Separation Architectures for Lossy Compression

Ying-zong Huang* and Gregory W. Wornell
Massachusetts Institute of Technology
Dept. of Electrical Engineering and Computer Science
Cambridge, MA 02139
{zong,gww}@mit.edu

***Abstract*—High-performance Model-Code Separation (MCS) architectures for lossless compression are practically viable with graphical message-passing in the decoder. This paper extends separation architectures to lossy compression by constructing model-free but semantics-aware encoders and contributes a new inference-friendly low-density hashing quantizer (LDHQ) to support decoding.**

***Index Terms*—lossy compression, rate distortion, quantization, graphical inference**

I. Introduction

In lossless compression, two distinct architectural styles for system design are possible. Shannon's classical proof of achievability based on random codebook generation leads to schemes in which a *data model* is bound into the coding structure itself, typically in the encoder. This design, a *Joint Model-Code (JMC)* architecture, is prevalent in nearly all existing compression systems. On the other hand, Slepian-Wolf coding based on random binning leads to schemes in which the data model is treated as informationally and structurally separate from the code. In [1] we proposed a practical realization of this *Model-Code Separation (MCS)* architecture (Fig. 1), based on a modular, message-passing decoder. MCS has considerable advantage over traditional JMC in terms of systemic flexibility. Certain applications like encrypted compression require the data model to be concealed from the encoder, and more generally we would like data models to be freely upgradeable. Separation that allows us to place distinct types of information in their proper locations in the system pipeline in compression — and more broadly, communication — systems is highly useful. This paper turns to the setting of lossy compression and shows how properly constructed separation is possible there.

II. Problem setting and background

Consider an input string $s^n = s_1, ..., s_n$ as drawn from a probabilistic source $\mathsf{s}^n \sim p_{\mathsf{s}^n}$. When the typical set of s^n is too large to represent losslessly (e.g. infinite in size), or we are unwilling to expend enough bits to represent each of its elements distinctly, we resort to lossy compression where $s^n \in \mathbf{S}^n$ is mapped to a reproduction $\hat{s}^n \in \mathbf{S}^n$, where generally $s^n \neq \hat{s}^n$. The mapping is evaluated under a *distortion measure* $d_n : \mathbf{S}^n \times \mathbf{S}^n \to \mathbf{R}_{\geq 0}$ to assess its average *distortion level*

$$\bar{d}_n \triangleq \mathbb{E} d_n(\mathsf{s}^n, \hat{\mathsf{s}}^n) \tag{1}$$

*The author is employed by Celect, Inc. and a Research Affiliate at MIT.

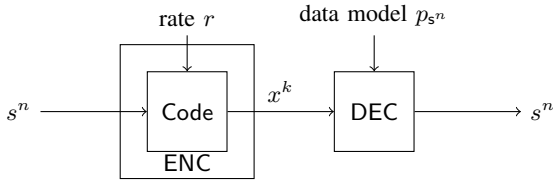

Figure 1: A canonical system with Model-Code Separation, with a model-free hashing encoder and a message-passing inferential decoder binding a data model.

The lossy compression problem under present consideration is to minimize the average distortion level for a given average rate budget, and vice versa.

Rate-distortion theory *[2], [3]* defines the *rate-distortion function* as

$$\mathbb{R}(\Delta; \mathsf{s}) \triangleq \lim_{n\to\infty} \frac{1}{n} \mathbb{R}(\Delta; \mathsf{s}^n) = \lim_{n\to\infty} \frac{1}{n} \left(\inf_{p_{\hat{\mathsf{s}}^n|\mathsf{s}^n} : \bar{d}_n \leq \Delta} \mathbb{I}(\mathsf{s}^n; \hat{\mathsf{s}}^n) \right) \tag{2}$$

whose value lower bounds the average number of bits required to represent each symbol of s^n to within average distortion level Δ.[1] The distribution $p^*_{\hat{\mathsf{s}}^n}$ on $\hat{\mathsf{s}}^n$ that achieves the infimum in Eq. 2 is the *optimal reproduction distribution*, and in a lossy compression analog of the Shannon random codebook scheme, it is used directly to achieve $\mathbb{R}(\Delta; \mathsf{s})$ but gives a JMC design.

In prior literature, Wyner-Ziv coding is suggested as the lossy compression counterpart to Slepian-Wolf coding. In Wyner-Ziv coding, additional side information $y^n = y_1, ..., y_n$ drawn from $(\mathsf{s}^n, \mathsf{y}^n) \sim p_{\mathsf{s}^n\mathsf{y}^n}$ is available at the decoder, and compression rate can be improved to

$$\mathbb{R}^{\text{WZ}}(\Delta; \mathsf{s}|\mathsf{y}) \triangleq \lim_{n\to\infty} \frac{1}{n} \left(\inf_{p_{\hat{\mathsf{s}}^n|\mathsf{s}^n} : \bar{d}_n \leq \Delta} [\mathbb{I}(\mathsf{s}^n; \hat{\mathsf{s}}^n) - \mathbb{I}(\hat{\mathsf{s}}^n; \mathsf{y}^n)] \right) \tag{3}$$

with a scheme that uses an intermediate codebook drawn according to $p^{\text{WZ}}_{\hat{\mathsf{s}}^n}$, the distribution on $\hat{\mathsf{s}}^n$ that achieves the infimum in Eq. 3, then followed by random binning. This style of coding is sometimes exploited in video compression,

[1] Notationally, $\mathbb{R}(\Delta; \mathsf{s})$ is more commonly written as $R(\Delta)$ with the source s understood. The theory holds for (1) stationary ergodic sources of bounded total distortion, i.e. $\mathbb{E} d_n(\mathsf{s}^n, \hat{s}^n) < \infty$, and (2) finite-span distortion measures $d_n(s^n, \hat{s}^n) = (n - g + 1)^{-1} \sum_{k=0}^{n-g} d(s_{k+1,\ldots k+g}, \hat{s}_{k+1,\ldots,k+g})$, where $d : \mathbf{S}^g \times \mathbf{S}^g \to \mathbf{R}_{\geq 0}$.

for example, to enable scalable quality improvement and error resilience [4], [5], [6], [7], [8].

Wyner-Ziv coding has the ingredients of separation but falls short. (If we consider y^n to be a hidden variable on which s^n depends, then there is partial model-code separation, but it is not clear that we can remove all non-coding information to the decoder without forsaking compression gain entirely.) This is because there are two types of non-coding information in lossy compression: the entropic information provided by the data model, and the semantic information provided by the distortion measure.

In the sequel, we propose a separation for lossy compression that keeps semantic information in the encoder and removes entropic information to the decoder. We show this architecture can be realized as an extension of the practical MCS system already constructed for lossless compression, with an additional quantizer component that explicitly supports low-complexity inferential decoding. We also show the performance implications of this separation.

III. Proposed system

Since entropic information and semantic information are inherently different, we need a more nuanced separation design than in Fig. 1. In particular, it can be shown that naive hashing in the absence of semantic information is catastrophic, so lossy coding must be quantization-like, and the distortion measure must accompany it within the encoder to make semantics-aware decisions. However, nothing prevents the data model from being separated from coding; indeed practical lossy compression systems for complex data essentially quantize in the absence of the data model (e.g. images [9], videos [10]). Thus in addition to Model-Code Separation, another separation principle emerges in lossy compression between processing that binds a data model and processing that binds a distortion measure. We call it *Model-Quantizer Separation (MQS)*. Together, we have a design for lossy compression (Fig. 2).

To compress an n-symbol string $\mathrm{s}^n \in \mathbf{S}^n$, we require (1) a stochastic data model $\mathsf{s}^n \sim p_{\mathsf{s}^n}$; (2) a distortion measure $d_n : \mathbf{S}^n \times \mathbf{S}^n \to \mathbf{R}_{\geq 0}$; (3) a collection of quantizers indexed by l, $\mathsf{Quantize}_l : \mathbf{S}^n \to \mathbf{Q}^l$, along with a choice map $\Gamma : \mathbf{Q}^* \to \mathbf{S}^n$; (4) a coding ensemble $\mathscr{H}(l,k)$ of $k \times l$ parity matrices of a rate k/l LDPC source code over some alphabet $\mathbf{Z}$; and (5) (optionally) a representational map $t_{|\mathbf{Q}|\to|\mathbf{Z}|} : \mathbf{Q} \to \mathbf{Z}^{\log|\mathbf{Q}|/\log|\mathbf{Z}|}$ for serializing quantizer output to the alphabet of the code — for simplicity let us assume both are already binary. Construct the following encoder and decoder.

A. Model-free encoder

The encoder performs model-free quantization, followed by model-free coding.

***Quantization* —** Setting l to target an average distortion level Δ, quantize s^n to $\mathrm{q}^l \in \mathbf{Q}^l$ by

$$\mathrm{q}^l = \mathsf{Quantize}_l(\mathrm{s}^n) \tag{4}$$

the average distortion level being

$$\bar{d}_n = \mathbb{E} d_n(\mathsf{s}^n, \Gamma(\mathsf{Quantize}_l(\mathsf{s}^n))) \tag{5}$$

***Coding* —** Setting k to target an overall compression rate of $r = k/n$ bits per input symbol, choose a random $H_{k\times l} \in \mathscr{H}(l,k)$, and produce a hash

$$\mathrm{x}^k = H\mathrm{q}^l \tag{6}$$

as the compressed result. This step is the same as in model-free lossless encoding.

The encoder setup here captures various types of traditional and non-traditional quantization, from vector quantization [11], to coded quantization such as in [12] and locality-sensitive hashing [13], [14].

B. Inferential decoder

The decoder performs message-passing inference on a composite graph combining information on the data model, the quantization procedure, and the coding. Briefly, in this graph (Fig. 3(a)), there are nodes for variables of input $\mathcal{S} \triangleq \{\mathsf{s}_1, ..., \mathsf{s}_n\}$ and quantizer output $\mathcal{Q} \triangleq \{\mathsf{q}_1, ..., \mathsf{q}_l\}$ (and serialized output $\mathcal{Z} \triangleq \{\mathsf{z}_1, ..., \mathsf{z}_l\}$ if appropriate); there are also nodes for factors relating to the data model (Ψ, not shown), to the quantization processing ($\boxed{\mathsf{Q}}$), and to the coding ($\mathcal{X}$). The belief-propagation (BP) algorithm is executed over the entire graph.

At any time, to trial decode $\hat{s}^n$, first let $\beta^b(q_b)$ be the belief on each q_b, then we computing $\hat{q}^l$ as

$$\hat{q}_b = \arg\max_{q_b} \beta^b(q_b) \tag{7}$$

and apply $\hat{s}^n = \Gamma(\hat{q}^l)$. If BP converges, and the inferred values of $\hat{q}^l = \mathsf{Quantize}_l(\hat{s}^n) = \mathrm{q}^l$ and $\hat{x}^k = H\hat{q}^l = \mathrm{x}^k$ are correct, then decoding succeeds, and we achieve the rate-distortion pair (r, Δ).

Note that the graphical models for the data model ($\mathcal{G} = (\mathcal{S}, \Psi, \mathcal{E})$) and coding ($\mathcal{C} = (\mathcal{S}, \mathcal{X}, \mathcal{F})$) portions, as well as message-passing for them, are unchanged from lossless coding [1], and indeed there is a sub-block that performs lossless compression with model-code separation on the quantizer output, within the presently discussed system. The only portion newly added is a graphical model of the quantization process (Fig. 3(b)).

To be concrete, let us next describe encoding and decoding for a particular quantizer that we construct — a very different kind of quantizer than what exists.

IV. Low-density hashing quantizer

Beginning with the inferential decoder design in mind, we can propose a new quantizer design that is more directly suitable than existing quantizers in some combination of ease of implementation, complexity, and performance in graphical decoding.

Referring to Fig. 3, a general full vector quantizer would involve one $\boxed{\mathsf{Q}}$ factor that connects to all $\mathcal{S}$ nodes, and producing one large-alphabet output $\mathcal{Q} = \{\mathsf{q}\}$. However,

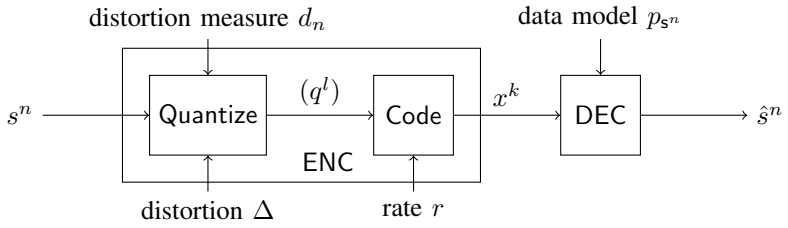

Figure 2: System diagram for a lossy compression architecture featuring model-quantizer separation (MQS) and model-code separation (MCS). The encoder is model-free but distortion measure aware.

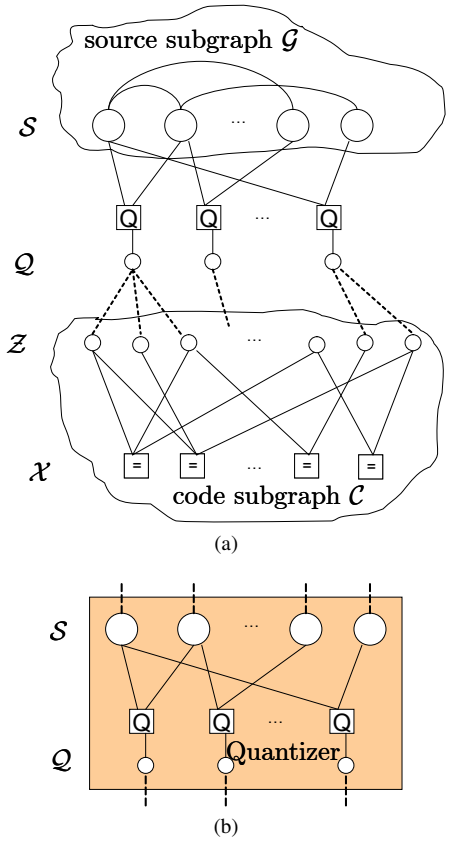

Figure 3: The structure of the decoder inference graph (top), and the modularized quantizer component (bottom) for separation architecture lossy compression. This top drawing assumes a pairwise model on data (factors Ψ along edges not shown), and serialization between $\mathcal{Q}$ and $\mathcal{Z}$ (may be omitted in some settings).

since we would follow that with serialization to take $\mathcal{Q}$ to binary symbols $\mathcal{Z}$, we can consider the vector quantizer to be equivalently decomposed into a multitude of bitwise $\boxed{\mathsf{Q}}$ factors each producing one bit $\mathsf{q}_i \in \mathcal{Q}$. Each $\boxed{\mathsf{Q}}$ now acts like a one-bit quantizer or bit hash. For $\boxed{\mathsf{Q}}$ implementing arbitrary functions, the complexity can still be exceedingly high, given its $\mathcal{O}(n)$ neighborhood — a full vector quantizer still has high complexity even to produce 1 bit.

A natural reduction is to connect each $\boxed{\mathsf{Q}}$ not to all the $\mathcal{S}$ nodes but only to some, giving rise to a product quantizer structure [11]. If the connection is sparse enough, the collection of bitwise quantizers may be called a *low-density hashing quantizer (LDHQ)*.

Now let us describe encoding and decoding with the LDHQ for the binary Hamming case.

A. Quantization function

Each bitwise quantizer $\boxed{\mathsf{Q}}$ implements a specific quantization function, defined as follows in terms of its decoder.

Connect $\boxed{\mathsf{Q}}$ to a random δ nodes of $\mathcal{S}$: $\{\mathsf{s}_{j_1}, \mathsf{s}_{j_2}, ..., \mathsf{s}_{j_\delta}\} \subseteq \mathcal{S}$, so let $s = s_{j_1}, s_{j_2}, ..., s_{j_\delta}$. Let $u \in \mathbf{S}^\delta$ be a random vector called the *reference point*. Let $\bar{u}$, i.e. the inversion of u, be called the *antipodal point*. Then,

$$\mathsf{Quantize}_1(s;u) = \begin{cases} 0 & \text{if } d_\delta(s,u) < d_\delta(s,\bar{u}) \\ 1 & \text{if } d_\delta(s,u) > d_\delta(s,\bar{u}) \\ \mathsf{q} \sim \mathbf{Bern}(\frac{1}{2}) & \text{if } d_\delta(s,u) = d_\delta(s,\bar{u}) \end{cases}$$

defines a 1-bit stochastic quantizer (we can also use a deterministic tiebreak for the third case). Since $d_\delta(s,u)+d_\delta(s,\bar{u}) = \delta$, the quantization function can be computed trivially by

$$\mathsf{Quantize}_1(s;u) = \mathbb{1}\{d_\delta(s,u) \gtrless \frac{\delta}{2}\} \tag{8}$$

B. LDHQ in encoder

We only describe the quantization step of Section III-A. To quantize s^n, setting l to target the total number of quantization bits — this sets the target distortion — choose a random bipartite graph $\mathcal{L} = (\mathcal{S}, \mathcal{Y}, \cdot)$, where $\mathcal{Y} \triangleq \{g_1, ..., g_l\}$ denotes the l $\boxed{\mathsf{Q}}$ factor functions, and the degree on each g_b is δ_b. Write $A_b = \mathcal{N}_b^{\mathcal{L}}$ for the neighborhood of g_b on the $\mathcal{S}$ side. Choose also a collection of random reference points $\mathscr{U} = \{u_1, u_2, ..., u_l\}$, $u_b \in \mathbf{S}^{\delta_b}$. Apply

$$\begin{aligned} \mathrm{q}^l &= \mathsf{Quantize}_l(\mathrm{s}^n; \mathscr{U}, \mathcal{L}) \\ &\triangleq \begin{bmatrix} \mathsf{Quantize}_1(\mathrm{s}_{A_1}; u_1) \\ \mathsf{Quantize}_1(\mathrm{s}_{A_2}; u_2) \\ \vdots \\ \mathsf{Quantize}_1(\mathrm{s}_{A_l}; u_l) \end{bmatrix} \end{aligned} \tag{9}$$

This is followed by coding as usual.[2]

[2] $\mathscr{U}$, $\mathcal{L}$, just as H, are to be generated pseudo-randomly, and likewise need not be described explicitly to the decoder beyond synchronizing a seed.

C. LDHQ in decoder

We only describe the quantizer component of Section III-B. Analogous to the *hash constraint function* of coding, $c(q^l) \triangleq \mathbb{1}\{\mathrm{x}^l = H\mathrm{q}^l\}(q^l)$, define the *quantizer constraint function*,

$$\begin{aligned} g(s^n, q^l) &\triangleq \mathbb{1}\left\{\mathrm{q}^l = \mathsf{Quantize}_l(s^n; \mathscr{U}, \mathcal{L})\right\}(s^n, q^l) \\ &= \prod_{b=1}^{l} \mathbb{1}\left\{\mathrm{q}_b = \mathsf{Quantize}_1(s_{A_b}; u_b)\right\}(s_{A_b}, q_b) \\ &= \prod_{b=1}^{l} g_b(s_{A_b}, q_b) \end{aligned} \tag{10}$$

Together with the data model $p_{\mathsf{s}^n}(s^n)$, the entire decoder attempts to marginalize

$$u(s^n, q^l) \triangleq c(q^l) g(s^n, q^l) p_{\mathsf{s}^n}(s^n) \tag{11}$$

for maximization over each s_i.

To obtain the messages for BP, notice the quantizer component sends and receives external messages on the ports of both sets of variable nodes ($\mathcal{S}$ and $\mathcal{Q}$) (Fig. 3(b)). Let $\left[M^{i\leftarrow}\right](s_i)$ and $\left[M^{i\rightarrow}\right](s_i)$ denote respectively the input and output external messages on variable s_i. Let $\left[M^{b\leftarrow}\right](q_b)$ and $\left[M^{b\rightarrow}\right](q_b)$ denote respectively the input and output external messages on variable q_b. Denote by $\nu^{i\rightarrow b}(s_i)$ a message passed from s_i to g_b, and by $\nu^{i\leftarrow b}(s_i)$ one passed from g_b to s_i. Then, the local message updates of the LDHQ quantizer component are:[3]

$$\begin{aligned} \mathcal{S} \text{ node output:} &\quad \nu^{i\rightarrow b} \Leftarrow \nu^{i\leftarrow\sim b}\left[M^{i\leftarrow}\right] \\ \mathcal{S} \text{ external message output:} &\quad \left[M^{i\rightarrow}\right] \Leftarrow \nu^{i\leftarrow *} \\ \mathcal{Q} \text{ node ouptut:} &\quad \nu^{i\leftarrow b} \Leftarrow g^i_{\sim i,b}\nu^{\sim i\rightarrow b}\left[M^{b\leftarrow}\right] \\ \mathcal{Q} \text{ external message output:} &\quad \left[M^{b\rightarrow}\right] \Leftarrow g^b_* \nu^{*\rightarrow b} \end{aligned}$$

Interestingly, in LDHQ decoding, there is no explicit choice map Γ. It emerges from the decoding process. In other words, the $\hat{s}^n$ that decoding converges to for a given $\hat{q}^l$ is not pre-determined, though it will of course satisfy quantizer constraint.

D. Results

Although we have sought no theoretical guarantee that LDHQ performs well, (partly because performance is a function of a tradeoff with degree and complexity of the graph $\mathcal{L}$), we can nevertheless demonstrate that the very important linear processing bound of Ancheta [15] can be breached at low complexity, even within the context of a separation design. We compress $\mathsf{s}^n \sim \mathbf{Bern}(p)$, $p = 1/2$, with Hamming distortion $d_n(s^n, \hat{s}^n) \triangleq (1/n)\sum_{i=1}^n \mathbb{1}\{s_i \neq \hat{s}_i\}$. The rate-distortion function for this source is, for $0 \leq \Delta \leq \min\{p, 1-p\}$,

$$\mathbb{R}(\Delta; \mathsf{s}) = h(p) - h(\Delta) \tag{12}$$

[3] $\nu^{i\leftarrow\sim b}$ means the product of $\nu^{i\leftarrow x}(s_i)$ for all g_x except g_b in the neighborhood of s_i in $\mathcal{L}$; $\nu^{i\leftarrow *}$ is the same but does not except g_b. $\nu^{\sim i\rightarrow b}$ means the product of $\nu^{x\rightarrow b}(x)$ for all s_x in the neighborhood of g_b except s_i; $\nu^{*\rightarrow b}$ is the same but does not except g_b. $g^i_{\sim i,b}\nu^{\sim i\rightarrow b}$ means summing out all variables except s_i in $g_b(s_{A_b}, q_b)\nu^{\sim i\rightarrow b}(s_{A_b\setminus \mathsf{s}_i})$; $g^b_*\nu^{*\rightarrow b}$ means summing out all variables in $g_b(s_{A_b}, q_b)\nu^{*\rightarrow b}(s_{A_b})$ except q_b.

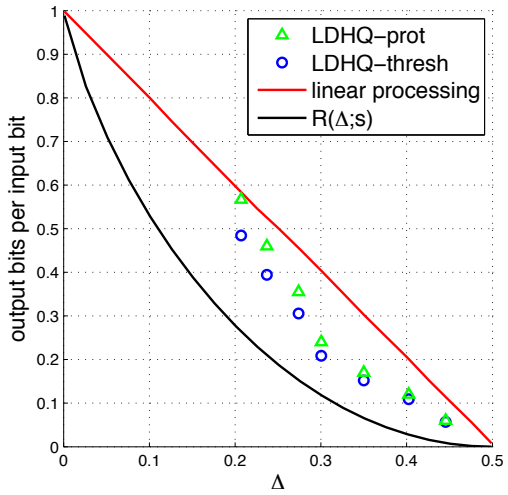

Figure 4: Compressing $\mathbf{Bern}(\frac{1}{2})$ under Hamming distortion with LDHQ. $n = 1000$, 4% doping bits included in reported rate.

The linear processing bound for $p = 1/2$ is, for $0 \leq \Delta \leq p$,

$$R^{\text{linear}}(\Delta) = 1 - 2\Delta \tag{13}$$

Fig. 4 shows a few trials where we borrow the decoding graph of a regular LDPC code as $\mathcal{L}$, and set a constant degree on $\boxed{\mathsf{Q}}$ of only 3. `LDHQ-prot` shows the coding rates and `LDHQ-thresh` shows the threshold rates of BP decoding [16]. We see that, particularly at lower rates, decoding succeeds with demonstrably better performance than the linear processing bound.

The result suggests that forms of low-complexity, semantics-aware hashing may provide the desired performance in lossy compression problems to support a separation architecture.

V. Cost of Separation

Like Wyner-Ziv coding, and unlike Slepian-Wolf coding, there is a performance cost to MQS at low rates. We report a result on how much this cost must be.

The main result is on the scenario of using an arbitrary codebook $\mathscr{Q}_n$ for quantization, and entropy coding on its output $\hat{s}^n(q^l)$, under the specific condition that we know the original source distribution p_{s^n} at the decoder — not a condition usually considered. To do this, we explicitly view the quantization codebook as a stochastic collection $\mathsf{Q}_n \triangleq \{(\hat{\mathsf{s}}^n)_1, (\hat{\mathsf{s}}^n)_2, ..., (\hat{\mathsf{s}}^n)_{|\mathscr{Q}_n|}\}$ of reproduction words drawn from a *design distribution* on the words of $\mathbf{S}^n$.

Definition 1. Let P be the source law, i.e. p_{s^n}. Let W be an arbitrary stochastic encoding law, i.e. $p_{\hat{\mathsf{s}}^n|\mathsf{s}^n}$. Let PW be the joint distribution of the source-reproduction pair under source law P and encoding law W, i.e. $p_{\mathsf{s}^n\hat{\mathsf{s}}^n}$. Let Y be the output distribution under source law P and encoding law W, which is to say, $p_{\hat{\mathsf{s}}^n} = \int_{s^n} p_{\mathsf{s}^n}(s^n) p_{\hat{\mathsf{s}}^n|\mathsf{s}^n}(\hat{s}^n|s^n)$ or $Y = \int_P PW$ for short. (Y is the *effective codebook distribution*; it can differ

from the design distribution.) Let Q be an arbitrary codebook design distribution. Let $W_Q \triangleq \arg\inf_W D(PW||P \times Q)$ denote the optimal encoding law when coding a P-source with an arbitrary codebook of design distribution Q. Let $Y_Q = \int_P PW_Q$ denote the associated effective codebook distribution at the output.

The performance of lossy compression as fundamentally a codebook mismatch problem is extensively surveyed in [17] by large deviation methods and we modify Theorem 1 to take into account the fact that we know P (and of course Q and therefore Y_Q) after quantization.

Theorem 2. *The optimal rate for rate-distortion coding a P-source, using a quantizer with codebook design distribution Q at the encoder, with knowledge of P and Q at the decoder, is*

$$\begin{aligned} R^{Q,\mathrm{EC}}(\Delta) &= \inf_W D(PW||P \times Y_Q) \\ &= \inf_W \left[D(PW||P \times Q)\right] - D(Y_Q||Q) \\ &\triangleq R^Q(\Delta) - D(Y_Q||Q) \end{aligned}$$

with all optimizations subject to $\mathbb{E}d_n(\mathsf{s}^n, \hat{\mathsf{s}}^n) \leq \Delta$.

$R^Q(\Delta)$ is the rate required to use codebook Q on P. The term $D(Y_Q||Q)$ is the rate recovered by entropy coding the quantizer output. Furthermore, the encoder can apply model-free coding (lossless compression) at rate $R^{Q,\mathrm{EC}}(\Delta)$, knowing the decoder, with P and Q at hand, will "compute" and apply the model for Y_Q, the utilized, effective distribution of the codebook Q_n, rather than Q itself. Note that this "computation" need not be an explicit one, since the distribution emerges on the graphical nodes s^n from the inferential decoder applying the data model $\mathcal{G}$ and the quantizer design $\mathcal{L}$ (encoding law and design distribution).

A. Example

Fig. 5 shows the various rate functions computed for the i.i.d. source $\mathbf{Bern}(p)$ (codebooks are also i.i.d., thus all quantities refer now to single-letter marginals) with Hamming distortion. The optimal reproduction distribution is

$$Q^* = \mathbf{Bern}\left(\frac{p-\Delta}{1-2\Delta}\right) \tag{14}$$

We are particularly interested in the uniform (or other entropy maximizing) distribution for the quantizer, because a model-free quantizer almost certainly has to use it. We see $R^{Q,\mathrm{EC}}(\Delta)$ converging to $\mathbb{R}(\Delta;\mathsf{s})$ at high rate and also perform well at intermediate rates. At lower rates, time-sharing in an architecturally compatible way with the $\mathbb{R}(\Delta;\mathsf{s})$ zero-rate extremum (i.e. not coding some symbols) gives performance much closer to $\mathbb{R}(\Delta;\mathsf{s})$.

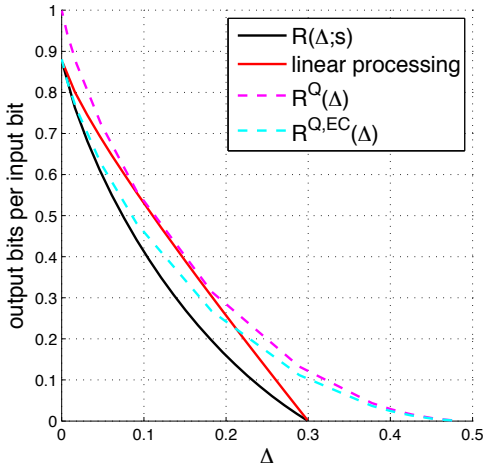

Figure 5: Performance of compressing $P = \mathbf{Bern}(0.3)$ with $Q = \mathbf{Bern}(0.5)$ when $Q^* = \mathbf{Bern}(p \neq 0.5)$.

REFERENCES

[1] Y.-z. Huang and G. Wornell, "A class of compression systems with model-free encoding," in *Proc. ITA*, Feb. 2014.

[2] T. Berger, *Rate Distortion Theory: Mathematical Basis for Data Compression*. Prentice Hall, Oct. 1971.

[3] T. M. Cover and J. A. Thomas, *Elements of information theory*. Wiley, 1991.

[4] A. Aaron, E. Setton, and B. Girod, "Towards practical Wyner-Ziv coding of video," in *2003 ICIP, 2003. ICIP 2003. Proc.*, vol. 3, Sep. 2003, pp. III–869–72 vol.2.

[5] A. Aaron, S. Rane, E. Setton, and B. Girod, "Transform-domain Wyner-Ziv codec for video," in *Proc. SPIE*, vol. 5308, 2004, pp. 520–528.

[6] D. Rebollo-Monedero and B. Girod, "Generalization of the rate-distortion function for Wyner-Ziv coding of noisy sources in the quadratic-gaussian case," in *Data Compression Conference, 2005. Proc.. DCC 2005*. IEEE, 2005, pp. 23–32.

[7] D. Kubasov, K. Lajnef, and C. Guillemot, "A hybrid encoder/decoder rate control for Wyner-Ziv video coding with a feedback channel," in *IEEE 9th Workshop on Multimedia Signal Processing, 2007. MMSP 2007*, Oct. 2007, pp. 251–254.

[8] D.-k. He, A. Jagmohan, L. Lu, and V. Sheinin, "Wyner-Ziv video compression using rateless LDPC codes," in *Proc. VCIP*, vol. 8, 2008.

[9] A. Skodras, C. Christopoulos, and T. Ebrahimi, "The JPEG 2000 still image compression standard," *IEEE Signal Proc. Magazine*, vol. 18, no. 5, pp. 36–58, Sep. 2001.

[10] H. Malvar, A. Hallapuro, M. Karczewicz, and L. Kerofsky, "Low-complexity transform and quantization in h.264/AVC," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 598–603, Jul. 2003.

[11] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression*. Springer, Nov. 1991.

[12] M. Wainwright, E. Maneva, and E. Martinian, "Lossy source compression using low-density generator matrix codes: Analysis and algorithms," *IEEE Trans. Inform. Theory*, vol. 56, no. 3, pp. 1351–1368, Mar. 2010.

[13] P. Indyk and R. Motwani, "Approximate nearest neighbors: Towards removing the curse of dimensionality," in *Proc. the Thirtieth Annual ACM Symposium on Theory of Computing*. ACM, 1998, pp. 604–613.

[14] P. T. Boufounos and S. Rane, "Efficient coding of signal distances using universal quantized embeddings," in *Data Compression Conference (DCC), 2013*. IEEE, 2013, pp. 251–260.

[15] J. L. Massey, "Joint source and channel coding," 1977.

[16] T. Richardson and R. Urbanke, *Modern Coding Theory*. Cambridge Univ. Press, Mar. 2008.

[17] A. Dembo and I. Kontoyiannis, "Source coding, large deviations, and approximate pattern matching," *IEEE Trans. Inform. Theory*, vol. 48, no. 6, pp. 1590–1615, Jun. 2002.