WEIGHTED UNIVERSAL BIT ALLOCATION: OPTIMAL MULTIPLE QUANTIZATION MATRIX CODING

M. Effros

Dept. of Electrical Engineering, 116-81
California Institute of Technology
Pasadena, CA 91125

P.A. Chou

Xerox Palo Alto Research Center
3333 Coyote Road
Palo Alto, CA 94304

ABSTRACT

We introduce a two-stage bit allocation algorithm analogous to the algorithm for weighted universal vector quantization (WUVQ) [1, 2]. The encoder uses a collection of possible bit allocations (typically in the form of a collection of quantization matrices) rather than a single bit allocation (or single quantization matrix). We describe both an encoding algorithm for achieving optimal compression using a collection of bit allocations and a technique for designing locally optimal collections of bit allocations. We demonstrate performance on a JPEG style coder using the mean squared error (mse) distortion measure. On a sequence of medical brain scans, the algorithm achieves up to 2.5 dB improvement over a single bit allocation system, up to 5 dB improvement over a WUVQ with first- and second-stage vector dimensions equal to 16 and 4 respectively, and up to 12 dB improvement over an entropy constrained vector quantizer (ECVQ) using 4 dimensional vectors.

1. INTRODUCTION

Typically implemented with the use of quantization matrices, bit allocation is a key step in a wide array of transform codes. Given a sequence of data in the transform domain (e.g., DCT coefficients), a bit allocation algorithm determines how to allocate the available rate among the frequency bands. The optimal bit allocation depends on the overall statistics of the source. For example, images containing large amounts of high frequency information require that more rate be allocated to the high frequency coefficients than do images made up primarily of low frequency information.

An approach common to a wide array of transform codes (including the JPEG algorithm) involves breaking an incoming data sequence into blocks, performing an independent transform on each of those blocks, and then coding the blocks using a single bit allocation (e.g., a single quantization matrix). The performance of this type of scheme can be greatly improved by replacing the single bit allocation by a collection of bit allocations. While a single bit allocation can be designed to do well on average across a particular data sequence, no single bit allocation strategy can track local variation in data statistics. A code that incorporates a variety of bit allocation options and allows the bit allocation to change from image to image or even from data block to data block can better track local source statistics. The expense of tracking this information is the side information necessary to describe the chosen bit allocation.

We here present a technique for designing an optimal collection of bit allocations for a given source or distribution. The design algorithm is functionally equivalent to the WUVQ design strategy [1, 2], which in turn mirrors the generalized Lloyd algorithm. We call our technique weighted universal bit allocation (WUBA).

In Section 2 we describe weighted universal bit allocation and the WUBA design algorithm; in Section 3 we describe a simple transform code, and in Section 4 we use this code to demonstrate the gain of WUBA over a single bit allocation system.

2. THE WUBA ALGORITHM

Let $x^l = (x_1, \ldots, x_l) \in \mathcal{X}^l$ represent transformed data (e.g., DCT coefficients with $l = 64$) that are to be represented by quantizing the ith component to rate b_i, $i = 1, \ldots, l$. Then b^l represents a bit allocation for the quantization system. Suppose that we are given some generic scheme for encoding x^l with bit allocation b^l. Then associated with any b^l is a quantizer $C = \beta \circ \alpha$ with encoder $\alpha : \mathcal{X}^l \to \mathcal{S}$ and decoder $\beta : \mathcal{S} \to \hat{\mathcal{X}}^l$ that together map the input space $\mathcal{X}^l$ of possible data vectors to the output space $\hat{\mathcal{X}}^l$ of reproductions by way of a binary prefix code $\mathcal{S}$. Let $d(x^l, b^l) = d(x^l, \beta(\alpha(x^l)))$ be the total distortion achieved by quantizing x^l with bit allocation b^l and let $r(x^l, b^l) = |\alpha(x^l)|$ denote the associated rate. While $r(x^l, b^l)$ equals $\sum b_i$ on average, $r(x^l, b^l)$ may vary with x^l in a variable-rate system.

We next consider a collection $b_1^l, b_2^l, \ldots, b_M^l$ of bit allocations. Using the quantization interpretation of a two-stage weighted universal code [2], we consider this collection to be a *codebook* of bit allocations. Thus we define a "first-stage quantizer" $\tilde{\beta} \circ \tilde{\alpha}$ with encoder $\tilde{\alpha} : \mathcal{X}^N \to \tilde{\mathcal{S}}$ and decoder $\tilde{\beta} : \tilde{\mathcal{S}} \to \mathcal{B}$ that maps the input space of possible data blocks x^N to the output space $\mathcal{B}$ of possible bit allocations b^l. We here assume that N is a multiple of l. The first-stage encoder chooses for each N-block a single bit allocation. We then use the chosen bit allocation to encode each of the l-vectors in x^N. In JPEG, N equals the size of a single image. For many applications, we may want smaller N to allow the

This material is based upon work partially supported by an AT&T Graduate Scholarship and by a grant from the Center for Telecommunications at Stanford.

0-7803-2431-5/95 $4.00

bit allocation to change within a single image. The example of Section 4 uses $N = l$, which means that we describe a new bit-allocation for each transform block.

The total distortion associated with encoding data block x^N with bit allocation $\tilde{\beta}(\tilde{\alpha}(x^N))$ is

$$d(x^N, \tilde{\beta}(\tilde{\alpha}(x^N))) = \sum_{i=1}^{N/l} d(x_i^l, \tilde{\beta}(\tilde{\alpha}(x^N))).$$

The total rate associated with encoding x^N includes both the rate associated with describing a bit allocation and the rate associated with using the chosen bit allocation. Thus

$$r(x^N, \tilde{\beta}(\tilde{\alpha}(x^N))) = |\tilde{\alpha}(x^N)| + \sum_{i=1}^{N/l} r(x_i^l, \tilde{\beta}(\tilde{\alpha}(x^N))).$$

Then, using a Lagrangian in order to minimize the distortion subject to a constraint on the rate, the optimal first-stage encoder $\tilde{\alpha}^*$ for a given collection of bit allocations $\tilde{\beta}$ is

$$\tilde{\alpha}^* = \arg\min_{s \in \tilde{\mathcal{S}}} [d(x^N, \tilde{\beta}(s)) + \lambda r(x^N, \tilde{\beta}(s))]$$

for every x^N. We call the optimal first-stage encoder a *nearest neighbor* encoder.

Likewise, the optimal first-stage decoder $\tilde{\beta}^*$ for a given first-stage encoder $\tilde{\alpha}$ satisfies

$$\tilde{\beta}^*(s) = \arg\min_{b^l \in \mathcal{B}} E\left[\sum_{i=1}^{N/l} d(X_i^l, b^l) + \lambda r(X_i^l, b^l) \middle| \tilde{\alpha}(X^N) = s\right]$$

for every $s \in \mathcal{S}$. We call the process of designing the optimal first-stage decoder *decoding to the centroid.* This step may be accomplished by any number of optimal bit-allocation design algorithms, one of which is discussed briefly in Section 3.

The WUBA design algorithm employs an iterative descent technique to minimize the expected Lagrangian performance. We initialize the algorithm with an arbitrary prefix code $\tilde{\mathcal{S}}$ and collection $\{\tilde{\beta}(\tilde{s}) : \tilde{s} \in \tilde{\mathcal{S}}\}$ of bit allocations. Each iteration proceeds through three steps which we enumerate below.

1. *Nearest Neighbor Encoding*
   Optimize the first-stage encoder $\tilde{\alpha}$ for the given first-stage decoder $\tilde{\beta}$ and prefix code $\tilde{\mathcal{S}}$.
2. *Decoding to the Centroid*
   Optimize the first-stage decoder $\tilde{\beta}$ for the newly redesigned first-stage encoder and the given first-stage prefix code $\tilde{\mathcal{S}}$.
3. *Optimizing the Prefix Code*
   Optimize the first-stage prefix code $\tilde{\mathcal{S}}$ for the newly redesigned first-stage encoder $\tilde{\alpha}$ and decoder $\tilde{\beta}$. The optimal prefix code s^* for a given first-stage encoder $\tilde{\alpha}$ and decoder $\tilde{\beta}$ is the entropy code matched to the probabilities $P\{\tilde{\alpha}(X^N) = s\}$, for which the ideal codelengths are

$$|s^*| = -\log P\{\tilde{\alpha}(X^N) = s\}.$$

Each step of the algorithm decreases the expected Lagrangian performance. Since the Lagrangian performance cannot be negative, the algorithm is guaranteed to converge. The proposed algorithm therefore guarantees a locally optimal solution.

3. A SIMPLE DCT-BASED SOURCE CODER

We here present a simple DCT-based transform code and the corresponding optimal single bit allocation design strategy that will be used in the next section to test the proposed weighted universal bit allocation algorithm. The transform code is similar to JPEG but simpler for our purposes. Given a two-dimensional array of integer or real input values $[x_{i,j}]$ (typically representing a digital image) we first break the input data sequence into 8×8 blocks. Each block will henceforth be considered independently. Each data block passes first through a two-dimensional DCT transform. The encoder $\alpha : \mathbb{R}^{64} \to \mathcal{S}$ maps the resulting 64-dimensional real data block into a binary string. The encoder accomplishes this mapping in two steps. The encoder first passes the data block through a quantization matrix $[Q_{u,v}]$, where the encoder divides the (u,v)th component $\mathcal{F}_{u,v}$ by the corresponding quantization matrix component $Q_{u,v}$, truncating to obtain the integer $\mathcal{M}_{u,v} = \lfloor \mathcal{F}_{u,v}/Q_{u,v} \rfloor$. This truncation represents the lossy step in the quantization process. The encoder then losslessly describes the resulting quantized sequence to the decoder using a collection $\{\mathcal{S}_{u,v}\}$ of entropy codes. The decoder simply reverses the process, retrieving $[\mathcal{M}_{u,v}]$ from its binary representation, and then scaling back up to achieve a frequency domain reproduction $[\hat{\mathcal{F}}_{u,v}]$, where $\hat{\mathcal{F}}_{u,v} = \mathcal{M}_{u,v} Q_{u,v}$. We find the spatial domain reproduction $\hat{x}_{i,j}$ by taking an inverse DCT of $[\hat{\mathcal{F}}_{u,v}]$.

In this case, as in JPEG, bit allocation is controlled by the quantization matrix $Q_{u,v}$ which determines how coarsely or finely a particular component will be quantized and therefore how much rate will be used. The algorithm differs from the JPEG algorithm in several details, which we now briefly point out and discuss. First, we remove the differential encoding of the DC component so that each source block may be treated independently. Second, we remove the run-length encoder from the lossless coding step, so that the quantizer may code each component within a given frequency block independently. We make these modifications merely for simplicity in the quantization matrix design process, which we next describe.

Given the independence imposed by the above modifications to the JPEG algorithm and assuming an additive distortion measure in the frequency domain, changing the (u,v)th component in the quantization matrix affects only the rate and distortion associated with the (u,v)th component of the data blocks encoded with the given sequence. In designing a quantization matrix for a given sequence of training data, we may therefore consider each quantization matrix component independently. We therefore choose each component $Q_{u,v}$ to minimize the Lagrangian performance over a given training sequence, where larger $Q_{u,v}$ values correspond to higher distortions but lower rates. The optimal entropy code for each component is the entropy code matched to that component's statistics. We therefore achieve a simple design process for optimizing a bit

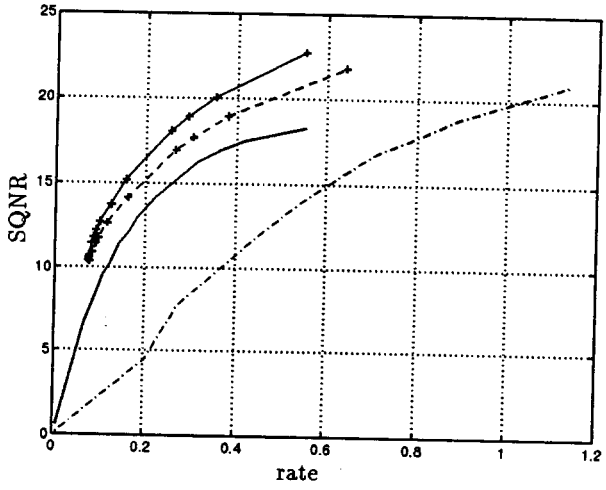

Figure 1: Comparison of SQNR results on a collection of 5 MR brain scans. The systems tested include WUBA (pluses with solid line), a single bit allocator system (pluses with dashed line); variable-rate WUVQ with first-stage vector dimension equal to 16 and second-stage vector dimension equal to 4 (solid line), and ECVQ with vector dimension equal to 4 (broken line).

allocation system for a particular set of training data. We note that the resulting quantization table represents a good starting value from which we can design the quantization and Huffman tables for the JPEG algorithm. A simple variation on the entropy-constrained variation of the generalized Lloyd algorithm can then be used to iteratively update the quantization and Huffman tables for the given training set. We here stick with the earlier described procedure for simplicity.

4. EXPERIMENTAL RESULTS

In Figure 1, we compare the performance of the WUBA algorithm to the performance of single bit allocation and the performances of WUVQ and ECVQ. WUBA contains 64 bit allocations and uses $N = l = 64$. WUVQ uses a first-stage vector dimension of 16. Both WUVQ and ECVQ use a base vector dimension of 4. Each system was trained on 20 medical brain scans and then tested on 5 scans outside of the training set. All rates are reported in terms of entropy. WUBA achieves up to 2.5 dB improvement over single bit allocation systems, up to 5 dB improvement over WUVQ with first- and second-stage vector dimensions equal to 16 and 4 respectively, and up to 12 dB improvement over ECVQ. The performance curves for the WUBA and single bit allocation systems can both be expected to shift slightly to the left if they use a lossless code more efficient than independent entropy coding, such as the JPEG code (run-length followed by Huffman coding) or a zerotree code. Figure 2 shows a sample image.

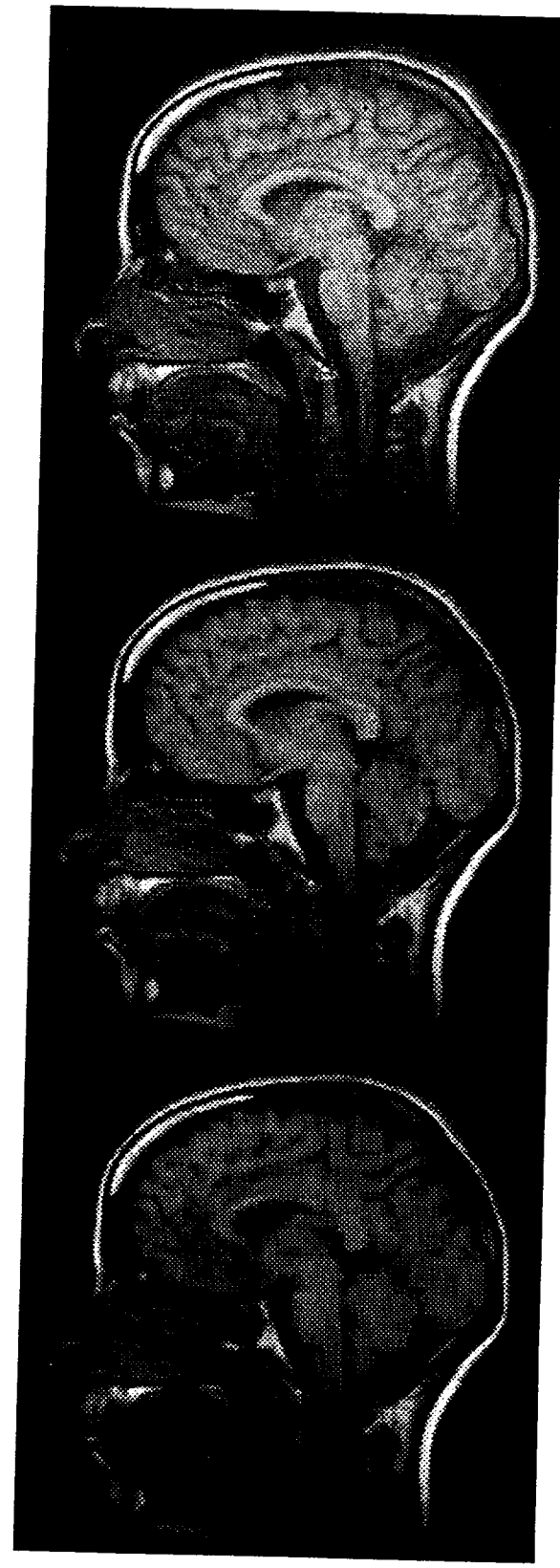

Figure 2: The top picture is the original image. The middle picture is coded to .45 bpp using a single bit allocation. The bottom picture is coded to .35 bpp using a collection of 64 bit allocations.

5. REFERENCES

[1] P. A. Chou. Code clustering for weighted universal VQ and other applications. In *Proceedings of the IEEE International Symposium on Information Theory*, page 253, Budapest, Hungary, June 1991.

[2] P. A. Chou, M. Effros, and R. M. Gray. A vector quantization approach to universal noiseless coding and quantization. *IEEE Transactions on Information Theory.* Submitted.