the output entropy fixed. Vector quantizers get performance closer (although usually not equal) to $R(D)$ than scalar ones, and performance usually increases with the number of dimensions.

2.2. LOSSY COMPRESSION: THE THREE-STEP APPROACH

Sometimes, a simpler statistical source model can be found if we perform a transformation on the data before we apply a compression scheme. This usually leads to better performance even if the compression scheme is adaptive since simpler statistics tend to be learned faster. For example, in image compression applications, almost all high performance coders employ such transformation step prior to compression. This is called the *three-step approach*: The first step is a *transformation step* where data from the source is transformed to another domain; for example the DCT (discrete cosine transform) in the JPEG coder [8] and the DWT (discrete wavelet transform) in SPIHT and EZW coders [9, 10]. This transformations allow the use of simple although useful assumptions on the source model in the transformed domain. The second step is the *quantization step* where the entropy of the transformed source data is reduced by quantizers optimized for the transformed source. The third step is the *entropy coding* step where redundancy on the quantized data is reduced by lossless compression algorithms.

Still-image codecs based on the DWT usually outperform those based on the DCT. Both DWT and DCT are linear transformations where a vector $\mathbf{X}$ of size N is represented as a linear combination of N basis vectors $\mathbf{b}_k$ as:

$$\mathbf{X} = \sum_{k=0}^{N-1} \mathcal{X}_k \mathbf{b}_k \tag{1}$$

The N coefficients $\mathcal{X}_k$ are the components of a vector $\mathcal{X}$ that is the transformed version of $\mathbf{X}$. The properties of a transformation depends on the choice of the basis vectors. For example, the DCT basis vectors are well localized in frequency but poorly localized in time. If we want to represent a vector well localized in time, like an impulse, using the DCT, we get a representation where all coefficients contain significant energy, because the information we need is spread across the hole basis. The DWT basis vectors are better spatial localized and can handle vectors like the impulse better than the DCT, but for vectors well localized in frequency the DCT can perform better than the DWT.

It would be interesting if we could have vectors well localized in time and in frequency in the same basis. One way to achieve that is to drop the linear independence constraint an move to *overcomplete* expansions using frames instead of bases.

A set of vectors $\mathcal{D} = \{\phi_k\}_{k=0}^{K-1}$ in a Hilbert space $\mathcal{H}$ is called a *frame* [11] if there are two constants $A > 0$ and $B < \infty$, such that for all $\mathbf{X} \in \mathcal{H}$:

$$A||\mathbf{X}||^2 \leq \sum_k | < \phi_k, \mathbf{X} > |^2 \leq B||\mathbf{X}||^2 \tag{2}$$

where $< ., . >$ denotes inner product.

The constants A and B are the *frame bounds* and when $A = B$ the frame is *tight*.

If $||\phi_k|| = 1$ for all k, then the constant A gives the redundancy ratio of the tight frame. For example if $A = 2$ there are twice as many vectors as needed to span the space $\mathcal{H}$.

A vector $\mathbf{X} \in \mathcal{H}$ can be represented as a linear combination of the $K > N$ vectors $\phi_k \in \mathcal{F}$ as:

$$\mathbf{X} = \sum_{k=0}^{K-1} \mathcal{X}_k \phi_k \tag{3}$$

The expansion in equation 3 is not unique because the vectors in the frame are linearly dependent. In fact, the linear dependence means that $\sum_{k=0}^{K-1} \alpha_k \phi_k = 0$ has a nontrivial solution (some $\alpha_k \neq 0$) so $\mathbf{X} = \sum_{k=0}^{K-1} (\mathcal{X}_k + \alpha_k)\phi_k$ is also a valid expansion.

3. THE MATCHING PURSUITS ALGORITHM

We can use a subset of $N < K$ vectors from $\mathcal{D}$ to approximate $\mathbf{X}$ as:

$$\hat{\mathbf{X}} = \sum_{i=0}^{N-1} \alpha_i \phi_{k_i} \tag{4}$$

We would like to minimize the error norm $||\mathbf{X}-\hat{\mathbf{X}}||^2$ in the approximation of equation 4 by choosing the N best vectors of $\mathcal{D}$ to represent $\mathbf{X}$. Instead of trying to find at once those N best vectors (it is an NP hard problem [12]) we can find a sub-optimal-N solution by a greedy algorithm called *matching pursuits* [13]. The matching pursuits algorithm attempts to optimize the choice of vectors from $\mathcal{D}$ one at a time.

The decomposition begins by choosing the k_0 value that maximizes the inner product $< \phi_{k_0}, \mathbf{X} >$. Then a *residue* $R^1\mathbf{X}$ is computed as:

$$\begin{aligned} R^0\mathbf{X} &= \mathbf{X} \\ R^1\mathbf{X} &= R^0\mathbf{X} - < \phi_{k_0}, R^0\mathbf{X} > \phi_{k_0} \end{aligned} \quad (5)$$

This residue $R^1\mathbf{x}$ is then expanded in the same way as $\mathbf{x}$. At step n we have:

$$R^{n+1}\mathbf{x} = R^n\mathbf{X} - < \phi_{k_n}, R^n\mathbf{X} > \phi_{k_n} \quad (6)$$

After N steps, the vector $\mathbf{X}$ can be approximated like in equation 4 as:

$$\hat{\mathbf{X}} = \sum_{i=0}^{N-1} < \phi_{k_i}, R^i\mathbf{X} > \phi_{k_i} \quad (7)$$

This procedure is quite general and convergence is guaranteed for any arbitrary frame as $N \rightarrow \infty$. However, if the frame is an orthonormal basis ($A = 1$ and $||\varphi_k|| = 1$), then the matching pursuits algorithm finds the optimal solution in N steps. MP has some useful properties. For example it has the *energy compaction property* since the frame elements which are closer to the vector $\mathbf{x}$ are chosen first. Therefore the vector and the coefficients used in the expansion are naturally ordered by importance. In [13] a frame of *gabor functions* (time-scaled, shifted and modulated Gaussian functions) was used with matching pursuits defining an adaptive time-frequency transform with good resolution both in time and frequency.

The three-step solution for the compression problem has some important characteristics:

- The optimal transform may be data dependent.
- The design of the quantizers rely on optimization techniques and are strongly dependent on the source model's choice. If the source statistics varies widely or is unknown, some kind of adaptive quantization should be used.
- The transform, the quantizer and the entropy code can be designed independently but the performance usually improves if we jointly optimize the three steps. For example, in [7] an ECVQ is optimized considering a huffman code for entropy coding.

This characteristics lead us to search for adaptive methods for lossy data compression that combine the three steps in one.

In the next section we describe The UMMP algorithm. It is based on matching pursuits, but it creates its own frame while expanding the data. It also uses string matching to achieve entropy coding, merging transformation, quantization and entropy coding in a single step.

4. UMMP DESCRIPTION

The Matching Pursuits algorithm (MP), described in section 3., operates with an overcomplete fixed dictionary $\mathcal{D}$ that is known before the expansion of the data begins. In a sense the expansion is adaptive since MP explicitly chooses the subset $\mathcal{D}_{opt}$ of the dictionary $\mathcal{D}$ that better fits the input data. However, the level of adaptation available depends on the amount of redundancy of the dictionary vectors. If the dictionary is highly redundant, the expansion is highly adaptable but the cost to encode the dictionary indexes is also high. It would be interesting to investigate some method that, instead of just using a subset $\mathcal{D}_{opt}$ of the dictionary, attempts to build $\mathcal{D}_{opt}$ directly while encoding the data. The Lempel-Ziv algorithm [5] is a lossless compression algorithm based on matching variable-sized vectors extracted from the input signal with variable-sized vectors in a dictionary. This algorithm builds its dictionary by including in it concatenations of previously encoded vectors. Since the algorithm is lossless, the components of the vectors constructed by concatenation are likely to form *typical sequences* [4]. Therefore the algorithm performs entropy coding because the dictionary is composed by typical sequences that become almost equally probable as the size of the dictionary vectors increases. We can use the rule of concatenation of previously encoded vectors to build a lossy compression algorithm based in Matching Pursuits that attempts to build a dictionary $\mathcal{D}_{opt}$.

We now describe UMMP (Universal multiscale Matching Pursuits) [1], an algorithm to lossy compress data from any source without previous knowledge of it's statistics. UMMP operates on a vector $\mathbf{X} = (X_0 \ldots X_{N-1})^T$ output by a vector source $\mathbf{x} = (x_0 \ldots x_{N-1})^T$. It uses:

1)

1. A dictionary $\mathcal{D}$ of vectors, initially set to $\mathcal{D}_0 = \{\mathbf{s}_0, \ldots, \mathbf{s}_{M-1}\}$, as the frame to expand the data,
2. A scalar quantizer defined by the set of reproducing values $\mathcal{Q} = \{\alpha_0, \ldots, \alpha_{K-1}\}$ and
3. A target distortion d^*.

The dictionary vectors are not normalized. To begin the expansion, all the vectors in $\mathcal{D}_0$ are initially scaled to size N using a transformation $\mathbf{s}_i^{(N)} = T_N^{\ell(\mathbf{s}_i)}[\mathbf{s}_i]$, where $\ell(\mathbf{s}_i)$ is the length of $\mathbf{s}_i$. The scale transformation is a function $T_N^M : \mathbb{R}^M \rightarrow \mathbb{R}^N$ that maps a vector of size M into a vector of size N. For the sake of simplicity, we will drop the superscript $\ell(\mathbf{s}_i)$ and write just $T_N[\mathbf{s}_i]$ for the scale transformation. In the spirit of matching pursuits, UMMP searches for the vector $\mathbf{s}_i^{(N)}$ which maximizes the inner product $< \mathbf{s}_i^{(N)}/||\mathbf{s}_i^{(N)}||, \mathbf{X} >= \mathbf{s}_i^{(N)^T}\mathbf{X}/||\mathbf{s}_i^{(N)}||$. It then

chooses $\alpha_k \in \mathcal{Q}$ to minimize the residue $\mathbf{R} = \mathbf{X} - \alpha_k \mathbf{s}_i^{(N)}$. If the residue's squared norm $||\mathbf{R}||^2$ is not greater than the target distortion d^* times the vector size N, then the expansion is finished. Otherwise, the residue $\mathbf{R} = (R_0 \dots R_{N-1})^T$ is parsed in two segments $\mathbf{R}' = (R_0 \dots R_{p-1})^T$ and $\mathbf{R}'' = (R_p \dots R_{N-1})^T, p \in [0, N-2]$.

For example, let
$\mathbf{X} = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 6 \end{pmatrix}^T$, $\mathcal{D}_0 = \{1\}$, $d^* = 0.5$ and
$\mathcal{Q} = \{-2, -1, -0.5, 0, 0.5, 1, 2\}$.

- We first scale all vectors in the dictionary to length 7, that is,
  $\mathbf{s}_0^s = T_7[\mathbf{s}_0] = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}^T$.
- Then we evaluate the inner product
  $< \mathbf{X}, \mathbf{s}_0^s / ||\mathbf{s}_0|| >= 10.2050$.
- To minimize the residue's norm we choose
  $\hat{\mathbf{X}} = \alpha_6 \mathbf{s}_0^s = \begin{pmatrix} 2 & 2 & 2 & 2 & 2 & 2 & 2 \end{pmatrix}^T$. Therefore
  $\mathbf{R} = \mathbf{X} - \hat{\mathbf{X}} = \begin{pmatrix} -1 & 0 & 1 & 2 & 3 & 4 & 4 \end{pmatrix}^T$ and
  $||\mathbf{R}||^2 = 47$.
- Since $||\mathbf{R}||^2 > Nd^* = 3.5$ the residue is split in two segments $\mathbf{R}'$ and $\mathbf{R}''$.

The segmentation point p can be chosen to minimize some performance criterion. For example, we could choose p such that $||\mathbf{R}' - \hat{\mathbf{R}}'||^2 + ||\mathbf{R}'' - \hat{\mathbf{R}}''||^2$ is minimum, where $\hat{\mathbf{R}}'$ and $\hat{\mathbf{R}}''$ are the best approximations to $\mathbf{R}'$ and $\mathbf{R}''$ that we can make using the current dictionary $\mathcal{D}$. In the above example, the best p equals 2 so $\mathbf{R}' = \begin{pmatrix} -1 & 0 \end{pmatrix}^T$ and $\mathbf{R}'' = \begin{pmatrix} 1 & 2 & 3 & 4 & 4 \end{pmatrix}^T$. This process is illustrated in figure 1. At this moment we have an integer sequence, composed by a scalar quantizer index $k = 6$, a dictionary index $i = 0$, a one bit flag 0 to indicate splitting and a segmentation point $p = 2$, representing the encoded data.

Next, all the vectors in $\mathcal{D}$ are scaled to size p using a transformation $\mathbf{s}_i^{(p)} = T_p[\mathbf{s}_i]$ and the same procedure is recursively applied to $\mathbf{R}'$. Then the vectors in $\mathcal{D}$ are scaled to size $N-p$ using a transformation $\mathbf{s}_i^{(N-p)} = T_{N-p}[\mathbf{s}_i]$ and the process is repeated for $\mathbf{R}''$. After that, we will have two approximations $\hat{\mathbf{R}}'$ and $\hat{\mathbf{R}}''$, and we can then form an estimate of the original residue $\mathbf{R}$ as $\hat{\mathbf{R}} = (\hat{\mathbf{R}}'^T \hat{\mathbf{R}}''^T)^T$. Finally, the dictionary $\mathcal{D}$ is updated by including $\hat{\mathbf{R}}$ and $\hat{\mathbf{R}} + \alpha_k \mathbf{s}_i^{(N)}$ in it. In our example, the first segment $\mathbf{R}'$ will be parsed in two, and each part will be successfully approximated by the vectors in the dictionary. After that the algorithm updates the dictionary and then proceed to encode $\mathbf{R}''$. Figure 2 illustrates the recursive application of the procedure to $\hat{\mathbf{R}}'$ and figure 3 illustrates the procedure applied to $\hat{\mathbf{R}}''$.

The UMMP algorithm has some interesting features:

i)

1. Due to the use of the multiscale approach, UMMP can perform arbitrarily close to the source $R(D)$ when $d^* \to \infty$. Other attempts on universal lossy coding, such as the Lossy Lempel-Ziv, usually have performances bounded away from the $R(D)$ at these rates [14, 15].

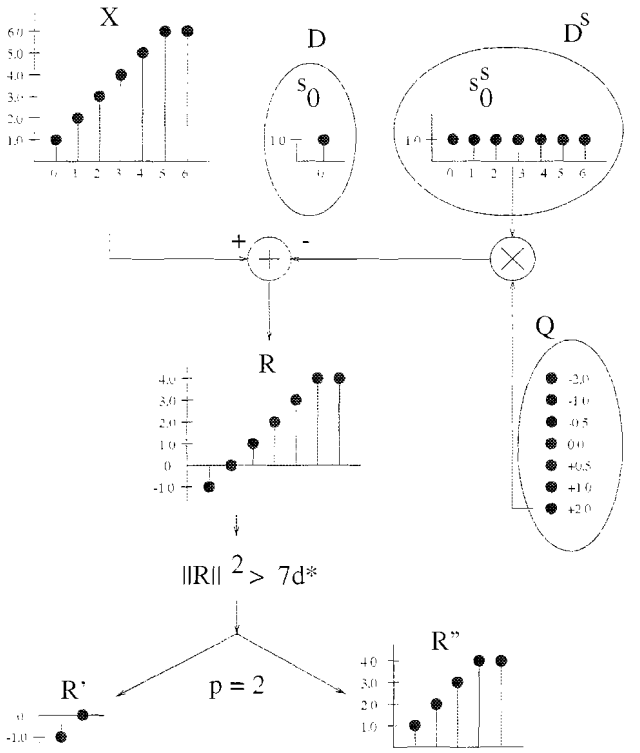
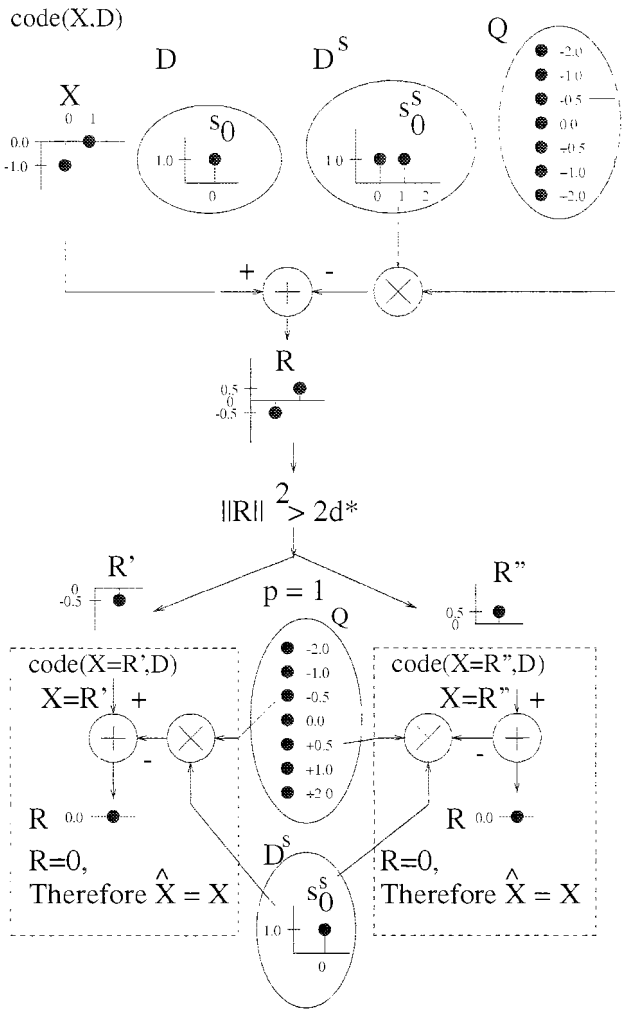


Figure 1. Segmentation of the first residue.

2. Fidelity criteria other than the mean squared error can be easily accommodated in the algorithm.
3. It is easily extendable to higher dimensions. That is, instead of operating on a vector it can operate on a matrix or multi-dimensional array.
4. It is fully adaptive and therefore universal at least in the sense that no prior knowledge of a source model is needed for it to perform best.
5. It merges the transformation, the quantization and the entropy coding steps of the three step approach in one.

We also observed , after some computer simulations, the following properties:

1. If we are using $\log_2(N)$ bits (using for example an arithmetic coder with uniform N-level distribution) to encode p, we found that it is better, in a rate-distortion performance sense, to use a fixed segmentation point $p = N/2$, and encode it with 0 bits. Unless a more complex integer code is used for p, the gain we get with the better partition is not worth the increase in bit rate.
2. Better performance is achieved if the quantizer has one reconstruction level, $\mathcal{Q} = \{1\}$. We observed that when the number of quantization level increases, the number of segmentations (and therefore the number of different vectors learned by the algorithm) decreases and vice-versa. In fact, the algorithm learns the same basis vectors with several different amplitudes when the number of quantization levels is small. Therefore, unless a complex joint entropy coding method is used to encode both the dictionary indexes and the quantizer



After returning from code(X=R',D) and code(X=R",D), D becomes:

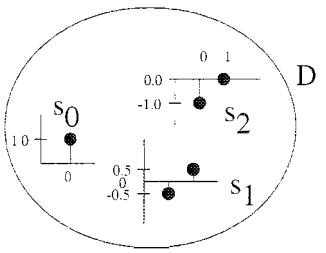


Figure 2. Coding of the first segment of the first residue.

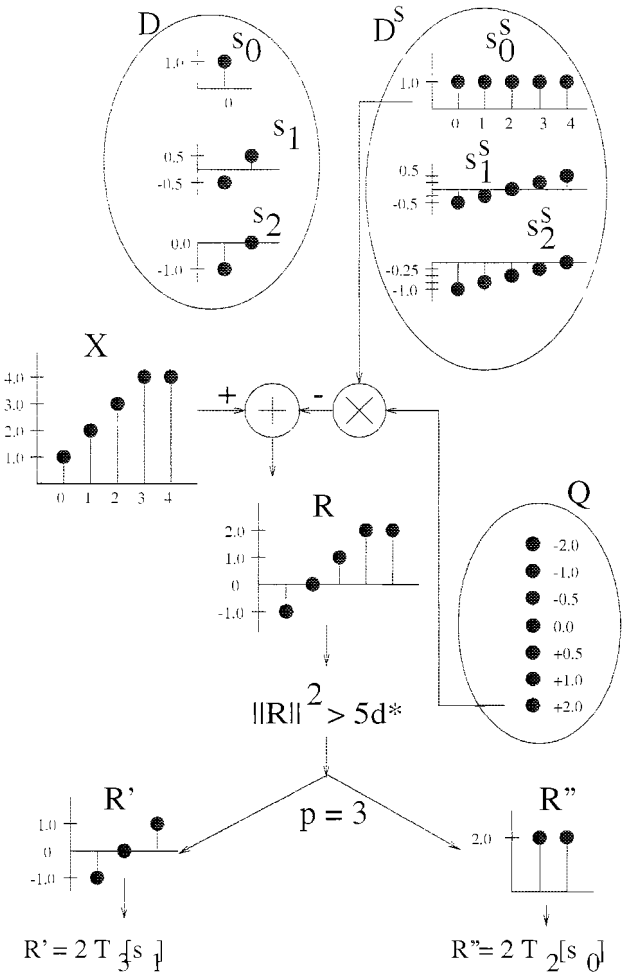


Figure 3. Coding of the second segment of the first residue.

indexes, it is better to use a one-level quantizer and allow the dictionary to explore the joint statistics.

5. IMPLEMENTATION DETAILS

We have implemented UMMP in computer programs. We wanted to evaluate the rate-distortion performance of UMMP when applied to two-dimensional data, specifically gray-scale digital images. So we implemented the two-dimensional version of UMMP, the 2D-UMMP, that operates on two-dimensional arrays, or matrices, instead of vectors. The basic difference to the one-dimensional algorithm is that the segmentation point $\mathbf{p} = \begin{pmatrix} p_l & p_c \end{pmatrix}^T$ is a point in a two-dimensional space. We use as the squared norm $||\mathbf{X}||^2$ of a matrix $\mathbf{X}$ the sum of its squared elements. If we allow generic segmentation points, a residue matrix $\mathbf{R}$ is segmented in four sub-matrixes as:

$$\mathbf{R} = \begin{pmatrix} \mathbf{R}' & \mathbf{R}'' \\ \mathbf{R}''' & \mathbf{R}'''' \end{pmatrix} \tag{8}$$

The sub-matrices in equation 8 are of sizes $p_l \times p_c$, $(p_l \times (M - p_c)$, $(N - p_l) \times p_c$ and $(N - p_l) \times (M - p_c)$. As in the one-dimensional case however, we get better performance if we use a fixed partition rule that splits the residue matrix in two. The rule we adopted is:

- `if` $\mathbf{R}$ is $N \times M$ where $N > M$, `then` $p_l = N/2$ and $p_c = 0$.
- `else` $p_l = 0$ and $p_c = M/2$.

We have implemented 2D-UMMP with this fixed segmentation rule. This implementation also used a one level scalar quantizer $\mathcal{Q} = \{1\}$ because, as in the one-dimensional case, this lead to better performance. The scale transformation in the two-dimensional case is a function $T_{N,M}^{K,L} : \mathbb{R}^{KL} \rightarrow \mathbb{R}^{NM}$ that maps a matrix of size $K \times L$ into a matrix of size $N \times M$. We have limited the matrices sizes to powers of two, that is, $\mathbf{X}$ must be of size $(N \times M) = (2^K \times 2^L)$ with K, L integers. The 2D-UMMP is described next.

Let:

- $\mathbf{X}$ be an $(N \times M) = (2^K \times 2^L)$ matrix.
- $\mathcal{D} = \{\mathbf{s}_0, \ldots, \mathbf{s}_{I-1}\}$, a dictionary with I matrices of arbitrary sizes. This dictionary must be initialized to $\mathcal{D} = \mathcal{D}_0$.
- d^* be a target distortion.
- $T_{N,M}[\mathbf{X}]$ be a scale transformation that maps the matrix $\mathbf{X}$ in a matrix of size $N \times M$.

Procedure $\hat{\mathbf{X}} = \text{encode}(\mathbf{X}, d^*)$:

step 1

1. scale all matrices in the dictionary $\mathcal{D}$ to the size of $\mathbf{X}$, that is evaluate $\mathbf{s}_i^{(N,M)} = T_{N,M}[\mathbf{s}_i]$ for $i = 0, \ldots, |\mathcal{D}| - 1$.
2. find index i in the dictionary such that $||\mathbf{X} - \mathbf{s}_i^{(N,M)}||$ is minimum and make $\hat{\mathbf{X}} = \mathbf{s}_i^{(N,M)}$.
3. output index i
4. `if` $N = M = 1$ return $\hat{\mathbf{X}}$.
   `else` go to step 5.
5. `if` $||\mathbf{X} - \hat{\mathbf{X}}||^2 \leq NMd^*$ `then` output flag '1' and return $\hat{\mathbf{X}}$.
   `else` go to step 6.
6. make $\mathbf{R} = \mathbf{X} - \hat{\mathbf{X}}$ and output flag '0'.
7. `if` $N > M$ split $\mathbf{R} = \begin{pmatrix} \mathbf{R}' \\ \mathbf{R}'' \end{pmatrix}$,
   where $\mathbf{R}'$ and $\mathbf{R}''$ are $N/2 \times M$
   `else` split $\mathbf{R} = \begin{pmatrix} \mathbf{R}' & \mathbf{R}'' \end{pmatrix}$
   where $\mathbf{R}'$ and $\mathbf{R}''$ are $N \times M/2$
8. compute $\hat{\mathbf{R}}' = \text{encode}(\mathbf{R}', d^*)$
9. compute $\hat{\mathbf{R}}'' = \text{encode}(\mathbf{R}'', d^*)$
10. `if` $N > M$ make $\hat{\mathbf{R}} = \begin{pmatrix} \hat{\mathbf{R}}' \\ \hat{\mathbf{R}}'' \end{pmatrix}$,
    `else` make $\hat{\mathbf{R}} = \begin{pmatrix} \hat{\mathbf{R}}' & \hat{\mathbf{R}}'' \end{pmatrix}$
11. make $\hat{\mathbf{X}} = \hat{\mathbf{X}} + \hat{\mathbf{R}}$.
12. $\mathcal{D} = \mathcal{D} \cup \{\hat{\mathbf{X}}\} \cup \{\hat{\mathbf{R}}\}$
13. return $\hat{\mathbf{X}}$

Decoding is easily done following the procedure below:

Procedure $\hat{\mathbf{X}} = \text{decode}(N, M)$:

step 1

1. input index i
2. `if` $N = M = 1$ return $\hat{\mathbf{X}} = T_{N,M}[\mathbf{s}_i]$.
   `else` go to step 3.
3. input flag. `if` flag = 1, return $\hat{\mathbf{X}}$.
   `else` go to step 4.
4. `if` $N > M$ make $n = N/2$ and $m = M$.
   `else` make $n = N$ and $m = M/2$.
5. compute $\hat{\mathbf{R}}' = \text{decode}(n, m)$
6. compute $\hat{\mathbf{R}}'' = \text{decode}(n, m)$
7. `if` $N > M$ make $\hat{\mathbf{R}} = \begin{pmatrix} \hat{\mathbf{R}}' \\ \hat{\mathbf{R}}'' \end{pmatrix}$,
   `else` make $\hat{\mathbf{R}} = \begin{pmatrix} \hat{\mathbf{R}}' & \hat{\mathbf{R}}'' \end{pmatrix}$
8. make $\hat{\mathbf{X}} = \hat{\mathbf{X}} + \hat{\mathbf{R}}$.
9. $\mathcal{D} = \mathcal{D} \cup \{\hat{\mathbf{X}}\} \cup \{\hat{\mathbf{R}}\}$
10. return $\hat{\mathbf{X}}$

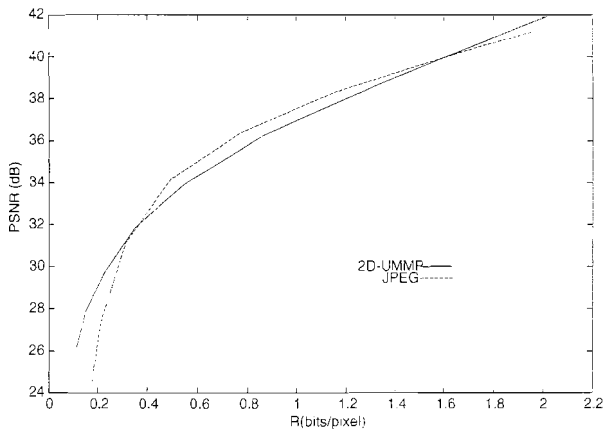


Figure 4. Rate × distortion for 2D-UMMP with LENA 512 × 512.

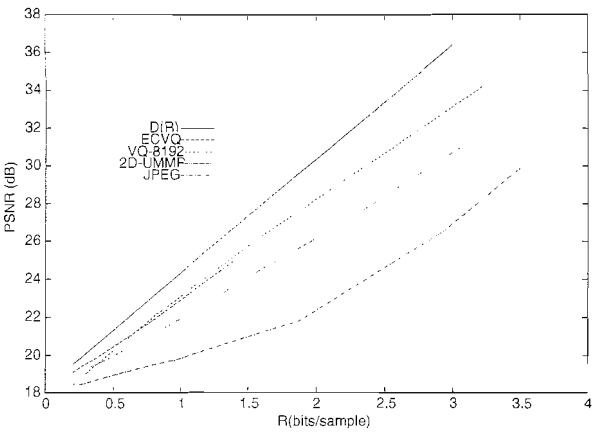


Figure 5. $R(D)$ curve for 2D-UMMP with Gaussian memoryless source.

We used a multiple dictionary approach to speed-up the execution of the algorithm as follows: Instead of applying the scale transformation $T_{N,M}[\cdot]$ to all matrices $\mathbf{s}_i$ of the dictionary $\mathcal{D}$ each time we want to use it, we keep a copy of the dictionary at each different possible scale. Therefore we only compute the scaled version of a matrix once, at the time of its inclusion in the corresponding copy of the dictionary. It should be noted that the decoding algorithm is much faster than the encoding algorithm. In fact, most of the computational effort at the encoder is on the search for the optimum element of the dictionary. The decoder, on the other hand, performs simple look-up table and update operations.

6. EXPERIMENTAL RESULTS

We used the computer programs to lossy compress the image LENA 512 × 512. The image was initially divided in 8 × 8 blocks that were processed in sequence by the algorithm. Figure 4 shows the results for 2D-UMMP. The size of the dictionary was limited to $|\mathcal{D}| \leq 8192$ by the use of a pruning strategy. Whenever a new element is included in the dictionary, the least used element is discarded. The initial dictionary was $\mathcal{D}_0 = \{-128, -124, \ldots, 0, 4, \ldots, +128\}$, of cardinality 64. The transformation $T_{NM}[\cdot]$ was a classical sampling rate change operation using a linear interpolator as the filter [16]. The integers output by the algorithm were encoded using an adaptive arithmetic coder with different models for the dictionary indexes at each scale. The flags were also encoded using the adaptive arithmetic coder with independent models for each flag. Also shown are the results for JPEG [8] for reference. The performance of UMMP with LENA is very good for a universal algorithm, being close to JPEG, a still-image-optimized transform coder.

Figure 5 shows the performance of 2D-UMMP with a Gaussian memoryless source. The parameters used in the algorithm were the same as in the case of the image LENA. The source $\mathbf{X}$ was a 256 × 256 matrix of samples of a memoryless Gaussian process with mean 128.0 and variance 960.0. Also shown are the expected performance of a vector quantizer with 8192 samples of a memoryless Gaussian vectors with the same mean and variance as $\mathbf{X}$, the performance of the ECVQ, and the optimum theoretical $R(D)$. We can see that the performance is not far from the ECVQ, that is optimized for this source, and superior to that of JPEG, optimized for other type of source. This illustrates the universal character of UMMP.

Figure 6. LENA at $R = 0.22$ bits/pixel and $PSNR = 29.73$ dB.

Figure 6 show the image LENA 512 × 512 at rate $R = 0.22$ bits/pixel and $PSNR = 29.73$ dB. Figure 7 show the image LENA 512 × 512 at rate $R = 0.55$ bits/pixel and $PSNR = 33.93$ dB.

7. CONCLUSION

We described UMMP, a universal algorithm for multi-dimensional lossy data compression based on approximate matching of recurrent patterns with scales. UMMP uses concatenations of previously encoded vectors to adaptively build a dictionary while encoding the source. The vectors in the dictionary can be dilated/contracted and are used to expand the input data. UMMP is naturally extendable to higher dimensions, effectively exploiting multi-dimensional correlations. When applied to still-image data,

Figure 7. LENA at $R = 0.55$ bits/pixel and $PSNR = 33.93$ dB.

the performance of UMMP was close to the image-optimized JPEG. When applied to data from a memoryless Gaussian source, its performance was not far from that of an ECVQ, optimized for that source. This results are very good, illustrating the universal character of UMMP.

REFERENCES

[1] M. B. de Carvalho e E. A. B. da Silva, "Compressão de sinais multi-dimensionais via decomposição em base e casamento de padrões," 17^o Simpósio Brasileiro de Telecomunicações, 1999.

[2] C.E. Shannon, "A mathematical theory of communication," Bell Syst. Tech. Journal, vol. 27, pp. 379-423, 1948.

[3] C.E. Shannon, "Coding theorems for a discrete source with a fidelity criterion," in IRE National Convention Record, Part 4, pp. 142-163, 1959.

[4] R. E. Blahut, "Principles and Practice of Information Theory" Addison-Wesley publishing Company, 1988.

[5] J. Ziv and A. Lempel, "compression of individual sequences via variable-rate coding," *IEEE Transactions on Information Theory*, vol. it-24, No. 5, pp. 530-536, September 1978.

[6] Y. Linde, A. Buzzo and R. M. Gray "An algorithm for vector quantizer design," IEEE Transactions on Communications, VOL. COM-28, No. 1, January, 1980.

[7] P. A. Chou, T. Lookabaugh and R. M. Gray, "Entropy-constrained vector quantization," IEEE Transactions on Acoustics Speech and Signal Processing, VOL. 37, N0. 1, January 1989.

[8] W. Pennebaker and J. Mitchel, "JPEG Still Image Data Compression Standard," Van Nostrand Reinhold, 1994.

[9] A. Said and W. A. Pearlman, "A new, fast and efficient image codec based on set partitioning in hierarchical trees," *IEEE transactions on circuits and systems for video technology*, vol. 6, pp. 243–250, June 1996.

[10] J. M. Shapiro, "embedded image coding using zerotrees of wavelet coeficients," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 41, pp. 3445-3462, December 1993.

[11] M.Vetterli, J. Kovačević, "Wavellets and Subband Coding," Prentice-Hall, 1995.

[12] G. Davis, "Adaptive nonlinear approximations," Ph.D. dissertation, Mathematics Departament, NYU, September 1994.

[13] S. G. Mallat and Z. Zhang, "matching pursuits with time-frequency dictionaries," *IEEE Transactions on Signal Processing*, vol. 41, No. 12, pp. 3397-3415, December 1993.

[14] W. A. Finamore, M. B. de Carvalho, and J. Kieffer, "lossy compression with the Lempel-Ziv algorithm," in 11th Brasilian Telecommunication Conf.. 1993. pp. 141-146.

[15] T. Łuczak and W. Szpankowski, " a suboptimal lossy date compression based on aproximate pattern matching," IEEE Transactions on Information Theory, Vol. 43, No. 5, september 1997.

[16] P. P. Vaidyanathan, "Multirate Systems and Filter Banks," Prentice-Hall Inc., 1993.

Murilo B. de Carvalho was born in Petrópolis, Brazil in 1964. He received the B.E. degree in Electrical Engineering from Universidade Federal Fluminense (UFF), Brazil in 1986 and the M.Sc. degree in Telecommunications from Pontifícia Universidade Católica do Rio de Janeiro (PUC-RJ) in 1994. In 1994 he joined the Department of Telecommunications Engineering of UFF. Currently he is on leave from UFF working toward his D.Sc. degree at the Department of Electronics Engineering, UFRJ. His main interests include digital image/video processing, source/channel coding and digital signal processing.

Eduardo A. B. da Silva was born in Rio de Janeiro, Brazil in 1963. He received the Engineering degree in Electronics from Instituto Militar de Engenharia (IME), Brazil, in 1984, the M.Sc. degree in Electronics from Universidade Federal do Rio de Janeiro (COPPE/UFRJ) in 1990 and the Ph.D. degree in Electronics from the University of Essex, England, in 1995. In 1987 and 1988 he was with the Department of Electrical Engineering at Instituto Militar de Engenharia, Rio de Janeiro, Brazil. Since 1989 he has been with the Department of Electronics Engineering (the undergraduate dept.), UFRJ. He has also been with the Program of Electrical Engineering (the graduate studies dept.), COPPE/UFRJ, since 1996. He is a member of the National Excellence Center in Signal Processing. He has taught short courses in image and video coding for several television broadcast

companies in Brazil. In 1994 he won the British Telecom Postgraduate Publication Prize for his IEEE paper on aliasing cancellation in sub-band coding. His research interests lie in the fields of digital signal and image processing, especially image and video coding. He is a member of the IEEE.

Weiler Alves Finamore received the B.Sc. degree in electrical engineering from the Catholic University of Rio de Janeiro (PUC-Rio) in 1969, the M.Sc. and Ph.D. degrees, both in Electrical Engineering, from the University of Wisconsin-Madison in 1974 and 1978.

From 1970 to 1979 he was with the Federal University of Pará, Belém Electrical Engineering Departament. He spent three years, 1990-1991 and 1994 , with the Rio Scientific Center, IBM Brasil as a Visiting Researcher. Since 1979 he has been with the Center for Telecommunications Studies, Catholic University of Rio de Janeiro where he is engaged in teaching and research in communications, information theory and image processing.