ADAPTIVE QUANTIZATION WITHOUT SIDE INFORMATION

Antonio Ortega [*]

Dept. of EE-Systems and SIPI,
Univ. of Southern California
Los Angeles, CA 90089
ortega@sipi.usc.edu

Martin Vetterli

Dept. of EECS,
Univ. of California
Berkeley, CA 94720
martin@eecs.berkeley.edu

ABSTRACT

We propose to extend some of the ideas of adaptive lossless compression to design an adaptive quantization algorithm. Noting that the performance of an arithmetic coder is as good as its estimation of the statistics of the input, we split our quantizer into two building blocks: *model estimation* and *quantizer design*. The main idea is that, as long as the model estimation manages to track down the changes in source statistics, a standard quantizer design technique *which assumes that the source follows the estimated model* can be used. As an example of this type of design we study an adaptive scalar quantization scheme where we impose the restriction that no side information can be sent, i.e. encoder and decoder must perform their adaptation based on the quantized information.

1. INTRODUCTION

The most successful methods for lossless compression of data, such as arithmetic coding [1], Lempel-Ziv coding [2] or dynamic Huffman coding [3], are all adaptive (see [4] for an extensive review of lossless compression). While the initial work on entropy coding (e.g. Huffman coding) relied on knowing, or measuring, the source distribution, adaptive schemes make no prior assumptions on the source statistics, which the coders try to learn. Here, we are concerned with designing adaptive quantization algorithms which exhibit characteristics similar to those of the abovementioned adaptive lossless compression schemes. In the most general context we can define our problem as that of adapting some or all of the parameters of a quantizer/entropy coder system (including bin sizes, reconstruction levels, codeword lengths and dynamic range) to the changing statistics of an input source. We make few assumptions on the source and will study the algorithm's performance for both independent identically distributed (i.i.d.) sources and non-i.i.d. sources exhibiting long term memory.

We are concerned with systems where the adaptation occurs based *only* on the causal past so that both encoder and decoder can adapt in the same manner, and no extra information needs to be sent. We propose to split the adaptation algorithm in two parts (see Fig. 1): (i) Model estimation: Based on the previous N samples we estimate the distribution function of the source, and (ii) Quantizer design: for the given estimated distribution the new quantizer parameters are computed. The advantage of splitting the algorithm in this manner is that well known optimal quantization techniques can then be used. If we correctly estimate the distribution then we are guaranteed optimality.

A further question arises as to how much memory should be used in estimating the distribution. Clearly, if the source input were i.i.d., it would be reasonable to accumulate statistics over a long time window. Conversely, if the source input distribution were changing over time, shorter windows would have to be used. We are thus interested in systems (see Fig 1) where the window size or, equivalently, the speed of adaptation, can be changed over time.

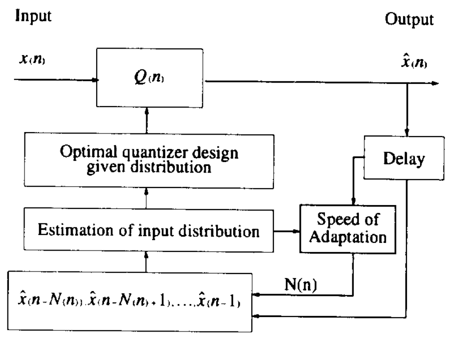

Input
Output
$x(n)$
$Q(n)$
$\hat{x}(n)$
Optimal quantizer design given distribution
Delay
Estimation of input distribution
Speed of Adaptation
N(n)
$\hat{x}(n-N(n)), \hat{x}(n-N(n)+1), \ldots, \hat{x}(n-1)$

Figure 1: Adaptive quantizer. The current input distribution is estimated based on previously quantized samples.

In some adaptive quantization techniques[5, 6] the objective is to adjust the support region of a uniform scalar quantizer, so that this quantizer can be used in conjunction with a predictor in a DPCM system. In [7] an initial tree structured vector quantizer (TSVQ) is first designed, with a rate higher than the rate available for transmission. Then the adaptive algorithm chooses which subtree of the previously designed tree has to be used at every instant, by keeping counts of the number of samples that corresponded

[*]Work supported in part by a scholarship from the Fulbright Commission and the Ministry of Education and Science of Spain. This work was performed while at the Dept. of Elec. Eng. and Center for Telecom. Research, Columbia University, NY, NY

to each of the nodes of the tree, selecting the subtree which minimizes the expected distortion. While both methods resort to an underlying of the input source, here we seek to explicitly obtain the model before redesigning the quantizer.

The topic of adaptation has been extensively dealt with in the area of lossless data compression, where the two main approaches are [4] are model-based (e.g. Arithmetic Coding (AC) or adaptive Huffman coding) and dictionary-based (e.g. Lempel Ziv (LZ) coding), where the adaptivity comes from dynamically updating, respectively, the model and the dictionary. In the AC algorithm [1], in the simpler case of a binary source, the encoder has to update the probabilities of the 0's and 1's. If the source is stationary and the model is correct then AC can provide a performance very close to the first order entropy. However, in real life environments, where sources need not be stationary, the performance of the algorithm is determined by how well it adapts to the changing statistics of the source. In that sense, the model tracking part of the AC algorithm plays an essential part in the system performance. It is also worth noting the link between the compression problem and that of obtaining of a good model for random data, which is the basis for the minimum description length (MDL) technique introduced by Rissanen [8].

Section 2 will describe the algorithm by detailing the building blocks of Fig. 1. Section 3 will present some experimental results.

2. ADAPTATION ALGORITHM

2.1. Estimation of input distribution

Objective 1 *Given the N most recent quantized sample occurrences $\hat{x}(n-N)$, $\hat{x}(n-N+1)$, ..., $\hat{x}(n-1)$, where N might be a constant or can be changed by the speed adaptation algorithm, find an estimate $\hat{f}(x)$ of the probability distribution function of the source, $f(x)$*[1].

We will use the following notation. The quantizer has L reconstruction levels r_i with $L-1$ decision levels denoted $b_1, \ldots, b_{L-1}$. The counts of how many samples (out of the last N) fell into each of the bins are denoted $n_0, \ldots, n_{L-1}$, where n_0 and n_{L-1} are the number of samples that fell in the "outer" bins. Our goal is to, given the knowledge of $n_0, \ldots, n_{L-1}$ and $b_1, \ldots, b_{L-1}$, find a good approximation $\hat{f}(x)$. From the observed data we can deduce that:

$$P_i = \int_{b_i}^{b_{i+1}} f(x)\,dx = \frac{n_i}{N}, \qquad (1)$$

for $i = 0, \ldots, L-1$, and $b_0 = -\infty, b_L = +\infty$. Although strictly speaking the equality holds only in the limit as N goes to infinity, it is a sufficiently good approximation.

The task of determining $\hat{f}(x)$ is complicated by the fact that we are limiting ourselves to accessing only the quantized data. The problem can be separated into two parts: (i) estimating $\hat{f}(x)$ in the two outer bins, where we can only rely on knowing one of the boundaries, and (ii) estimating $\hat{f}(x)$ within the inner bins, where we know both boundaries.

[1]Although we here refer to $f(x)$ as the pdf, we do so by an abuse of language. Strictly speaking we are gathering short term data and assuming that there is an underlying pdf or model which produced the data.

For simplicity, we resort to approximation fucntions $\hat{f}(x)$ that are piecewise linear. In the more general case, we can choose a set of P, $P \geq L$, points, $x_0, \ldots, x_{P-1}$ and our objective will be to find $\hat{f}(x_0), \ldots, \hat{f}(x_{P-1})$, while $\hat{f}(x)$ can be linearly interpolated at other points x. The x_i can be chosen arbitrarily within the estimated dynamic range of the source, say $[b_0, b_L]$. The task of approximating the dynamic range will be dealt with in more detail in section 2.1.1.

Assume, thus, $[b_0, b_L]$ given and choose $P \geq L$ points which, for simplicity we assume equally spaced. Further assume that the pdf $f(x)$ that we are trying to approximate is smooth in some sense. Then we can aim at finding $\hat{f}$ such that

$$\int_{b_i}^{b_{i+1}} \hat{f}(x)\,dx = P_i, \quad \text{for} \quad i = 0, \ldots, L-1. \qquad (2)$$

Since $\hat{f}$ is a piecewise linear approximation we can write the equations (2) as a function of the P unknowns $\hat{f}(x_0)$, ..., $\hat{f}(x_{P-1})$. This can be seen as a typical *inverse problem* which in the case of $P > L$ is overdetermined [9]. A linear regularization method that has the advantage of resorting to the pdf smoothness assumption is described in [10]. We now propose a simpler approach that requires only $P = L$ points and involves no iterations.

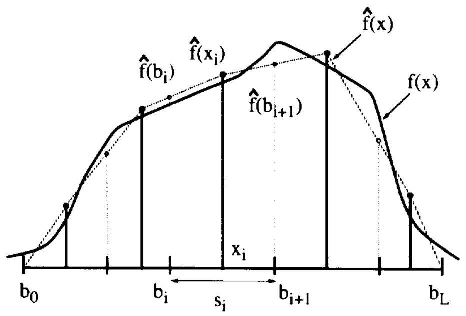


Figure 2: Notation used in the model estimation algorithm. The b_i's denote the decision levels, with b_0 and b_L denoting the outer boundaries of the finite support approximation. The x_i are the knots of the piecewise linear approximation.

Assume again that we have chosen the boundaries b_0 and b_L, such that $\hat{f}(b_0) = \hat{f}(b_L) = 0$, as our estimate of the dynamic range (Refer to Fig. 2). Furthermore, assume that we estimate that our choice of b_0, b_L is expected to "leave out" a fraction of the tail of the distribution such that $\int_{-\infty}^{b_0} f(x)dx = \int_{b_L}^{+\infty} f(x)dx = P_{out}$ (see section 2.1.1). Then, denoting $P_0' = P_0 - P_{out}$ and $P_{L-1}' = P_{L-1} - P_{out}$ with $P_i' = P_i$ for $i = 1, \ldots, L-2$, we can choose $P = L$ points x_i at which we need to calculate the function values $\hat{f}(x_i)$ such that $\hat{f}$ will meet the constraint of (2). To restrict the number of degrees of freedom we arbitrarily choose the x_i to be center of each of the inner bins.

Now we can rewrite (2) by computing the integrals over each bin $[b_i, b_{i+1}]$ of $\hat{f}(x)$. Our goal is to find the $\hat{f}(x_i)$ such that

$$(\hat{f}(x_i)+\hat{f}(b_i))(x_i-b_i)+(\hat{f}(x_i)+\hat{f}(b_{i+1}))(b_{i+1}-x_i)=2P_i \tag{3}$$

where $\hat{f}(b_i)$ can be found by linear interpolation. Note that, since we have only one "knot" per bin, each of the equations (3) involves at most three unknowns $\hat{f}(x_{i-1}), \hat{f}(x_i), \hat{f}(x_{i+1})$ so that the system we have to solve is

$$\mathbf{T}\cdot\mathbf{f}=\mathbf{p}' \tag{4}$$

where $\mathbf{T}$ is a $L\times L$ tridiagonal matrix and $\mathbf{p}'$ denotes the vector of observed probabilities (with the corrected tails). Efficient gaussian substitution methods can be used to solve this system [9].

2.1.1. Estimation of the dynamic range

Objective 2 *Find b_0 and b_L, defined as the points such that we estimate the source pdf to be "almost zero". For these points we will have thus $\hat{f}(b_0)=0$ and $\hat{f}(b_L)=0$.*

The difficulty here stems from the fact that we have limited information: we know that n_0, resp. n_{L-1}, samples fell below b_1, resp. above b_{L-1}, but we need to use some of our assumptions to estimate b_0 and b_L. Obviously the main assumption is that the outer bins should contain the tails of the distribution. Based on the available information, i.e. the counts n_i, the current decision levels b_i, $i=1, L-1$, and b_0^{old} and b_L^{old} the dynamic range estimates obtained in the previous iteration, we will consider three cases as follows (we outline the algorithm for adjusting b_0, but the same ideas apply for b_L):

(1) if $n_0=0$, i.e. the outer bin is empty, we readjust the boundaries so that $b_0=b_1$ (unless the adjacent bin is also empty), and we then "split" one of the inner bins (e.g. the one where we observed more samples), say i, and we assign $n_i/2$ samples to each of the newly formed bins.
(2) if $n_0/(b_1-b_0^{old}) > n_1/(b_2-b_1)$ then clearly our current estimate is incorrect since we assume smoothly decaying tails for the distribution and we are observing more "sample density" in the outer bin . We have to expand the quantizer range and thus choose the new boundaries so that the two adjacent bins have the same sample density, thus we pick $b_0=b_1-(n_0/n_1)(b_2-b_1)$.
(3) the two previous cases occur when there is a large enough disparity between our current estimate and the "true" short term source distribution. When our estimate is sufficiently good that neither (1) nor (2) apply, we assume that the tail of the distribution is gaussian. We estimate the mean and variance based on the n_i and and we choose the outer boundary so that the tail beyond b_0 has a probability of less than some threshold P_{out} and thus the requirement that $\hat{f}(b_0)\simeq 0$ is met.

Note that cases (1) and (2) have to be dealt with separately since they represent cases where our previous estimates are incorrect and therefore would result in incorrect mean and variance estimates. Furthermore, it is clear that cases (1) and (2) would not occur if we updated the quantizer sufficiently often (as in [5] where the quantizer is recomputed after each quantized sample is received). In that sense (1) and (2) are safeguards to enable a less complex operation of the algorithm.

2.2. Quantizer design for estimated distribution

Objective 3 *Redesign the quantizer for the given distribution $\hat{f}$. This can be done by using an optimal quantizer design algorithm which assumes $\hat{f}$ as the input distribution.*

As an example, we can design a constant rate quantizer simply using the Lloyd-Max algorithm [11] for the given piecewise linear approximation. The task is to choose a new set of bin boundaries b_i', as well as the corresponding reconstruction levels r_i', such that the expected distortion for the distribution $\hat{f}(x)$ is minimized. Note that, as is the case with Huffman coding for example, one can guarantee optimality provided the model matches the source distribution. The same framework can be used with a variable rate entropy constrained design [12].

It is important to note that once we have estimated a model (i.e. chosen the $\hat{f}(x_i)$) the model is not modified by the algorithm that redesigns the quantizer. Furthermore, since our system keeps a running memory of the counts for each bin (the counters are not reset to zero after the quantizer has been redesigned) we also change the counters to adjust for the new bin sizes. Therefore, after the quantizer design stage, and calling b_i' and n_i', respectively, the new bin boundaries and the updated estimated bin counts, we have that:

$$n_i' = N\cdot\int_{b_i'}^{b_{i+1}'}\hat{f}(x)\,dx. \tag{5}$$

2.3. Determining the speed of adaptation

Objective 4 *Dynamically determine at every iteration the number of past samples N that should be used in estimating the pdf.*

The classes of error produced by the choice of memory can be separated into two classes:

(a) if not enough memory (N small) is used we may be dealing with a *non-significant* (in a statistical sense) set of data and our estimation will necessarily be erroneous. **(b)** if the source statistics (as determined by time averages over finite windows) change over time then an excess of memory (N large) will not permit sufficient adaptivity and will result in loss of performance.

In our experiments we choose to keep two set of counters, one accumulating the long term statistics, the other accumulating the latest pattern of sample arrivals. We choose to use the short term data to estimate the model only if the difference between short and long term data exceeds a threshold. In this way, we try to detect the changes in statistics while avoiding always using a short term estimate, and thus risking having to deal with non-significant data.

3. EXPERIMENTAL RESULTS

In this section we present several examples to illustrate the performance of the adaptive quantization scheme of Section 2. A study of the convergence characteristics of the algorithm can be found in [10]. Most examples are provided for fixed rate quantizers at a rate of two bits per sample. The examples with variable rate quantization indicate the achieved SNR vs. entropy trade-off. Note that we use the normalized SNR, $\log(\sigma_x^2/\sigma_r^2)$ where σ_x^2 and σ_r^2 are respectively the variance of the signal and that of the error and are computed using time averages over finite windows.

3.1. Advantages of adaptivity

An adaptive algorithm can be useful even in the case of stationary sources. In particular, adaptive schemes do not require a previous design and can learn the distribution "on the fly" (for instance, they could operate in "training mode" part of the time, typically at the beginning of the transmission). Furthermore, because they are not designed for a specific distribution they do not suffer the shortcoming of loss of performance in the face of mismatch between the actual source distribution and the one that was assumed in the design. Two examples of this can be seen in Figs. 3(a) and (b), where the behavior of the adaptive algorithm and a Lloyd-Max quantizer are compared when the mean and variance of the source, respectively, do not match those assumed in the design.

A second advantage of using an adaptive algorithm is that it can outperform systems that are designed considering only long term statistics, by attempting to find short term trends in the data. As an example, Fig 4(a) shows the performances of the Lloyd-Max algorithm (trained on the sequence) and the adaptive algorithm for a bimodal source which randomly switches between two states each producing different mean. When an i.i.d. source is considered though, the adaptive approach will be less effective although, as shown in Fig. 4(b) for a gaussian distribution, only marginally so (less than $0.05dB$). Note that the results of Fig. 4 were obtained using the *same* parameters in the algorithm (initilization, thresholds, etc) for both sources. Fig. 5 shows that the advantage of adaptivity can also be obtained within an entropy constrained variable rate quantization framework [12].

3.2. Loss due to adaptivity

To estimate the loss due to adaptivity, we initialize the adaptive algorithm with the optimal Lloyd-Max quantizer trained on the gaussian i.i.d. source, rather than a uniform quantizer as was usually the case. In this way, since our first "guess" was optimal, the loss in performance is due exclusively to the adaptivity.

In Table 1 the recurrence time is the period between consecutive quantizer updates. The memory (measured in units of the recurrence times) represents the number of samples that are considered to generate the new quantizer. For instance a memory of 1.25 implies that the previous 50 samples are used when the recurrence time is 40, a memory of $+\infty$ means that all previous samples are considered at every update. We note that, as the number of samples becomes small the main factor becomes the "non-significance" error, i.e. not enough information is used in updating the quantizers. This error can be overcome by appropriate choice of the speed of adaptation. Conversely, for long update intervals the main factor becomes the error introduced by the algorithm itself due to its manipulating quantized data, rather than the original samples as in the Lloyd-Max algorithm. This error can be seen to be very small.

	Recurrence time T (samples)			
Memory (times T)	40	200	400	2000
1.25	8.824	9.157	9.220	9.259
1.67	8.903	9.210	9.241	9.264
2.5	9.109	9.240	9.257	9.266
5	9.154	9.260	9.265	9.267
$+\infty$	9.241	9.264	9.266	9.267

Table 1: SNR at different speeds of adaptation for an i.i.d. source when the adaptive algorithm was initialized with the Lloyd-Max quantizer designed on the actual data. The Lloyd-Max performance is 9.271 dB.

4. REFERENCES

[1] G. G. Langdon and J. Rissanen, "Compression of black-white images with arithmetic coding," *IEEE Trans. on Comm.*, vol. COM-29, pp. 858-867, Jun. 1981.

[2] J. Ziv and A. Lempel, "A universal algorithm for sequential data compression," *IEEE Trans. on Info. Th.*, vol. IT-23, pp. 337-343, May 1977.

[3] R. S. Gallager, "Variations on a theme by Huffman," *IEEE Trans. on Info. Th.*, vol. IT-24, pp. 668-674, Nov. 1978.

[4] T. C. Bell, J. G. Cleary, and I. H. Witten, *Text Compression*. Englewood Cliffs, NJ: Prentice Hall, 1990.

[5] N. S. Jayant, "Adaptive quantization with a one-word memory," *Bell Sys. Tech. J.*, vol. 52, pp. 1119-1144, Sept. 1973.

[6] A. R. Calderbank, S. W. McLaughlin, and D. F. Lyons, "A low complexity two-stage adaptive vector quantizer," in *Proc. of the 25th CISS*, (Baltimore, Md.), pp. 582-587, Mar. 1991.

[7] R. Chang, W. Chen, and J. Wang, "Image sequence coding using adaptive tree-structured vector quantisation with multipath searching," *IEE Proc. I*, vol. 139, pp. 9-14, Feb. 1992.

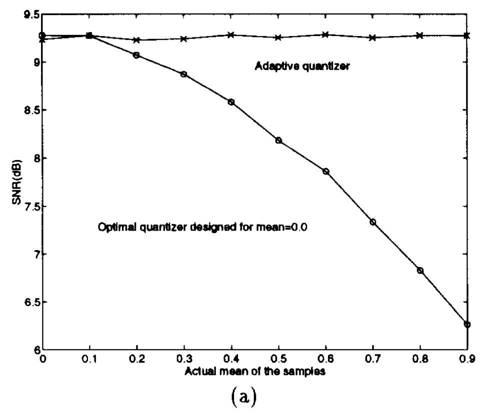
[8] J. Rissanen, "Universal coding, information, prediction and estimation," *IEEE Trans. on Info. Th.*, vol. IT-30, pp. 629-636, 1984 1984.

[9] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C.* Cambridge University Press, 2nd ed., 1992.

[10] A. Ortega, *Optimization Techniques for Adaptive Quantization of Image and Video under Delay Constraints*. PhD thesis, Dept. of Electrical Engineering, Columbia University, New York, NY, 1994.

[11] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression*. Kluwer Academic Publishers, 1992.

[12] P. A. Chou, T. Lookabaugh, and R. M. Gray, "Entropy-constrained vector quantization," *IEEE Trans. on Signal Proc.*, vol. 37, pp. 31-42, Jan. 1989.



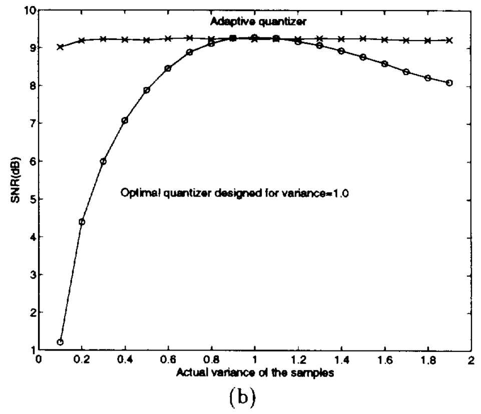
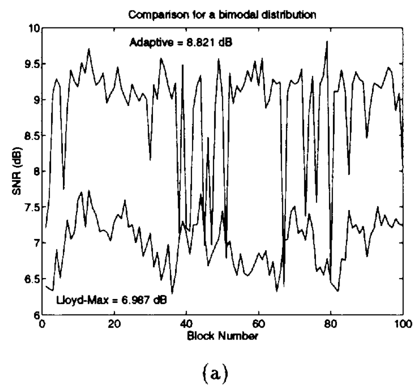


Figure 3: Adaptive vs. Lloyd-Max for Gaussian i.i.d. source. (a) Mean mismatch. (b) Variance mismatch.



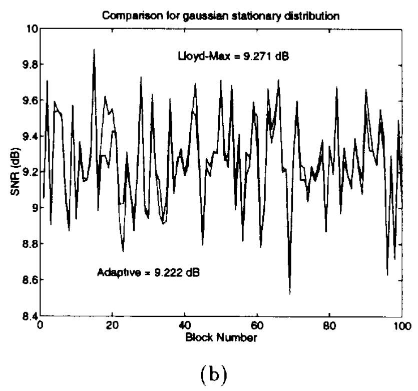
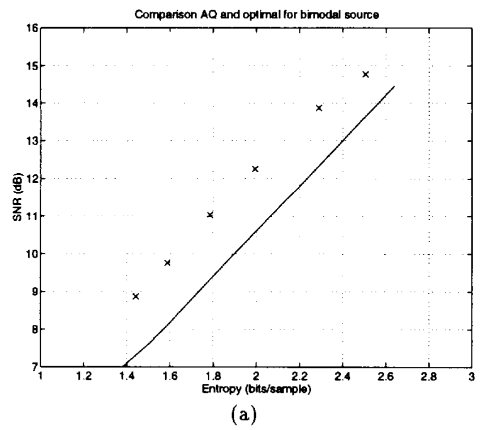


Figure 4: Adaptive vs. Lloyd-Max. The SNR is the average measured over blocks of 2000 samples. (a) Bimodal source (each mode has same variance but different mean). (b) i.i.d. gaussian source.



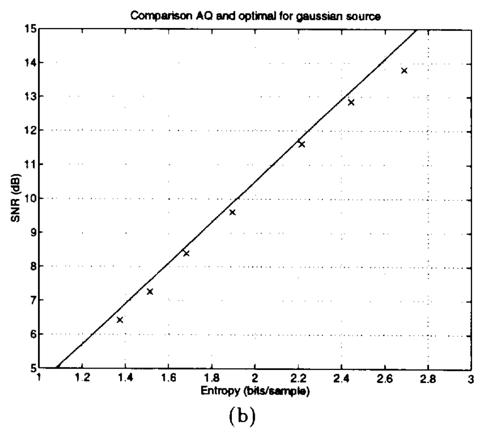


Figure 5: Comparison in the entropy constrained case. The average entropy of the quantizer is used. (a) Bimodal source. (b) i.i.d. gaussian source.