Transform Coding with Integer-to-Integer Transforms

Vivek K Goyal, *Member, IEEE*

***Abstract*—A new interpretation of transform coding is developed that downplays quantization and emphasizes entropy coding, allowing a comparison of entropy coding methods with different memory requirements. With conventional transform coding, based on computing Karhunen–Loève transform coefficients and then quantizing them, vector entropy coding can be replaced by scalar entropy coding without an increase in rate. Thus the transform coding advantage is a reduction in memory requirements for entropy coding. This paper develops a transform coding technique where the source samples are first scalar-quantized and then transformed with an integer-to-integer approximation to a nonorthogonal linear transform. Among the possible advantages is to reduce the memory requirement further than conventional transform coding by using a single common scalar entropy codebook for all components. The analysis shows that for high-rate coding of a Gaussian source, this reduction in memory requirements comes without any degradation of rate-distortion performance.**

***Index Terms*—Entropy coding, Gaussian sources, memory reduction, transform coding.**

I. Introduction

TRANSFORM coding is the most successful and pervasive technique for lossy compression of audio, images, and video. The conventional framework of transform coding was introduced by Huang and Schultheiss [1][1]: one is given a discrete-time, continuous-valued, vector source with correlated components; instead of quantizing the components separately, one uses a linear transform to compute *transform coefficients* and scalar-quantizes the transform coefficients. In either case, entropy codes may be used to improve the coding efficiency. Transform coding is contrasted to direct quantization in the top and bottom paths of Fig. 1.

Transform coding is an inherently suboptimal source coding technique because it uses the Cartesian product of scalar quantizers, called simply a *scalar quantizer.* Compared to an appropriately designed vector quantizer, a scalar quantizer has a *space-filling loss* [3], [4] that cannot be neutralized by linear pre- and post-processing. This is caused by the high second moment of a cube, as compared to a sphere of the same volume. On the other hand, transform coding has much lower complexity than less constrained forms of vector quantization. Thus transform coding is often called a low-complexity alternative to vector quantization. The transform changes coordinates to improve the performance of scalar quantization.

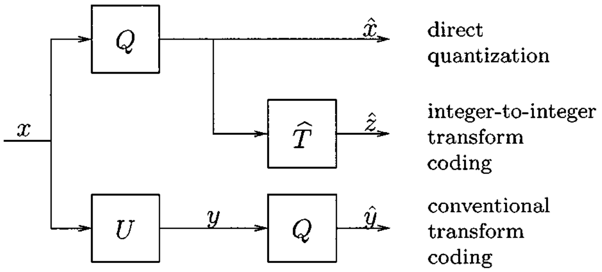

Fig. 1. Schematic representation of a system that produces the three representations considered in this paper.

The discussion above places great emphasis on quantization. This paper shows that this emphasis is undue. At high rates, the performance obtained by Huang and Schultheiss can be matched by a system that quantizes the original source components and then uses a transform to reduce scalar entropy (see the middle path of Fig. 1). This suggests that the value of transform coding comes not from changing the coordinate system for quantization, but almost exclusively from decorrelation.

The most important feature of the proposed system is that the transform is now discrete, not continuous as before. It is shown that the transform can be designed so that the memory requirement of the entropy coding block is reduced without sacrificing rate-distortion performance. Also, since the source is immediately discretized, it is not important that it is continuous-valued. Thus the techniques described here can be applied to discrete-valued sources.

The paper is organized as follows: Section II gives an example of the proposed scheme to provide motivation for the general framework, which is described in Section III. Section IV summarizes the results.

II. An Illustrative Example

Let x be a Gaussian random vector with mean zero and covariance matrix

$$R_x = U^T \mathrm{diag}\left(1,\ \tfrac{1}{8}\right) \mathrm{U}$$

where

$$U = \begin{bmatrix} \cos\pi/6 & \sin\pi/6 \\ -\sin\pi/6 & \cos\pi/6 \end{bmatrix}.$$

We will compare three methods for representing x with approximately the same mean-squared error (MSE) distortion. These methods all include uniform scalar quantization; they differ in their use of transforms, as shown in Fig. 1. Each of the three is then combined with different forms of entropy coding, yielding

Manuscript received March 16, 1999; revised September 24, 1999. The material in this paper was presented in part at the 32nd Asilomar Conference on Signals, Systems, and Computers, Pacific Grove, CA, November 1–4, 1998.

The author is with the Mathematics of Communications Research Department, Bell Laboratories, Lucent Technologies, Murray Hill, NJ 07974-0636 USA.

Communicated by P. A. Chou, Associate Editor for Source Coding.
Publisher Item Identifier S 0018-9448(00)01363-8.

[1] Earlier work by Kramer and Matthews [2] did not include quantization; hence, it is disconnected from the current practice of transform coding.

several overall techniques with different rates and memory requirements.

The first method is to just quantize the components of x. Let $Q_{\Delta,\,n}$ denote a symmetric uniform scalar quantizer with cell width Δ and n cells. Such a quantizer with an odd number of cells has the form

$$Q_{\Delta,\,2N+1}(w) = \begin{cases} -N\Delta, & \text{for } w < \left(-N+\frac{1}{2}\right)\Delta \\ j\Delta, & \text{for } \left(j-\frac{1}{2}\right)\Delta \le w < \left(j+\frac{1}{2}\right)\Delta, \\ & \quad j = -N+1,\, -N+2,\, \cdots,\, N-1 \\ N\Delta, & \text{for } w \ge \left(N-\frac{1}{2}\right)\Delta. \end{cases}$$

We will use the same notation for a product quantizer that acts independently on the components of a vector. In order to create a concrete example, let us choose $n = 9$ cells and quantization step size $\Delta = 2/3$ (in each dimension). The “direct” representation is given by $\hat{x} = Q_{2/3,\,9}(x)$. The per-component distortion in this representation is $\frac{1}{2}E\|x-\hat{x}\|^2 \approx 0.03714$ and the scaled probability density function (p.d.f.) of $\hat{x}$ is shown in Fig. 2(a). It is clear from Fig. 2(a) that $\hat{x}_1$ and $\hat{x}_2$ are not independent. The rate for coding $\hat{x}$ will be considered later, after we form the other two descriptions.

The conventional approach to transform coding this source prescribes the computation of $y = Ux$ followed by the scalar quantization of y. The representation $\hat{y} = Q_{2/3,\,9}(y)$ has virtually the same distortion as the earlier representation: $\frac{1}{2}E\|x - U^{-1}\hat{y}\|^2 \approx 0.03742$.[2] The scaled p.d.f. of $\hat{y}$ is shown in Fig. 2(b); were it not for rounding, it would be clear that $\hat{y}_1$ and $\hat{y}_2$ are independent.

Though both have the same distortion, the representation $\hat{y}$ is better than $\hat{x}$ because its components are independent, and hence they can be efficiently entropy-coded separately. A similar effect can be achieved with a transform applied to $\hat{x}$. Define an integer-to-integer transform[3] $\hat{T}: \Delta\mathbb{Z}^2 \to \Delta\mathbb{Z}^2$ by

$$\hat{T}(\hat{x}) = \left[\begin{bmatrix}1 & 0\\ -0.1454 & 1\end{bmatrix}\left[\begin{bmatrix}1 & 1.2401\\ 0 & 1\end{bmatrix} \cdot \left[\begin{bmatrix}1 & 0\\ -0.9922 & 1\end{bmatrix}\hat{x}\right]_\Delta\right]_\Delta\right]_\Delta \qquad (1)$$

where $[\cdot]_\Delta$ denotes rounding to the nearest multiple of Δ. For the time being, accept this transform without explanation, and apply it to $\hat{x}$ to get a new random vector $\hat{z} = \hat{T}(\hat{x})$. The behavior of $\hat{T}$ is shown in Fig. 3. In this plot, the $(\hat{x}_1/\Delta,\, \hat{x}_2/\Delta)$ position is labeled with $(\hat{z}_1/\Delta,\, \hat{z}_2/\Delta)$. The lines show the inverse image of the axes, i.e., the $(\hat{x}_1,\, \hat{x}_2)$ pairs that map to $(\hat{z}_1,\, 0)$ or $(0,\, \hat{z}_2)$. These are nonorthogonal and approximately straight, suggesting correctly that $\hat{T}$ is an approximation to a nonorthogonal linear transform. Since it is invertible, $\hat{T}$ has no effect on the distortion. The p.d.f. of $\hat{z}$ is shown in Fig. 2(c).

We have constructed three discrete representations of the source—$\hat{x}$, $\hat{y}$, and $\hat{z}$—with the same distortion. We now consider various methods for entropy coding these representations. The lowest rates are obtained with entropy codes that apply to entire vectors, both components together. These rates are given in the first row of Table I.[4] The rates for $\hat{x}$ and $\hat{z}$ are identical because $\hat{T}$ is invertible and is thus merely a relabeling of cells; unlike the continuous-domain transform U, the integer-to-integer transform $\hat{T}$ does not change the quantization cells. The difference in rates between $\hat{x}$ and $\hat{y}$ is small and vanishes as the rate is increased (Δ is decreased with n increased commensurately).

The rates in the first row of Table I show no advantage to the use of a transform. This emphasizes the point made in Section I that the coordinates chosen for quantization are not necessarily important. Remember, however, that the spirit of transform coding is to avoid this sort of joint vector processing. When separate entropy codes are used for each scalar component, the advantage of transform coding emerges. This is shown in the second row of Table I, where each entry is the average of the entropies of the two components. For representation $\hat{y}$, this separate processing of components does not increase the rate because the components are independent. The integer-to-integer transform $\hat{T}$ comes close to matching the performance of the KLT. As shown in Section III-C1, at high rates, the integer-to-integer transform method performs just as well as the KLT.

In such a small example, it may not seem important to use two scalar entropy codes instead of one vector entropy code. Assuming each entropy code is stored as a table, the difference is in the amount of memory needed to store the entropy codes. If each component quantizer has N cells and the source is k-dimensional, a vector entropy code has N^k entries while scalar entropy coding requires k tables with N entries. We may reduce the memory requirements further by using a single scalar entropy code—with N entries, designed for the average of the p.d.f.’s of the components—though this will generally increase the rate (see the third row of Table I). Remarkably, the rate increase for $\hat{z}$ is extremely small, and asymptotically there is no increase in rate, so the integer-to-integer transform allows a k-fold memory reduction in the entropy coder. This main result is shown in Section III-C2.

Analytical computations with the quantizer $Q_{\Delta,\,n}$ are difficult or impossible. The remainder of the paper uses the more convenient model of an unbounded uniform quantizer

$$[w]_\Delta = \lim_{n\to\infty} Q_{\Delta,\,n}(w) = k\Delta \qquad \text{for } \left(k-\tfrac{1}{2}\right)\Delta \le w < \left(k+\tfrac{1}{2}\right)\Delta, \qquad k \in \mathbb{Z}.$$

When this quantizer is used and Δ is small, the distortion is approximately $\Delta^2/12$ and the entropy of a quantized coefficient $[w]_\Delta$ can be estimated from the differential entropy of w [5]

$$H([w]_\Delta) \approx h(w) - \log_2 \Delta \qquad \text{(in bits)}.$$

Subject to these approximations, the best performance obtainable for transform coding the given source is

$$D = \frac{\pi e}{12\sqrt{2}} 2^{-2R}$$

[2] The distortions are nearly equal because Δ was chosen to make the overload distortion small. Both distortions are close to the high-resolution approximation of $D \approx \Delta^2/12 \approx 0.03704$.

[3] “Integer-to-integer” seems the best way to specify that the transforms are on a scaled $\mathbb{Z}$ lattice, despite the scaling by Δ. This term is used throughout.

[4] $H(\cdot)$ is used to denote the entropy of a discrete random variable. Elsewhere $h(\cdot)$ denotes the differential entropy of a continuous random variable.

$\hat{x}_2$ ↑

0.0002	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	
0.0067	0.0090	0.0052	0.0010	0.0001	0.0000	0.0000	0.0000	0.0000	
0.0235	0.1004	0.1687	0.1021	0.0217	0.0016	0.0000	0.0000	0.0000	
0.0105	0.1271	0.5761	0.9626	0.5910	0.1304	0.0100	0.0003	0.0000	
0.0005	0.0185	0.2327	1.0432	1.7137	1.0432	0.2327	0.0185	0.0005	$\to \hat{x}_1$
0.0000	0.0003	0.0100	0.1304	0.5910	0.9626	0.5761	0.1271	0.0105	
0.0000	0.0000	0.0000	0.0016	0.0217	0.1021	0.1687	0.1004	0.0235	
0.0000	0.0000	0.0000	0.0000	0.0001	0.0010	0.0052	0.0090	0.0067	
0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0001	0.0002	

(a) Quantized source $\hat{x}$.

$\hat{y}_2$ ↑

0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	
0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	
0.0002	0.0009	0.0026	0.0049	0.0061	0.0049	0.0026	0.0009	0.0002	
0.0167	0.0648	0.1891	0.3595	0.4453	0.3595	0.1891	0.0648	0.0167	
0.0640	0.2484	0.7253	1.3790	1.7083	1.3790	0.7253	0.2484	0.0640	$\to \hat{y}_1$
0.0167	0.0648	0.1891	0.3595	0.4453	0.3595	0.1891	0.0648	0.0167	
0.0002	0.0009	0.0026	0.0049	0.0061	0.0049	0.0026	0.0009	0.0002	
0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	
0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	

(b) Quantized KLT coefficients $\hat{y}$.

$\hat{z}_2$ ↑

0.0000												
0.0000	0.0000		0.0000									
	0.0000	0.0000	0.0000	0.0000								
	0.0000	0.0000	0.0000	0.0003	0.0005							
	0.0000	0.0000	0.0016	0.0100	0.0185	0.0105						
	0.0000	0.0001	0.0010	0.0217	0.1304	0.2327	0.1271	0.0235				
		0.0000	0.0052	0.1021	0.5910	1.0432	0.5761	0.1004	0.0067	0.0002		
		0.0001	0.0090	0.1687	0.9626	1.7137	0.9626	0.1687	0.0090	0.0001		
		0.0002	0.0067	0.1004	0.5761	1.0432	0.5910	0.1021	0.0052	0.0000		
				0.0235	0.1271	0.2327	0.1304	0.0217	0.0010	0.0001	0.0000	
						0.0105	0.0185	0.0100	0.0016	0.0000	0.0000	
							0.0005	0.0003	0.0000	0.0000	0.0000	
								0.0000	0.0000	0.0000	0.0000	
									0.0000		0.0000	0.0000
												0.0000

(c) Integer-to-integer transform coefficients $\hat{z}$.

Fig. 2. Probability densities of $\hat{x}$, $\hat{y}$, and $\hat{z}$, multiplied by 10. Due to rounding, an entry of "0.0000" indicates a nonzero cell probability under $5 \cdot 10^{-6}$. In (c), unmarked positions reflect impossible values of $\hat{z}$.

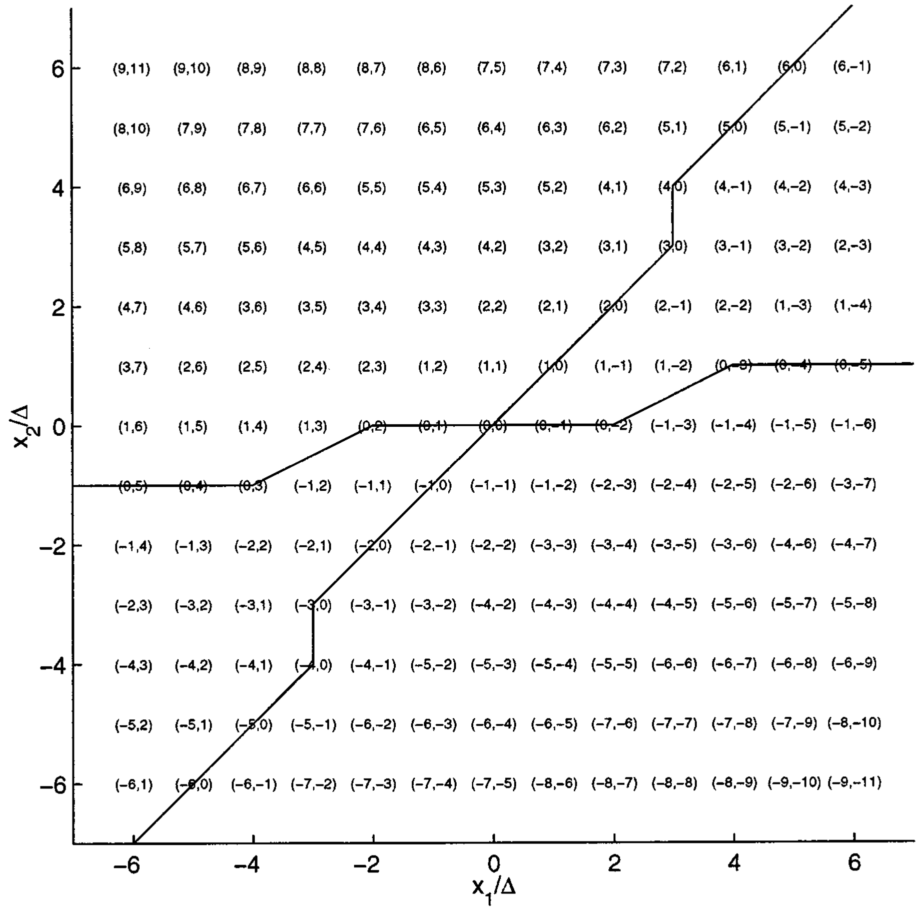

Fig. 3. Depiction of the integer-to-integer transform $\hat{T}$ given by (1). The $(\hat{x}_1/\Delta,\, \hat{x}_2/\Delta)$ position is labeled with $\hat{T}(\hat{x}_1/\Delta,\, \hat{x}_2/\Delta)$.

TABLE I
COMPARISON OF RATES, IN BITS PER VECTOR, WITH THREE REPRESENTATIONS AND THREE ENTROPY CODING METHODS. AT HIGH RATES, THE STARRED ENTRIES COALESCE TO A COMMON OPTIMAL VALUE

Performance with $n = 9$, $\Delta = 2/3$	Representation		
Entropy measure	$\hat{x}$	$\hat{y}$	$\hat{z}$
$\frac{1}{2}H(\cdot)$ (vector entropy code)	1.987*	1.979*	1.987*
$\frac{1}{2}[H(\cdot_1) + H(\cdot_2)]$ (two scalar entropy codes)	2.210	1.979*	1.992*
One common scalar entropy code for both components	2.257	2.189	1.993*

TABLE II
HIGH-RATE PERFORMANCE COMPARISON OF THREE REPRESENTATIONS AND THREE ENTROPY CODING METHODS. STARS REPRESENT OPTIMAL PERFORMANCE AND THE OTHER ENTRIES ARE GAPS FROM OPTIMAL PERFORMANCE IN BITS PER COMPONENT

Gap from optimal performance (bits per vector)	Representation		
Entropy coding method	$\hat{x}$	$\hat{y}$	$\hat{z}$
$H(\cdot)$ (vector entropy code)	$\star$	$\star$	$\star$
$H(\cdot_1) + H(\cdot_2)$ (two scalar entropy codes)	0.276	$\star$	$\star$
One common scalar entropy code for both components	0.330	0.249	$\star$

where the rate R is in bits per component. Table II gives the gap from optimal performance, in bits per component, for the three representations and three entropy-coding methods. Only the representation formed with the integer-to-integer transform allows optimal performance with the simplest entropy coding. Similar but distinct examples appear in [6].

III. A NEW FRAMEWORK FOR TRANSFORM CODING

This section develops a new framework for transform coding that is based on using integer-to-integer transforms, as in the middle path of Fig. 1. The example from the previous section provides a preview of the results. Specifically, we will show that at high rates the representation output from an appropriately designed integer-to-integer transform is optimal for vector entropy coding, scalar entropy coding with separate codebooks for each component, and scalar entropy coding with a single codebook for all the components. In other words, the starred entries in Table II will be justified.

Let x be a jointly Gaussian random vector taking a value in $\mathbb{R}^k$. Assume x has mean zero and denote the covariance matrix by K. Since K is a covariance matrix, it is symmetric, positive semidefinite, and has an orthogonal eigendecomposition $K = U^T \Lambda U$, where U is orthogonal and $\Lambda = \operatorname{diag}(\lambda_1, \lambda_2, \cdots \lambda_k)$ with $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_k$. We analyze three methods for representing x:

- Direct quantization: $\hat{x} = [x]_\Delta$.
- Karhunen–Loève transform coding (KLTC): $y = Ux$, $\hat{y} = [y]_\Delta$.
- Integer-to-integer transform coding: $\hat{z} = \hat{T}([x]_\Delta)$.

Given a discrete random vector $\hat{w}$ (which is one of $\hat{x}$, $\hat{y}$, and $\hat{z}$) we consider three ways to assign an associated average rate per component:

- Vector entropy coding: $R_v(\hat{w}) = k^{-1}H(\hat{w})$.
- Scalar entropy coding: $R_s(\hat{w}) = k^{-1}\sum_{i=1}^{k} H(\hat{w}_i)$.
- Entropy coding based on the average of the component densities: $R_a(\hat{w}) = H(\overline{p})$, where $\overline{p}$ is the mean of the p.d.f.'s of the $\hat{w}_i$'s. This is the minimum rate when using one common scalar codebook for all the scalar components (see Theorem 2, Section III-C2).

The bulk of the paper addresses the design of $\hat{T}$ to minimize the second and third rate measures. When these are minimized, the performance of integer-to-integer transform coding matches that of KLTC. But first, we show that direct quantization with vector entropy coding, KLTC with vector entropy coding, and KLTC with scalar entropy coding give asymptotically equal, optimal performance.

A. Preliminary Equivalences

It is well known that in conventional transform coding of a Gaussian source, as in the bottom path of Fig. 1 with scalar entropy coding, the use of a Karhunen–Loève transform is optimal.[5] For the given source, U is a KLT, and $y = Ux$ has independent components. It follows immediately from a basic property of entropy that

$$R_v(\hat{y}) = k^{-1}H(\hat{y}_1, \hat{y}_2, \cdots, \hat{y}_k) = k^{-1}\sum_{i=1}^{k} H(\hat{y}_i) = R_s(\hat{y}). \tag{2}$$

This shows that scalar entropy coding of KLT coefficients performs as well as vector entropy coding of these coefficients. Since the latter is more computationally difficult, it should not be used. This equivalence depends only on the KLT producing independent coefficients, as is the case for a Gaussian source.

Now we consider the asymptotic equality of $R_v(\hat{x})$ and $R_v(\hat{y})$. This is a clear consequence of the following lemmas.

Lemma 1 [5]: Let x be a random vector and let A be a compatibly-dimensioned square matrix. Then

$$h(Ax) = h(x) + \log|\det A|.$$

Lemma 2: Let x be a k-dimensional random vector and let $\hat{x} = [x]_\Delta$ be its uniformly scalar quantized version. If the p.d.f. of x is Riemann-integrable

$$H(\hat{x}) + k\log\Delta \to h(x) \text{ as } \Delta \to 0.$$

Proof: This is a straightforward k-dimensional generalization of [5, Theorem 9.3.1]. □

Since x and y are related by an orthogonal transform, the first lemma implies that x and y have the same differential entropy. Then two applications of the second lemma gives

$$\begin{aligned} R_v(\hat{x}) &= k^{-1}H(\hat{x}) \approx k^{-1}(h(x) - k\log\Delta) \\ &= k^{-1}(h(y) - k\log\Delta) \approx k^{-1}H(\hat{y}) = R_v(\hat{y}). \end{aligned} \tag{3}$$

So far, we have shown that there are three ways to get optimal performance, but we have not described optimal performance. This is easy to compute using Lemma 2. The differential entropy of a Gaussian random variable with variance σ^2 is

$$h(\mathcal{N}(0, \sigma^2)) = \frac{1}{2}\log_2 2\pi e\sigma^2 \text{ bits.}$$

Therefore,

$$\begin{aligned} R_s(\hat{y}) &= k^{-1}\sum_{i=1}^{k} H(\hat{y}_i) \approx k^{-1}\sum_{i=1}^{k}\left(\frac{1}{2}\log_2 2\pi e\lambda_i - \log_2\Delta\right) \\ &= \frac{1}{2}\log_2 \Delta^{-2}2\pi e\left(\prod_{i=1}^{k}\lambda_i\right)^{1/k}. \end{aligned} \tag{4}$$

Recalling that x_i is Gaussian with variance K_{ii}

$$R_s(\hat{x}) = k^{-1}\sum_{i=1}^{k} H(\hat{x}_i) \approx \frac{1}{2}\log_2 \Delta^{-2}2\pi e\left(\prod_{i=1}^{k}K_{ii}\right)^{1/k}.$$

Since the product of diagonal elements of a positive semidefinite matrix is lower bounded by the product of eigenvalues [9, Theorem 7.8.1], $R_s(\hat{x}) \geq R_s(\hat{y})$. Either rate can be written in terms of the distortion per component by substituting $D \approx \Delta^2/12$, e.g.,

$$R_s(\hat{y}) \approx \frac{1}{2}\log_2 \frac{\pi e\left(\prod_{i=1}^{k}\lambda_i\right)^{1/k}}{6D}. \tag{5}$$

B. A Class of Invertible Integer-to-Integer Transforms

We now turn to the main topic of this paper: the performance of integer-to-integer transform coding. The scalar quantized representation $\hat{x}$ lies in the lattice $\Delta\mathbb{Z}^k$, so we consider transforms $\hat{T}$: $\Delta\mathbb{Z}^k \to \Delta\mathbb{Z}^k$ to generate the representation $\hat{z}$. The set of such transforms is not only uncountable, but requires an

[5] This is shown under the assumption of Lloyd–Max quantization and fairly general bit allocation in [1]; a derivation that relies on high-resolution quantization theory and optimal bit allocation is given in [7]; a simple proof that relies on neither optimal quantization nor high-resolution approximations appears in [8].

infinite number of parameters to describe; therefore, we restrict our attention to a certain quasilinear set. These are easy to both describe and compute.

Let $T \in \mathbb{R}^{k \times k}$ have determinant 1. The transforms that we allow are approximations of T that are one-to-one and onto $\Delta\mathbb{Z}^k$. The integer-to-integer transform $\hat{T}$ may be constructed from T in a variety of ways [10]–[12]. All that is important for the ensuing analysis is that $\hat{T}$ is invertible and that the error between $\hat{T}(\hat{x})$ and $T\hat{x}$ vanishes for small Δ.

One simple way to define $\hat{T}$ is through a factorization of matrices, now known as a lifting construction [12]. If $k = 2$, any T with determinant 1 can be factored into three lower- and upper-triangular matrices with unit diagonals as follows:[6]

$$\begin{aligned} T &= \begin{bmatrix} a & b \\ b^{-1}(ad-1) & d \end{bmatrix} \\ &= \underbrace{\begin{bmatrix} 1 & 0 \\ b^{-1}(d-1) & 1 \end{bmatrix}}_{T_1} \underbrace{\begin{bmatrix} 1 & b \\ 0 & 1 \end{bmatrix}}_{T_2} \underbrace{\begin{bmatrix} 1 & 0 \\ b^{-1}(a-1) & 1 \end{bmatrix}}_{T_3}. \end{aligned} \tag{6}$$

Now

$$\hat{T}(\hat{x}) = [T_1[T_2[T_3\hat{x}]_\Delta]_\Delta]_\Delta \tag{7}$$

has the desired properties. When $\hat{x}$ is in the lattice $\Delta\mathbb{Z}^2$, a single stage

$$\left[\begin{bmatrix} 1 & \alpha \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \hat{x}_1 \\ \hat{x}_2 \end{bmatrix}\right]_\Delta = \left[\begin{bmatrix} \hat{x}_1 + \alpha\hat{x}_2 \\ \hat{x}_2 \end{bmatrix}\right]_\Delta = \begin{bmatrix} \hat{x}_1 + [\alpha\hat{x}_2]_\Delta \\ \hat{x}_2 \end{bmatrix}$$

is inverted by a change of sign

$$\begin{aligned} \left[\begin{bmatrix} 1 & -\alpha \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \hat{x}_1 + [\alpha\hat{x}_2]_\Delta \\ \hat{x}_2 \end{bmatrix}\right]_\Delta &= \left[\begin{bmatrix} \hat{x}_1 + [\alpha\hat{x}_2]_\Delta - \alpha\hat{x}_2 \\ \hat{x}_2 \end{bmatrix}\right]_\Delta \\ &= \begin{bmatrix} \hat{x}_1 \\ \hat{x}_2 \end{bmatrix}. \end{aligned}$$

Thus it is easily verified that $\hat{T}$ is invertible on $\Delta\mathbb{Z}^2$ with the inverse

$$\hat{T}^{-1}(\hat{z}) = [T_3^{-1}[T_2^{-1}[T_1^{-1}\hat{z}]_\Delta]_\Delta]_\Delta.$$

It is also easy to bound $||\hat{T}(\hat{x}) - T\hat{x}||$. For $\hat{x} \in \Delta\mathbb{Z}^2$, the computation (7) involves three rounding operations. Using δ_i's to denote the roundoff errors gives

$$\hat{T}(\hat{x}) = T_1\left(T_2\left(T_3\hat{x} + \begin{bmatrix} 0 \\ \delta_1 \end{bmatrix}\right) + \begin{bmatrix} \delta_2 \\ 0 \end{bmatrix}\right) + \begin{bmatrix} 0 \\ \delta_3 \end{bmatrix}.$$

Expanding and using $T_1T_2T_3 = T$, one can compute

$$\begin{aligned} ||\hat{T}(\hat{x}) - T\hat{x}||_\infty &= \left\| T_1T_2 \begin{bmatrix} 0 \\ \delta_1 \end{bmatrix} + T_1 \begin{bmatrix} \delta_2 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ \delta_3 \end{bmatrix} \right\|_\infty \\ &= \left\| \begin{bmatrix} b\delta_1 \\ d\delta_1 \end{bmatrix} + \begin{bmatrix} \delta_2 \\ b^{-1}(d-1)\delta_2 \end{bmatrix} + \begin{bmatrix} 0 \\ \delta_3 \end{bmatrix} \right\|_\infty \\ &\le (1 + \max\{|b|, |d| + |b^{-1}(d-1)|\})\frac{\Delta}{2}. \end{aligned}$$

This shows that $\hat{T}$ approximates T in a precise sense.

For $k \times k$ matrices, the process is similar. T is factored (nonuniquely) into a product of matrices with unit diagonals and nonzero off-diagonal elements only in one row or column: $T = T_1T_2 \cdots T_\ell$. The integer-to-integer version of the transform is then given by

$$\hat{T}(\hat{x}) = [T_1[T_2 \cdots [T_\ell\hat{x}]_\Delta]_\Delta]_\Delta.$$

The inverse of $\hat{T}$ can be implemented by reversing the calculations as before.

C. Transform Coding with Integer-to-Integer Transforms

We have defined a set from which to select a transform $\hat{T}$ for use as in Fig. 1. Now we consider the selection of $\hat{T}$ to optimize the performance of the system. Recall that $\hat{T}$ is invertible; thus it does not introduce any distortion and its design can be based solely on the minimization of rate for given Δ. Also, since $\hat{T}$ approximates the linear transform T from which it is derived, the design process may be simplified by consideration of T in place of $\hat{T}$.

1) Rate Minimization: Since $\hat{x}$ and $\hat{z}$ are related by an invertible transform, they have the same entropy. Thus irrespective of the choice of $\hat{T}$ (using (2)–(4))

$$\begin{aligned} R_v(\hat{z}) &= R_v(\hat{x}) \approx R_v(\hat{y}) \\ &= R_s(\hat{y}) \approx \tfrac{1}{2}\log_2 \Delta^{-2}2\pi e \left(\prod_{i=1}^{k} \lambda_i\right)^{1/k}. \end{aligned}$$

In the more interesting cases, those involving scalar entropy coding, the rate depends on the transform.

Let us first consider the estimation and minimization of $R_s(\hat{z})$. The discrete variable $\hat{z}$ is approximately equal to a uniform scalar quantized version of $z = Tx$. Then since z_i is Gaussian with variance V_{ii}, where $V = TKT^T$, the entropy of $\hat{z}_i$ is given approximately by

$$H(\hat{z}_i) \approx \frac{1}{2}\log_2 \Delta^{-2}2\pi e V_{ii}.$$

Averaging the contributions from the components of $\hat{z}$ gives

$$R_s(\hat{z}) = k^{-1}\sum_{i=1}^{k} H(\hat{z}_i) \approx \frac{1}{2}\log_2 \Delta^{-2}2\pi e \left(\prod_{i=1}^{k} V_{ii}\right)^{1/k}.$$

Thus the design problem is identical to that in conventional transform coding—to find T to minimize the product of the diagonal of TKT^T—but now over all transforms with determinant 1, instead of only over orthogonal transforms. Two questions naturally arise: Does the additional freedom in choosing T change the result of the minimization? And, why is there additional freedom?

The first question is answered negatively by the following theorem.

Theorem 1: Let $K \in \mathbb{R}^{k \times k}$ be symmetric and positive semidefinite. Then

$$\min_{T:\ \det T = 1} \prod \operatorname{diag}(TKT^T) = \prod_{i=1}^{k} \lambda_i \tag{8}$$

where $\{\lambda_1, \lambda_2, \cdots, \lambda_k\}$ is the set of eigenvalues of K.

[6] If $b = 0$, a similar factorization exists with two upper-triangular factors and one lower-triangular factor. The factorization is generally not unique.

Proof: Since K is symmetric and positive semidefinite, there is an orthogonal matrix U such that

$$UKU^T = \Lambda = \operatorname{diag}(\lambda_1, \lambda_2, \cdots, \lambda_k).$$

Let $V = TKT^T$. Then V is positive semidefinite and

$$\begin{aligned}\prod_{i=1}^{k} V_{ii} &\geq \det V = \det TU^T \Lambda U T^T \\ &= (\det TU^T)(\det \Lambda)(\det UT^T) = \prod_{i=1}^{k} \lambda_i\end{aligned}$$

where the inequality is Hadamard's inequality [9, Theorem 7.8.1]. If K is positive definite, equality holds if and only if V is diagonal. The minimum is (nonuniquely) achieved by $T = U$. □

Theorem 1 shows that one possible optimal choice for T is a KLT. This yields

$$R_s(\hat{z}) = k^{-1} \sum_{i=1}^{k} H(\hat{z}_i) \approx \frac{1}{2} \log_2 \Delta^{-2} 2\pi e \left(\prod_{i=1}^{k} \lambda_i \right)^{1/k}$$

which is equal to optimal rates $R_v(\hat{x})$, $R_v(\hat{y})$, $R_v(\hat{z})$, and $R_s(\hat{y})$ computed earlier. Matching the optimal performance without having to compute a transform on the original, high-resolution data may be useful in its own right. However, the more interesting fact is that optimal performance is achieved with many unused degrees of freedom.

Suppose the eigenvalues of K are distinct and positive. Then the *orthogonal* transforms solving the minimization (8) are unique up to multiplications of rows by -1 and permutations of rows. In particular, this means that not only is the product of the diagonal of TKT^T fixed, but also that the diagonal elements are exactly the eigenvalues of K. When minimizing over transforms with determinant 1, including nonorthogonal transforms, the resulting TKT^T is diagonal with any combination of nonnegative entries with product $\prod_{i=1}^{k} \lambda_i$. This property is demonstrated and exploited in Section III-C2.

The additional freedom in the design of the integer-to-integer transform over that in orthogonal transform coding is useful when a single entropy codebook is used for all of the scalar components of $\hat{z}$. To ascertain the source of this design freedom, we should ask why orthogonal transforms are used in conventional transform coding. The algebraic approach of [1], while precise, does not reveal much; instead, the question is best answered with reference to the geometry of quantization. If T is not orthogonal, encoding according to $\hat{y} = [Tx]_\Delta$ induces a noncubic partitioning of $\mathbb{R}^k$, i.e., the set $\{x : [Tx]_\Delta = c\}$ is not a hypercube. Such a cell has a higher second moment about its center—hence, higher distortion—than a hypercubic cell with the same volume. Orthogonality is not important in integer-to-integer transform coding because the partitioning is fixed, independent of the transform.

2) Codebook Size Minimization: We have seen that there are several ways to achieve the rate-distortion performance described by (5). The motivation for using either form of transform coding is to reduce the memory requirements in the entropy coder. As mentioned in Section II, the number of codewords in the entropy coder can be reduced from N^k to kN, where N is the number of quantization cells per dimension.[7] The memory requirement can be reduced further from kN to N if a single scalar entropy code is used for each of the components.

Suppose a common entropy code is applied to the discrete random variables $\hat{w}_1, \hat{w}_2, \cdots, \hat{w}_k$. The expected code length then depends not only on the entropies of the random variables, but also on the disparity in their densities. Denote the density of $\hat{w}_i$ by p_i and let q be the density corresponding to the code assignment. Then the expected code length for $\hat{w}_i$ is

$$E\ell(\hat{w}_i) = H(\hat{w}_i) + D(p_i \| q)$$

where $D(p_i \| q)$ is the relative entropy between p_i and q. Averaging over i gives the average expected code length per component as

$$R_a(\hat{w}) = k^{-1} \sum_{i=1}^{k} E\ell(\hat{w}_i) = R_s(\hat{w}) + k^{-1} \sum_{i=1}^{k} D(p_i \| q).$$

Now, what choice of q minimizes $R_a(\hat{w})$? The following theorem establishes that, as asserted earlier, the single scalar entropy code should be designed to match the mean of the p.d.f.'s $\bar{p} = k^{-1} \sum_{i=1}^{k} p_i$.

Theorem 2: The sum of relative entropies $\sum_{i=1}^{k} D(p_i \| q)$ is minimized by $q = k^{-1} \sum_{i=1}^{k} p_i$.

Proof: Writing out the sum, noting that q is a p.d.f. on the alphabet $\mathbb{Z}$, gives the following problem:

$$\text{Minimize} \sum_{i=1}^{k} \sum_{j \in \mathbb{Z}} p_i(j) \log \frac{p_i(j)}{q(j)} \text{ subject to} \sum_{j \in \mathbb{Z}} q(j) = 1.$$

Let

$$J = \sum_{i=1}^{k} \sum_{j \in \mathbb{Z}} p_i(j) \log \frac{p_i(j)}{q(j)} + \lambda \sum_{j \in \mathbb{Z}} q(j).$$

Taking a Lagrangian approach, the minimization is solved by the system of equations

$$\begin{aligned}\frac{\partial J}{\partial q(j)} &= 0, \qquad \text{for all } j \in \mathbb{Z} \\ \sum_{j \in \mathbb{Z}} q(j) &= 1.\end{aligned}$$

Now since

$$\frac{\partial J}{\partial q(j)} = \sum_{i=1}^{k} \frac{p_i(j)}{q(j)} + \lambda$$

the solution is

$$q(j) = -\lambda^{-1} \sum_{i=1}^{k} p_i(j)$$

with normalization $\lambda = -k$. □

[7] The codewords themselves would be shorter as well.

TABLE III
HIGH-RATE RATE-DISTORTION PERFORMANCE OF THREE REPRESENTATIONS AND THREE ENTROPY CODING METHODS. THE SOURCE IS GAUSSIAN WITH COVARIANCE MATRIX K, THE EIGENVALUES OF K ARE $\{\lambda_i\}_{i=1}^k$, AND f_{σ^2} DENOTES THE p.d.f. OF A ZERO-MEAN GAUSSIAN RANDOM VARIABLE WITH VARIANCE σ^2

		Rate–distortion performance		
Entropy coding method	Memory (codewords)	No transform	Conventional	Integer-to-integer
Vector entropy code	N^k	r_{opt}	r_{opt}	r_{opt}
k scalar entropy codes	kN	r_1	r_{opt}	r_{opt}
One scalar entropy code	N	r_2	r_3	r_{opt}

$r_{\text{opt}} = \frac{1}{2}\log_2 \frac{\pi e}{6D}(\prod_{i=1}^k \lambda_i)^{1/k}$

$r_1 = \frac{1}{2}\log_2 \frac{\pi e}{6D}(\prod_{i=1}^k K_{ii})^{1/k} = r_{\text{opt}} + \frac{1}{2}\log_2(\prod_{i=1}^k K_{ii}/\lambda_i)^{1/k}$

$r_2 = r_{\text{opt}} + k^{-1}\sum_{i=1}^k D(f_{K_{ii}} \,\|\, \bar{p})$, where $\bar{p} = k^{-1}\sum_{i=1}^k f_{K_{ii}}$

$r_3 = r_{\text{opt}} + k^{-1}\sum_{i=1}^k D(f_{\lambda_i} \,\|\, \bar{p})$, where $\bar{p} = k^{-1}\sum_{i=1}^k f_{\lambda_i}$

Applying the optimal code from Theorem 2, the rate with a single scalar entropy code is

$$R_a(\hat{w}) = R_s(\hat{w}) + k^{-1}\sum_{i=1}^{k} D(p_i \| \overline{p}). \tag{9}$$

Since $D(p_i||,\overline{p})$ is nonnegative and is zero if and only if $p_i = \overline{p}$, $R_a(\hat{w}) \geq R_s(\hat{w})$ with equality if and only if $p_1 = p_2 = \cdots = p_k = \overline{p}$. Integer-to-integer transform coding allows us to achieve this equality.

How is $R_a(\hat{z})$ minimized? First, we need to choose T optimally with respect to R_s. Furthermore, the components of $\hat{z}$ should have identical densities. This is equivalent to requiring

$$TKT^T = \left(\prod_{i=1}^{k} \lambda_i\right)^{1/k} I_k.$$

Such a T exists and can be constructed as follows.

Let $T_0 = U$. Then $T_0 K T_0^T = \Lambda$. Now for notational convenience, let us assign a name to the desired component variance: $\gamma = (\prod_{i=1}^k \lambda_i)^{1/k}$. Notice that

$$\tilde{T}_1 \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \tilde{T}_1^T = \begin{bmatrix} \gamma & 0 \\ 0 & \lambda_1\lambda_2/\gamma \end{bmatrix}$$

where

$$\tilde{T}_1 = \begin{bmatrix} \sqrt{\dfrac{\gamma}{2\lambda_1}} & \sqrt{\dfrac{\gamma}{2\lambda_2}} \\ -\sqrt{\dfrac{\lambda_2}{2\gamma}} & \sqrt{\dfrac{\lambda_1}{2\gamma}} \end{bmatrix}.$$

Defining

$$T_1 = \begin{bmatrix} \tilde{T}_1 & 0 \\ 0 & I_{k-2} \end{bmatrix}$$

$T_1 T_0 K T_0^T T_1^T$ is diagonal with γ in the first diagonal position. It is clear how to proceed: For $i = 2, 3, \cdots, k-1$, let

$$\tilde{T}_i = \begin{bmatrix} \sqrt{\dfrac{\gamma^i}{2\prod_{j=1}^i \lambda_j}} & \sqrt{\dfrac{\gamma}{2\lambda_{i+1}}} \\ -\sqrt{\dfrac{\lambda_{i+1}}{2\gamma}} & \sqrt{\dfrac{\prod_{j=1}^i \lambda_j}{2\gamma^i}} \end{bmatrix}$$

and

$$T_i = \begin{bmatrix} I_{i-1} & 0 & 0 \\ 0 & \tilde{T}_i & 0 \\ 0 & 0 & I_{k-i-1} \end{bmatrix}.$$

Then $T = T_{k-1} T_{k-2} \cdots T_0$ has the desired property.

IV. CONCLUSIONS

Compression of a Gaussian source with all quantization limited to scalars does not allow performance approaching the rate-distortion bound because of the inherent limitation of scalar quantization. Nevertheless, most current source coding systems use scalar quantization in conjunction with a transform.

Conventional transform coding, with an orthogonal linear transform preceding quantization, can be cast as a technique for simplifying entropy coding while maintaining the best rate-distortion performance possible without multidimensional quantization. For a source of dimension k and quantization with N cells per dimension, a single codebook with N^k elements can be replaced by k codebooks with N elements without a loss in performance. Using high-rate approximations, this paper has shown that a system that swaps the transform and quantization—placing quantization before an integer-to-integer transform—can simplify the entropy coding further without a loss in performance; only a single N-element codebook is needed. The results are summarized in Table III. The memory reduction shown in Table III is possible because of greater freedom in the design of an integer-to-integer transform than of an orthogonal linear transform. This greater freedom can be put to advantage in similar ways if mixtures of scalar and vector entropy coding are used.

This paper has considered only the compression of Gaussian sources and some of the statements depend intimately on this choice. In particular, uncorrelated jointly Gaussian variables are independent although this is not true of all random variables. The design of transforms for integer-to-integer transform coding is nothing more than the manipulation of discrete probability densities. This was simplified greatly by continuous approximations, but it should be possible to design an appropriate transform for any source.

ACKNOWLEDGMENT

This work grew out of related work with J. Kovačević and M. Vetterli. Thanks to both of them and to J. Mazo and anonymous reviewers for their valuable comments.

REFERENCES

[1] J. J. Y. Huang and P. M. Schultheiss, “Block quantization of correlated Gaussian random variables,” *IEEE Trans. Commun. Syst.*, vol. COM-11, pp. 289–296, Sept. 1963.

[2] H. P. Kramer and M. V. Mathews, “A linear coding for transmitting a set of correlated signals,” *IRE Trans. Inform. Theory*, vol. IT-23, pp. 41–46, Sept. 1956.

[3] T. D. Lookabaugh and R. M. Gray, “High resolution quantization theory and the vector quantization advantage,” *IEEE Trans. Inform. Theory*, vol. 35, pp. 1020–1033, Sept. 1989.

[4] S. Na and D. L. Neuhoff, “Bennett’s integral for vector quantizers,” *IEEE Trans. Inform. Theory*, vol. 41, pp. 886–900, July 1995.

[5] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. New York: Wiley, 1991.

[6] V. K Goyal and M. Vetterli, “Manipulating rates, complexity, and error-resilience with discrete transforms,” in *32nd Asilomar Conf. Signals, Systems and Computers*, vol. 1, Pacific Grove, CA, Nov. 1998, pp. 457–461.

[7] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression*. Boston, MA: Kluwer, 1992.

[8] V. K Goyal, J. Zhuang, and M. Vetterli, “Transform coding with backward adaptive updates,” *IEEE Trans. Inform. Theory*, to be published.

[9] R. A. Horn and C. R. Johnson, *Matrix Analysis*. Cambridge, U.K.: Cambridge Univ. Press, 1985.

[10] J. Hong, “Discrete Fourier, Hartley, and Cosine transforms in signal processing,” Ph.D. dissertation, Columbia Univ., New York, 1993.

[11] A. Zandi, J. D. Allen, E. L. Schwartz, and M. Boliek, “CREW: Compression with reversible embedded wavelets,” in *Proc. IEEE Data Compression Conf.*, Snowbird, UT, Mar. 1995, pp. 212–221.

[12] A. R. Calderbank, I. Daubechies, W. Sweldens, and B.-L. Yeo, “Wavelet transforms that map integers to integers,” *Appl. Comput. Harmonic Anal.*, vol. 5, no. 3, pp. 332–369, July 1998.