# OPTIMAL BIT ALLOCATION WITH SIDE INFORMATION

*Paolo Prandoni*[1] *and Martin Vetterli*[1,2]

[1] LCAV, Ecole Polytechnique Fédérale de Lausanne, Switzerland
[2] EECS Dept., University of California at Berkeley, USA
email: [prandoni,vetterli]@de.epfl.ch

## ABSTRACT

For a given set of quantizers and a data vector, the optimal bit allocation in a rate/distortion sense is the sequence of quantizers which minimizes the overall distortion for a given bit budget. In an operational framework, this sequence is dependent on the data realization rather than on its probabilistic model and the cost of describing the sequence itself becomes therefore part of the bit budget. We present an allocation algorithm based on dynamic programming which determines the optimal bit allocation taking into account the side information of describing the structure of the allocation itself; practical simplifications of the algorithm are also presented with respect to coding of continuous data sources.

## 1. INTRODUCTION

Bit allocation is a special case of the more general resource allocation problem in which a limited amount of resources has to be distributed among several recipients while minimizing a specified cost functional. In the bit allocation problem we usually have a set of different quantizers, operating at different bit rates, and a set of data sources; the goal is to choose a specific quantizer for each source so that the overall cumulative distortion measure (such as the MSE) is minimized and the total number of bits is at the same time below a prescribed bit budget.

The problem can be tackled from within two very different frameworks. In a probabilistic scenario we assume we possess the statistical description of each source and the ideal rate/distortion (R/D) curve of each quantizer. The goal is to design a quantization scheme which is asymptotically optimal for the given input class. Since a quantizer's R/D curve is generally strictly convex, the optimal bit allocation can be efficiently found using reverse water-pouring techniques or variations thereof [1]. The resulting quantization scheme can be considered "hard wired", since the design is done only once for an entire class of input data.

The second approach to resource allocation does away with statistical descriptions and deals exclusively with the particular data realization to be quantized. The first fundamental difference is that in this case we no longer rely on asymptotic R/D quantizer curves but on a discrete set of *operational* R/D points specific to each quantizer-source pair. In other words, the set of distortion values for each data source must be explicitly computed for all bitrates (see [2]). The second main difference is that, in this case, a description of the final optimal set of quantizers must be supplied along with the quantized data; this represents an unavoidable cost in term of effective bit rate, since a part of the overall bit budget must be spent to encode a sequence of quantizer descriptors. This cost is obviously data-dependent (imagine using the same quantizer for all sources versus a different quantizer for each source); true optimality for the allocation requires that this cost be part of the R/D cost functional in the minimization process and this in turn introduces dependence between allocation choices. While this second approach is definitely more expensive computationally, it can yield significant performance gains; where bitrate is a premium, it may very well be worth the extra effort, and this is the case we will treat in the following sections.

## 2. THE BIT ALLOCATION PROBLEM

Consider a sequence of $N$ data"points" $x_1^N = \{x_1, x_2, \ldots, x_N\}$; when the first and last subscripts are implicit, the data sequence will be indicated simply by $\boldsymbol{x}$. The $x_i$'s need not be identified with scalar values, while possibly being such; they are best viewed as finite subsets of signal samples: $(M \times M)$-pixel image blocks for instance, or contiguous, non overlapping segments of 1-D signals. Consider further a set of $P$ quantizers $\{q_k\}$, $k = 1, \ldots, P$ where each quantizer works at integer bitrate $b_k$; let $d_k(x_i)$ be the distortion when quantizing data point $x_i$ with quantizer number $k$. Let $\boldsymbol{k} = k_1^N$ be a sequence of $N$ indices between 1 and $P$: $D(\boldsymbol{k}, \boldsymbol{x})$ will denote the total distortion when coding the entire data vector with the sequence of quantizers indexed by $\boldsymbol{k}$; $B(\boldsymbol{k})$ will denote the rate associated to the given sequence of quantizers. The optimal bit allocation problem for a total bit budget of $B$ bits can therefore be formalized as

$$\begin{cases} \min\limits_{\boldsymbol{k}}\{D(\boldsymbol{k}, \boldsymbol{x})\} \\ B(\boldsymbol{k}) = B \end{cases} \qquad (1)$$

When rate and distortion values are independent for different data points, it is simply $D(\boldsymbol{k}, \boldsymbol{x}) = \sum_i d_{k_i}(x_i)$ and $B(\boldsymbol{k}) = \sum_i b_{k_i}$; in this case, a well-known algorithm for optimal allocation was proposed by Shoham and Gersho [3]: minimization of the global distortion is replaced by an *independent* minimization of the Lagrange cost functionals $J_i(\lambda) = d_{k_i}(x_i) + \lambda b_{k_i}$ for all data points, yielding a set of quantizer indices. If, for a given $\lambda$, the total corresponding bitrate happens to fulfill the bit budget requirement, then the corresponding set of quantizer is optimal; otherwise the minimization is iterated over different $\lambda$ values until the optimum is found. This algorithm is efficient but it has a series of practical drawbacks. First, as with all iterative algorithms, its computation time is non-fixed, depending heavily on the initial guess for $\lambda$. Second, each minimization provides the optimal allocation only for a single total rate and the minimization plus iteration process must be repeated if a different rate is desired; furthermore, a new global iteration is needed if the data vector is extended with new points. Finally, the algorithm does not work in its original form if the R/D values are not independent, as in the case of the side information requirements illustrated in the previous section.

Part of these problems were addressed in [4], where the GBFOS algorithm is used to obtain the optimal allocation for all possible rates simultaneously. The scheme is however designed for a set of quantizers with contiguous rates and, again, no provisions are made for side information or data vector extension, which makes the above algorithms more suitable to a fixed-design problem than to a strictly data-dependent encoding.

In the rest of this paper we will present a fairly general bit allocation algorithm based on dynamic programming. While dynamic programming is no new tool in this sense (dating back at least to [5]), it has not been widely used because of its purported computational complexity; for some examples see [6, 7]. We will attempt to show however that the dynamic programming framework is not only very flexible and very well suited to the problem of including side information, but offers practical algorithmic implementations which trade computational complexity for global optimality at a linear cost in the number of data points.

## 3. BIT ALLOCATION WITH SIDE INFORMATION

### 3.1. Independent Allocation

Let us initially consider the dynamic programming solution for the case of independent allocation. For any given bit budget $B$ the optimal allocation $\boldsymbol{k}^*$ is

$$\boldsymbol{k}^* = \arg \min_{\boldsymbol{k} | B(\boldsymbol{k})=B} \{D(\boldsymbol{k}, \boldsymbol{x})\} \tag{2}$$

The key observation is that, due to the additivity and non negativity of rate and distortion values:

$$\min_{k_1^n | B(k_1^n)=B} \{D(k_1^n, x_1^n)\} = \tag{3}$$

$$\min_{1 \le j \le P} \{ \min_{k_1^{n-1} | B(k_1^{n-1})=B-b_j} \{D(k_1^{n-1}, x_1^{n-1})\} + d_j(x_n)\}$$

In other words, the optimal solution at step $n$ with rate $B$ can be obtained from the partial solutions at step $n-1$ with rates "one quantizer away" from $B$. The algorithm can be organized on a trellis [8] as follows: let $S_n$ be the set of meaningful states in the trellis at step $n$; each element of $S_n$ is a $(r, k, c)$ triple where $r$ is the cumulative rate, $k$ is the last quantizer index, and $c$ is the cumulative distortion. Then:

```
Initialization:
1) let $S_0 = \{(0,0,0)\}$;
Allocation:
   At each step $n$, $1 \le n \le N$
2) let $S_n = \emptyset$;
3) precompute $d_j(x_n)$ for $j = 1, \dots, P$;
4) for all previous states $(r, k, c) \in S_{n-1}$
      for $j = 1$ to $P$
         compute new cumulative rate:
            $r' = r + b_j$
         compute new cumulative distortion:
            $c' = c + d_j(x_n)$

         (new state:)
         if there is no $(r', m, t)$ state in $S_n$ for any $m$ and $t$
           then $S_n = S_n \cup \{(r', j, c')\}$
           else if $c' < t$ (pruning)
                 then $S_n = (S_n \setminus \{(r', m, t)\}) \cup \{(r', j, c')\}$
         connect old state to new state;
Backtracking:
5) select $(r, k, c) \in S_N$ with $r = B$ (or $r$ closest to $B$)
   and collect the quantizer indices following back along the
   branches in the trellis;
```

Please note that in the end not only do we have the optimal allocation for all rates, but also the optimal allocation for all data subsets $x_1^j, j \le N$; extending the optimal allocation to a longer data vector is therefore straightforward.

The computational requirements of the algorithm can be estimated as follows. The computation of the R/D operational points requires on the order of $PN$ operations. At each step the number of trellis operations (sums, comparisons) is proportional to the number of states $|S_n|$. A simple estimate for this value is obtained noticing that, at each step, the number of different states is equal to the number of reachable rates. The smallest and largest such rates increase at each step by $b_{\min}$ and $b_{\max}$ respectively, where $b_{\min}$ and $b_{\max}$ are the rates of the minimum and maximum bitrate quantizers. The number of intermediate rates is decreased if all bitrates share a common factor, in which case only rates multiples of the factor are reachable. The number of states at step $n$ is therefore upper bounded as

$$|S_n| \le n \frac{b_{\max} - b_{\min}}{\mathrm{GCD}_j\{b_j\}} + 1 = n\Delta + 1. \tag{4}$$

Summing across all points, for a length-$N$ data vector storage and computational requirements are therefore proportional to $(\Delta/2)(N^2 + N) +$

### 3.2. Side Information

A data vector encoded with the optimal, data-dependent sequence of quantizers cannot be decoded unless the description of the sequence itself is provided. This description however will use up part of the total bit budget and therefore the final allocation cannot be considered truly optimal unless it takes into account the description's cost. While there are several different way to encode the quantizer sequence, a simple and effective way is to signal *quantizer transitions*: each time we switch to a different quantizer, we send its index. In terms of the allocation process we are introducing backward dependence in the coding rate of contiguous data points: suppose the cost of signaling a transition is $w$ bits; for a quantizer $q_j$ applied to data point $x_n$ the distortion is unchanged but the effective rate is $b_j$ or $b_j + w$ according to whether the previous quantizer for data point $x_{n-1}$ is also $q_j$ or not. Luckily, this kind of dependence is easily integrated in the previous dynamic programming framework and it determines only an increase in the size of the state space. Line 4 of the allocation algorithm can be modified as follows:

```
4) for all $(r, k, c) \in S_{n-1}$
      for $j = 1$ to $P$
         compute new cumulative rate:
            if $k = j$ then $r' = r + b_j$
                       else $r' = r + b_j + w$
         compute new cumulative distortion:
            $c' = c + d_j(x_n)$
```

```
(new state)
if there is no $(r', j, t)$ state in $S_n$ for any $t$
    then $S_n = S_n \cup \{(r', j, c')\}$
    else if $c' < t$ (pruning)
        then $S_n = (S_n \setminus \{(r', j, t)\}) \cup \{(r', j, c')\}$
connect old state to new state;
```

Note that now we are allowed to prune only between states with the same rate *and* the same quantizer index. The increase in state space size is due to the fact that the rate range at each step is augmented by $w$ bits and that states with the same rate but different quantizer index are distinct, which increases the number of states $P$ times. The upper bound in (4) remains valid with the substitution:

$$\Delta = \frac{P(b_{\max} - b_{\min} + w)}{\mathrm{GCD}_j\{b_j, b_j + w\}} \tag{5}$$

Please note that, although in this description we assumed that the cost of a transition is independent of the quantizer, the same algorithmic procedure can be used if the cost of a transition is $w(i, j)$, where $i$ and $j$ are the indices of old and new quantizer. If some prior information is available about the data which makes the use of a quantizer more likely than others, this information can be usefully incorporated in $w(i, j)$ to minimize side information. Similarly, unequal transition costs can be used in perceptual coding to penalize the use of discontinuities in modeling.

It is worthwhile to note at this point that we have not specified how the description of the allocation and the quantized data are to be merged in a single bitstream; this relates to the chosen communication protocol between encoder and decoder and is outside the scope of this paper. For an example of how to address the problem see for instance [9].

## 4. IMPLEMENTATION ISSUES

In the following section we will illustrate some implementation strategies which reduce the complexity of the algorithm in most practical cases. To facilitate the discussion, we will make use of a graphical representation of the trellis based on its *continuous approximation*. First of all notice that the quantizer rates can be normalized so that they are coprime and that the smallest rate is zero. In this case at each step $n$ the minimum rate is zero and the maximum rate is $n(q_{\max} - q_{\min}) = n\Delta$. With this normalization, any trellis can be represented graphically as a wedge (see Fig. 1): the step number $n$ runs along the $x$-axis while the rate $r$ is plotted along the $y$-axis (pointing downwards); the slope of the wedge is simply $\Delta$. In the continuous approximation we consider $n$ and $r$ as real valued quantities; this allows us to infer estimates on the number of states as surface measures: the number of states in Fig. 1 is $N^2\Delta/2$, which, for a moderately dense set of quantizers, is very close to the bound in (4) summed over all steps.

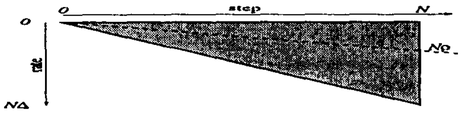

Figure 1: Continuous approximation for the trellis.

### 4.1. Zooming in

In most compression applications, we are concerned with one particular bit rate or at most with a narrow range of rates around a given bit budget. The trellis structure allows us to "zoom in" to the range of interest, reducing the total number of trellis states (and of operation) to $O(N)$.

Suppose we are interested in bit rate of $B$ at the final step $N$. At each intermediate step $n$, it is enough to keep only those states whose rate $r$ satisfies $B - (N - n)\Delta \leq r \leq B$; trellis paths through all other states will either undershoot or overshoot the target rate. This is represented in Fig. 3-(a); the total number of meaningful states for the trellis is proportional to the shaded area $W$ in the wedge, yielding

$$W = B(N - B/\Delta). \tag{6}$$

Similarly, if we zoom in on a range of rates between $B_{\min}$ and $B_{\max}$ (see Fig. 3-(b)), the number of states is proportional to $W = B_{\max}(N - B_{\min}/\Delta)$.

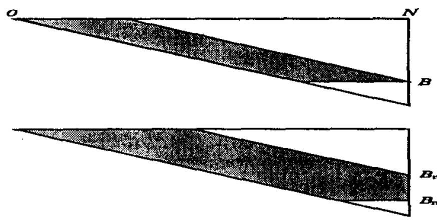

Figure 3: Zooming in: (a) single rate; (b) rate interval.

### 4.2. Block-by-block and continuous coding

Two classes of applications require us to terminate the trellis search before the global optimum is found; this happens when either the size of the data vector size is unknown a priori or it imposes storage requirements which are too large, or when there is a limit on the processing delay before quantization and encoding of data points. In both cases, the rate requirement can only be formulated in terms of an average bitrate of $\rho$ bits/symbol, with $q_{\min} \leq \rho \leq q_{\max}$; in the graphical representation the target bitrate becomes a line of slope $\rho$, as in Fig. 1.

A first approach is to partition the data into size-$L$ contiguous blocks and run the "zoom in" algorithm separately for each block for a target rate $B = \rho L$. The resulting trellis configuration is displayed in Fig. 4-(a).

A second approach, more in line with standard trellis practice, is the following. Assume we start building the trellis up to $n = L + 1$, at which point we backtrack $L$ steps and find the *locally optimal* path for rate $B = (L + 1)\rho$. For a sufficiently large $L$, we can assume that $s_1 = (r_1, k_1, c_1)$, the initial state of this path at $n = 1$, is actually the initial state of the *globally optimal path* and we can encode and send the associated first data point using quantizer $k_1$. If $s_1$ is indeed optimal, the globally optimal path will unwind inside a wedge starting at $s_1$ even if the globally optimal and the locally optimal path afterwards differ in all states but the first one. This means that at step $L + 2$ we need only update the states in $S_{L+1}$ whose rate $r$ satisfies

$$Lq_{\min} \leq r - r_1 \leq Lq_{\max} \tag{7}$$

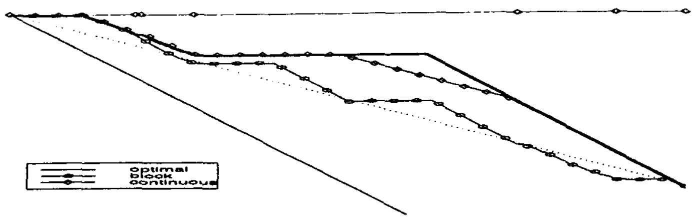

Figure 2: Block by block and continous coding solutions.

which requires on the order of $L$ operations and generates $L\Delta + 1$ new states. At the next step the process is repeated, and in general, at step $n > L$ we backtrack $L$ steps from rate $B_n = n\rho$, encode the point with the quantizer $k_{n-L}$ associated to state $s_{n-L}$, and update the subset of states in $S_n$ for which $Lq_{\min} \le r_n - r_{n-L} \le Lq_{\max}$. Finally, to ensure stability we only need to show that at each step the target rate $B_n$ is within the new reduced set of states $S_n$. This can be shown by induction: assume the proposition holds at step $n$; we backtrack and find the locally optimal rate $r_{n-L}$. Now, it must be $r_{n-L}+Lq_{\max} \le B_n \le r_{n-L}+Lq_{\min}$, otherwise $B_n$ would not be reachable in $L$ steps from $r_{n-L}$. At step $n+1$ it is $B_{n+1} = B_n + \rho$ and since $q_{\min} \le \rho \le q_{\max}$, it is also

$$r_{n-L} + Lq_{\max} + q_{\max} \le B_{n+1} \le r_{n-L} + Lq_{\min} + q_{\min}$$

and it is immediate to recognize in the two outer terms of the inequality the maximum and minimum rates in $S_{n+1}$.

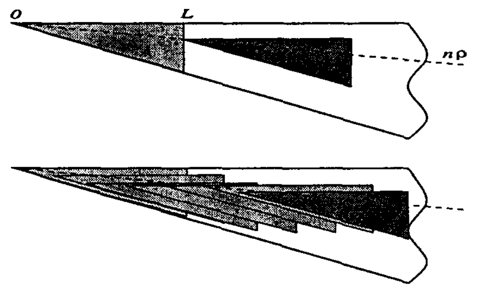

Figure 4: Continuous coding: (a) block by block; (b) backtracking.

The storage and computational costs of the algorithm are clearly linear in $n$. A graphical representation of the resulting "tiled wedges" is displayed in Fig. 4-(b). For both strategies, the resulting sub-optimality is heavily dependent on $L$. In the block by block approach, the bitrate requirement is followed exactly over length-$L$ segments; the price we pay is a high distortion if the block size is small compared to the rate of change of the signal. The continuous coding approach follows the optimal path more closely but usually does not yield a constant bitrate over fixed blocks. Both methods also introduce "out of budget" side information; while in the block algorithm however this happens only at each block boundary, in the continuous algorithm continuity of states is not preserved and the price can be substantially higher. The best tradeoff is obviously dependent on the quantization problem and on the data, and it must be necessarily tested "on the field". A numeric example is displayed in Fig. 2: a synthetic 500-point data vector is generated by a iid random number generator whose variance switches between $\alpha$ and $\beta$ according to a Poisson process with $\lambda = 0.01$. Two quantizers, 8 and 16 bits, are matched to $\alpha$ and $\beta$ and the cost of side information is 8 bits. The solid line represents the globally optimal path for a target bitrate of 4 bits/point, while the circles and the diamonds represent the block and continuous coding solutions for $L = 150$. Marks on the upper line indicate the switch points in the data source.

## 5. CONCLUSIONS

We presented in its general form an algorithm for optimal bit allocation using dynamic programming which takes into account the cost of describing the allocation itself to the decoder. While the global optimum for a length-$N$ data vector requires on the order of $N^2$ operations, in most practical cases the algorithm's complexity can be reduced to linear. Current research is under way towards minimum-delay applications in audio coding [10]; for upcoming results, please refer to:
`http://lcavwww.epfl.ch/~prandoni/optimal.html`

## 6. REFERENCES

[1] A. Gersho and R.M. Gray. *Vector Quantization and Signal Compression.* Kluver, 1992.

[2] K. Ramchandran and M. Vetterli. Best wavelet packet bases in a r/d sense. *IEEE Tran. on IP*, 2(2):160–175, April 1993.

[3] Y. Shoham and A. Gersho. Efficient bit allocation for an arbitrary set of quantizers. *IEEE Trans. Acoust., Speech, and Signal Proc.*, 36(9):1445–1453, September 1988.

[4] Eve Riskin. Optimal bit allocation via the G-BFOS algorithm. *IEEE Trans. IT*, 37(2):400–402, March 1991.

[5] A.V. Trushkin. Bit number distribution upon quantization of a random variable. *Probl. Inf. Trans.*, 16:76–79, 1980.

[6] A. Ortega, K. Ramchandran, and M. Vetterli. Optimal trellis-based buffered compression and fast approximations. *IEEE Trans IP*, 3:26–40, 1994.

[7] P. Prandoni, M. Goodwin, and M. Vetterli. Optimal time segmentation for signal modeling and compression. In *ICASSP Proc.*, volume 3, pages 2029–2032, Munich, April 1997.

[8] G.D. Forney. The Viterbi algorithm. *Proc. IEEE*, 61:268–278, March 1973.

[9] P. Prandoni and M. Vetterli. A mixed framework arithmetic coder. In *Proc. PCS97*, Berlin, 1997.

[10] Prandoni P. and Vetterli M. Perceptually hidden data transmission over audio signals. In *Proc. ICASSP98*, volume 6, pages 3665–3668, Seattle, USA, May 1998.