Capacity achieving quantizer design for multiple-input multiple-output thresholding channels

An Vuong*
School of Electrical and Computer Engineering
Oregon State University
Corvallis, OR, 97331

Thuan Nguyen
Department of Electrical and Computing Engineering
Tufts University
Medford, MA, 02148

Thinh Nguyen
School of Electrical and Computer Engineering
Oregon State University
Corvallis, 97331

***Abstract*—We consider a communication channel whose input is modeled as a discrete random variable X with distribution $\mathbf{p}_X$. X is transmitted over a noisy channel and distorted by a continuous-valued noise to result in a continuous-valued output signal U at the receiver. A thresholding quantizer Q is applied to reconstruct a discrete signal $V = Q(U)$ from the continuous-valued U. Our goal is to jointly design both the input distribution $\mathbf{p}_X$ and the thresholding quantizer Q to maximize the mutual information $I(X; V)$ between the input X and V since the accuracy of any decoding algorithm that estimates X from V fundamentally depends on $I(X; V)$. In this paper, an alternating maximization algorithm is proposed that guarantees to achieve a locally optimal solution. In addition, we numerically show that by randomly selecting a set of initial starting points, the proposed algorithm is capable of achieving the globally optimal solution. Both the theoretical and numerical results are provided to justify our approach.**

***Index Terms*—Quantization, mutual information, thresholding quantizer, channel capacity, optimal input distribution.**

I. Introduction

The goal of any communication system is to transmit information reliably and fast over a noisy channel. The capacity of a channel is defined as the fastest rate at which information can be sent between a sender and a receiver with arbitrarily small errors. Mathematically, channel capacity is the maximum value of mutual information between the input and the output of the channel which is a deterministic function of the input distribution and the channel matrix. For a given channel matrix, it is well-known that the mutual information is a concave function of the input distribution, and convex optimization methods or iterative algorithms can be used to determine the channel capacity [1]. Often times, channel capacity is computed for a given channel, i.e. the channel matrix is derived from underlying physical processes or technologies. In other settings, the channel matrix could be designed based application's needs [2]. For instance, if the discrete-input symbols are forwarded over a noisy channel and corrupted by a continuous-valued noise to output a continuous-valued signal, then to reconstruct the transmitted symbols, one usually needs a quantizer to convert these continuous-valued signals back to discrete signals. Selecting a quantizer

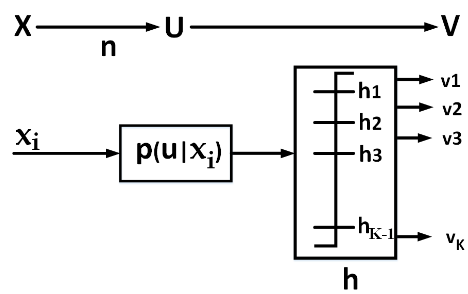

Figure 1: A thresholding channel: the discrete input X having N inputs is transmitted over a noisy channel to produce the continuous output U, a thresholding quantizer having $K-1$ thresholds is used to quantize U back to a discrete output V having K levels.

is equivalent to designing a channel matrix. Therefore, finding optimal quantizers that maximize the mutual information $I(X; V)$ between the input X and V is critically important since the accuracy of any decoding algorithm estimates X from V fundamentally depends on $I(X; V)$ [3]–[9].

Given the fact that both the input distribution as well as the quantizer (channel matrix) are design-able in practice, in this paper, we want to construct an algorithm that jointly finds the optimal input distribution and the quantization scheme that maximizes the mutual information between the channel input and the quantized output, *i.e.*, finding the input distribution and quantizer that achieve the channel capacity. To the best of our knowledge, this problem still remains a hard problem and state-of-the-art methods are only capable to deal with binary input-binary output channels and/or heuristically approximating the locally optimal solution for a larger number of input and output [10]–[12]. Particularly, Nguyen *et al.* [10] proposed a heuristic near-optimal algorithm that alternatively maximizes the mutual information over the input distribution for a given quantizer and minimizes the error rate over the quantization scheme for a fixed input distribution. However, the proposed algorithm in [10] only works well when the signal-to-noise ratio of the channel is high and fails to approximate the global optimal solution if the signal-to-noise ratio is low. Kurkoski and Yagi [11] introduced a near-optimal algorithm to

*Corresponding author: An Vuong, email: *vuonga2@oregonstate.edu*.

determine the optimal value of mutual information for binary input channels with an arbitrary number of the quantized output, however, this algorithm may declare a failure outcome. In [12], the authors proposed an efficient procedure that can simultaneously characterize both the optimal input distribution as well as the quantization scheme for any binary input-binary output channels. Unfortunately, the approach in [12] is not scalable for a higher number of inputs and outputs.

In contrast to the previous works, under a special quantization structure named thresholding quantizer, the contribution of this paper is threefold:

1) We propose an approximation iterative algorithm that guarantees to converge to a locally optimal solution. In practice, we numerically show that the global optimal solution can be obtained by initializing the proposed algorithm from a few random seeds. In addition, our algorithm can be employed with an arbitrary number of inputs and outputs.
2) Our numerical results in Section V give some intuitions about the unsolved problem of maximizing mutual information over both input distribution and quantization scheme. This may be useful for other work in the future.
3) Our well-implemented and easy-to-adapt algorithm can serve as a baseline for future work on tackling the problem of jointly designing the input distribution and quantization scheme for maximizing mutual information. Our implementation is released at this link[1].

The remainder of this paper is structured as follows. In Section II, we formally introduce the notations and define the problem formulation. Section III provides several previous theoretical results which motivate our practical algorithm in Section IV. Numerical results and their analysis are presented in Section V. Discussion and future work can be found in Section VI. Finally, the concluding remarks are presented in Section VII.

II. Problem Formulation

We consider a communication channel as shown in Fig. 1. The input set consists of N discrete transmitted symbols $x_i \in \mathbb{R}$, $x_1 < x_2 < \cdots < x_N$ with the *p.m.f* distribution $\mathbf{p}_X = \{p_1, p_2, \dots, p_N\}$. Due to an additive continuous noise, the received signal $u \in U \in \mathbb{R}$ is modeled via conditional densities $p_{U|X}(u|x_i)$, $i = 1, 2, \dots, N$. We note that $p_{U|X}(u|x_i)$ can have different statistics associated with each transmitted symbol x_i since, in general, different types of noise might be added into different transmitted symbols. To reconstruct the discrete input signal, the receiver uses a thresholding quantizer Q which is equivalent to a threshold vector $\mathbf{h} = \{h_1, h_2, \dots, h_{K-1}\}$ having $K-1$ thresholds $h_1 \leq h_2 \leq \cdots \leq h_{K-1}$, $h_i \in \mathbb{R}$, to quantize the continuous-valued signal U back to a discrete signal $V = \{v_1, v_2, \dots, v_K\}$ following the below rule:

$$Q(u) = v_i, \text{ if } h_{i-1} \leq u < h_i \tag{1}$$

[1]https://github.com/anvuongb/quantization_codes_anon

where $h_0 = -\infty$ and $h_K = +\infty$.

For given conditional densities $p_{U|X}(u|x_i)$, $i = 1, 2, \dots, N$ which completely depend on the noise, our goal is to jointly design the optimal input distribution $\mathbf{p}_X^*$ and the optimal quantizer Q^* (or equivalently, the optimal threshold vector $\mathbf{h}^*$) to maximize the mutual information $I(X;V)$ between the channel input X and the quantized output V. Formally, we want to solve the following optimization problem:

$$\mathbf{h}^*, \mathbf{p}_X^* = \arg\max_{\mathbf{h}, \mathbf{p}_X} I(X;V). \tag{2}$$

It is worth noting that the thresholding quantizer may not be the optimal quantizer for maximizing mutual information. In practice, there may exist other quantizers that group a few non-consecutive regions to form a single quantized output that can achieve higher mutual information [6]. However, since the thresholding quantizer has the most simple structure which is suitable for circuit implementation, in this paper, we limit our consideration to the class of thresholding quantizers.

III. Preliminaries

This section describes some results in the literature that support the development of our practical algorithm in Section IV.

Theorem 1. For a given quantizer Q, *i.e.*, for fixing the threshold vector $\mathbf{h} = \{h_1, h_2, \dots, h_{K-1}\}$, the mutual information between the input and the quantized output $I(X;V)$ is a concave function of the input distribution $\mathbf{p}_X$.

By fixing the quantization scheme, one fixes the conditional distribution between the input and the quantized output of channels, *i.e.*, fixing the channel matrix. Therefore, Theorem 1 directly follows the well-known results in [1] stated that mutual information between the input and output of a channel is a concave function of the input distribution for a given channel matrix. Given this fact, one can just apply the traditional convex optimization algorithms, for example, the CVX package [13] to find the optimal input distribution for a given threshold vector. To make the paper more self-contained, the procedure of computing the channel matrix for a given threshold vector as well as finding the optimal input distribution are described in Appendix A.

In practice, to design the optimal threshold vector, one usually limits the continuous-received signal U into a finite range, *i.e.*, only considering $U \in [-B, B]$ where B is a positive constant, and then discretizes $[-B, B]$ into M discrete intervals. The higher value of M, the more precise the thresholds can be designed.

Theorem 2. For a given input distribution $\mathbf{p}_X$, the global optimal threshold vector $\mathbf{h}^* = \{h_1^*, h_2^*, \dots, h_{K-1}^*\}$ can be found in time-complexity of $O(NKM^2)$ using a dynamic programming algorithm.

This directly follows the results in [14]. Indeed, under a mild assumption that the conditional density $p_{U|X}(u|x_i)$ is log-concave, one can determine the global optimal thresholds in lower time complexity by utilizing the SMAWK algorithm [14]. However, in this paper, we want to propose an algorithm that can handle a general case, therefore, dynamic programming appears as the best approach to quantify the optimal solutions. It is worth noting that without limiting the consideration to a special class of thresholding quantizers, finding the optimal quantizer is an NP-complete problem [15] and determining the global optimal solution for an arbitrary number of input and output is infeasible. Finally, to make the paper self-contained, more details about the dynamic programming algorithm proposed in [14] are described in Appendix A.

IV. Proposed Algorithm

In this section, we propose an approximation algorithm to deal with the optimization problem proposed in equation (2). Our algorithm is motivated by well-established results in Theorem 1 and Theorem 2. Particularly, for a given variable, *i.e.*, fixing the input distribution $\mathbf{p}_X$ or the threshold vector $\mathbf{h}$, Theorem 1 and Theorem 2 offer efficient ways to find the optimal solution for another variable. This leads to a natural idea of employing an alternative optimization algorithm to solve (2). In practice, we start with a random initial guess of one variable (input distribution or thresholds), then use the convex optimization or the dynamic programming algorithm to alternatively obtain the optimal solution for the rest variable. The process is repeated until the mutual information is converged. The pseudo-code of the proposed algorithm can be found in Algorithm 1.

Algorithm 1 Jointly designing input distribution and quantization thresholds for mutual information maximization.

Input: N, K, M, $p_{U|X}(u|x_i)$, $i = 1, 2, \ldots, N$, a small number ϵ that controls the convergence, an initial threshold vector $\mathbf{h}^{t=1} = \{h_1^{t=1}, h_2^{t=1}, \ldots, h_{K-1}^{t=1}\}$.
While $I^t(X;V) - I^{t-1}(X;V) > \epsilon$.
  Step 1:
    Computing the channel matrix A^t corresponding to the threshold vector $\mathbf{h}^t$ at t^{th} loop.
    Computing the optimal input distribution $\mathbf{p}_X^t$ at t^{th} loop.
    Using $\mathbf{p}_X^t$ and A^t to compute $I^t(X;V)$.
  Step 2:
    For given $\mathbf{p}_X^t$, computing the optimal threshold vector $\mathbf{h}^{t+1}$ as the initial threshold for $(t+1)^{th}$ loop.
  $t = t + 1$
Output: $\mathbf{h}^* = \{h_1^*, h_2^*, \ldots, h_{K-1}^*\}$, $\mathbf{p}_X^*$, $I^*(X;V)$.

To make the paper self-contained, more details about Step 1 and Step 2, for example, how the channel matrix A is computed for a given threshold vector in Step 1, and how the dynamic programming is used to find the optimal thresholds in Step 2 can be found in the Appendix A.

Proof of convergence. First, based on Theorem 1 and Theorem 2, for a given variable (input distribution or threshold vector), one is able to find the global optimal solution for another variable. Therefore, the mutual information between the input and the quantized output $I(X;V)$ must increase after performing Step 1 or Step 2 in Algorithm 1. In other words, $I(X;V)^{t-1} \leq I(X;V)^t$, $\forall t$, *i.e.*, $I(X;V)^t$ is an increasing sequence.

Second, we show that $I(X;V)^t$ is upper bounded by a finite value. Indeed,

$$\begin{aligned} I(X;V) &= H(X) - H(X|V) = H(V) - H(V|X) \\ &\leq \max\{H(X), H(V)\} \\ &\leq \max\{\log(N), \log(K)\} \end{aligned}$$

where the first inequality is due to $H(X|V) \geq 0$ and $H(V|X) \geq 0$; the second inequality due to X and V have N and K discrete symbols, respectively.

From the facts that $I(X;V)^t$ produces an increasing sequence and it is upper bounded by a finite number, $I(X;V)^t$ must converge to a stationary point when $t \to +\infty$. It is worth noting that since the optimization problem in (2) is not concave, the solution provided by Algorithm 1 is locally optimal.

V. Numerical Evaluations

A. Experimental setup

To evaluate the performance of Algorithm 1, we provide three examples corresponding to three channels having the number of inputs and outputs are three by three, four by four, and five by five, respectively.

1) Example 1: A three input-three output channel: In this example, we consider a communication channel with input $X = \{x_1 = -3, x_2 = 0, x_3 = 3\}$ which is corrupted by an *i.i.d* Gaussian noise $G(\mu = 0, \sigma = 1)$. Due to the additive property, $p_{U|X}(u|x_1) = N(-3, 1)$, $p_{U|X}(u|x_2) = N(0, 1)$, and $p_{U|X}(u|x_3) = N(3, 1)$. To perform the dynamic programming algorithm, we uniformly discretize $U = [-8, 8]$ into 50 intervals with the same width of 0.32. We use two thresholds h_1, h_2 to quantize the continuous-received signal U back to a discrete signal V having three levels.

2) Example 2: A four input-four output channel: In this example, we consider a communication channel with input $X = \{x_1 = -3, x_2 = -1, x_3 = 1, x_4 = 3\}$ which is corrupted by an *i.i.d* Gaussian noise $G(\mu = 0, \sigma = 1)$. Due to the additive property, $p_{U|X}(u|x_1) = N(-3, 1)$, $p_{U|X}(u|x_2) = N(-1, 1)$, $p_{U|X}(u|x_3) = N(1, 1)$, and $p_{U|X}(u|x_4) = N(3, 1)$. To perform the dynamic programming algorithm, we uniformly discretize $U = [-8, 8]$ into 50 intervals with the same width of 0.32. We use three thresholds h_1, h_2, h_3 to quantize the continuous-received signal U back to a discrete signal V having four levels.

3) Example 3: A five input-five output channel: In this example, we consider a communication channel with input $X = \{x_1 = -4, x_2 = -2, x_3 = 0, x_4 = 2, x_5 = 4\}$ which is corrupted by an *i.i.d* Gaussian noise $G(\mu = 0, \sigma = 1)$. Due to the additive property, $p_{U|X}(u|x_1) = N(-4, 1)$, $p_{U|X}(u|x_2) = N(-2, 1)$, $p_{U|X}(u|x_3) = N(0, 1)$, $p_{U|X}(u|x_4) = N(2, 1)$, and $p_{U|X}(u|x_5) = N(4, 1)$. To perform the dynamic programming algorithm, we uniformly discretize $U = [-8, 8]$ into 50 intervals with the same length of 0.32. We use four

Running time	Algorithm 1	Exhaustive search	Speedup ratio
Example 1	1.85	31.44	16.99
Example 2	2.97	1263.27	425.34
Example 3	7.46	15313.25	2052.71

Table I: Running time (in second) of Algorithm 1 *vs*. Exhaustive search algorithm.

thresholds h_1, h_2, h_3, h_4 to quantize the continuous-received signal U back to a discrete signal V having five levels.

Finally, to evaluate the performance of the proposed algorithm, the true channel capacity of each channel is also computed by exhaustive searching. Particularly, we first exhaustively search for all possible combinations of thresholds to compute all possible channel matrices. For example, with the three input-three output channels having two thresholds h_1, h_2, we compute the channel matrix for all possible combinations of $h_1 \in [-8, 8]$ and $h_2 \in (h_1, 8]$ with a resolution of 0.32 ($M = 50$). For a given channel matrix, a convex optimization algorithm is deployed to compute the optimal input distribution and its corresponding mutual information. After exhaustively searching over all possible channel matrices, the truly optimal mutual information (channel capacity) and its corresponding optimal thresholds and input distribution are recorded.

B. Numerical analysis

Since Algorithm 1 is only able to produce a locally optimal solution, we run Algorithm 1 from multiple random-selected threshold vectors. On average, we find that it only takes seven random seeds to reach the global maximum values of mutual information which are 1.1289, 1.1314, and 1.4336 for channels in Example 1, Example 2, and Example 3, respectively. These maximum values are matched with the results produced by the exhausted search algorithm. As reported in Table I, the running time of the exhaustive search is exponentially higher than the running time of Algorithm 1 with a single random seed. Therefore, even with multiple initializations, Algorithm 1 is still orders of magnitude faster than the exhaustive search algorithm.

Fig. 2, Fig. 3, and Fig. 4 illustrate the performance of Algorithm 1 across 100 different initial starting points for the channels in Example 1, Example 2, and Example 3, respectively. As seen, the algorithm converges to a stationary solution after at most eight iterations, and most runs terminated after only five iterations.

From our experiments, it is worth noting that the mutual information has multiple local stationary points, for example, the channel in Example 2 has three local optimal solutions. Therefore, the optimization problem in equation (2) is certainly non-convex which indicates the necessity of requiring multiple starting points in Algorithm 1.

Finally, from our numerical results, it is interesting to observe the following facts:

1) If the signal-to-noise ratio is high, then a special initial threshold which is the average of two consecutive input symbols could lead to the global optimal solution. For example, if the additive Gaussian distribution is $G(\mu = 0, \sigma = 0.5)$, then by starting from $h_i = (x_i + x_{i+1})/2$, one can obtain the optimal solution in Example 1, 2, and 3. However, when the signal-to-noise ratio is low, this observation may not be true. For example, if the additive Gaussian distribution is $G(\mu = 0, \sigma = 3)$, then by starting from $h_i = (x_i + x_{i+1})/2$, the optimal mutual information in Example 1, 2, and 3 are 0.3913, 0.4095, and 0.5735 which are lower than the corresponding optimal solutions of 0.3914, 0.4119, and 0.5740 provided by the exhaustive search.

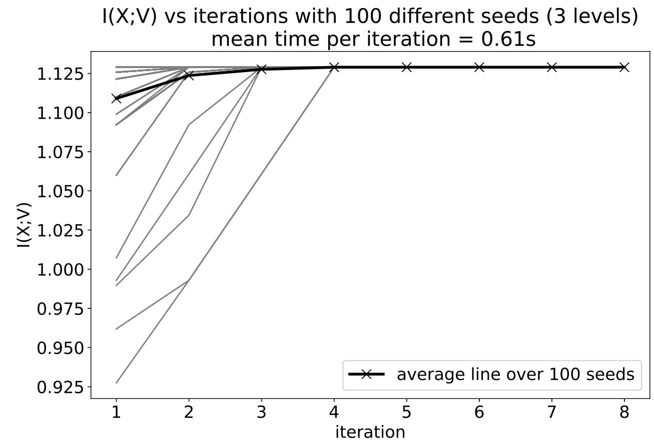

Figure 2: Performance of Algorithm 1 for the channel in Example 1: $I(X;V)$ *vs*. iterations across 100 different initial points. Within a few iterations and random seeds, Algorithm 1 achieves the global maximum value of 1.1289.

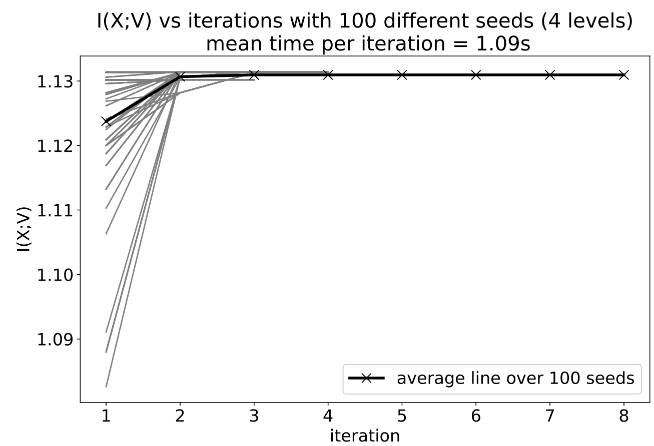

Figure 3: Performance of Algorithm 1 for the channel in Example 2: $I(X;V)$ *vs*. iterations across 100 different initial points. Within a few iterations and random seeds, Algorithm 1 achieves the global maximum value of 1.1314.

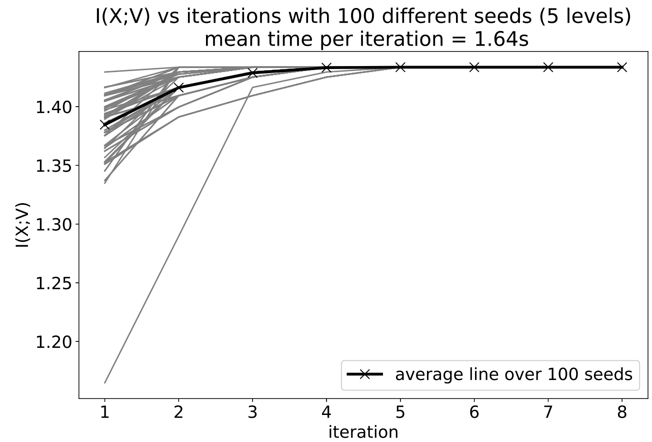

Figure 4: Performance of Algorithm 1 for the channel in Example 3: $I(X;V)$ *vs.* iterations across 100 different initial points. Within a few iterations and random seeds, Algorithm 1 achieves the global maximum value of 1.4336.

2) If the input X and the conditional densities $p_{U|X}(u|x_i)$ are symmetric, then the optimal input distribution and thresholds are also symmetric. For example, the optimal input distribution and thresholds in Example 1 are $[0.3817, 0.2366, 0.3817]$ and $[-1.36, 1.36]$, the optimal input distribution and thresholds in Example 2 are $[0.3616, 0.1384, 0.1384, 0.3616]$ and $[-1.76, 0, 1.76]$, and the optimal input distribution and thresholds in Example 3 are $[0.3437, 10^{-6}, 0.3126, 10^{-6}, 0.3437]$ and $[-2.4, -1.52, 1.52, 2.4]$.
3) It is always true that if the noise is identical between symbols and the symbols are allocated uniformly, *i.e.*, $x_i - x_{i-1} = \Delta$, $\forall i$, where Δ is a constant, then the global optimal solution will allocate a higher distribution to the most left and the most right symbol. This is, indeed, reasonable since these symbols are less noisy than other ones in middle.

Our implementation can be found at this link[2].

VI. Discussion and future work

From our numerical results, it is clear that the problem of jointly maximizing the mutual information over both input distribution and quantization scheme has multiple local solutions and it is not a concave maximization problem. Even though using multiple starting points may alleviate this problem and enable us to approximately achieve the global optimal solution, it is still desirable for developing a new algorithm that can deterministically output the truly global optimum. Some open questions will be investigated in our future work: (a) it is well-known that finding the optimal quantizer is an NP-complete problem [15], can we theoretically characterize the hardness of the optimization problem proposed in equation (2)? (b) can we design a polynomial algorithm for deterministically determining the global solution over both input distribution and quantization scheme?

[2]https://github.com/anvuongb/quantization_codes_anon

VII. Conclusion

In this paper, we mathematically formulate the problem of jointly maximizing the mutual information over both the input distribution and the quantization scheme variables. An alternative algorithm is proposed that can guarantee a local optimum for any thresholding channel. In practice, by running the proposed algorithm from multiple starting points, the global optimal solution can be approximately achieved. Our future research will include: characterizing the hardness of the proposed problem as well as designing more efficient algorithms that can capture the global optimal solution.

Acknowledgments

The authors would like to acknowledge Nam Nguyen for proofreading the article.

Appendix

A. Finding the optimal input distribution for a given threshold vector

For a given threshold vector $\mathbf{h} = \{h_1, h_2, \dots, h_{K-1}\}$, Step 1 in Algorithm 1 aims to find the channel matrix and then using a convex optimization algorithm, for example, the CVX package [13] to find the optimal input distribution. To shorten the notation, we define $\phi_i(u) = p_{U|X}(u|x_i)$. Due to $Q(u) = v_i$, if $h_{i-1} \leq u < h_i$ or $v_i = [h_{i-1}, h_i)$. By definition, the channel matrix A is a $N \times K$ matrix where:

$$A_{ij} = p(v_j|x_i) = \int_{u=h_{j-1}}^{h_j} \phi_i(u)du. \tag{3}$$

Recall that $\mathbf{p}_X = \{p_1, p_2, \dots, p_N\}$ denotes the input distribution vector, we have:

$$\begin{aligned} I(X;V) &= H(V) - H(V|X) \\ &= -\sum_{j=1}^{K} (A^T\mathbf{p}_X)_j \log (A^T\mathbf{p}_X)_j + \sum_{i=1}^{N}\sum_{j=1}^{K} p_i A_{ij} \log A_{ij} \end{aligned} \tag{4}$$

where T denotes the transpose operation, $(A^T\mathbf{p}_X)_j$ denotes the j^{th} component of the output distribution vector $A^T\mathbf{p}_X$. Next, the CVX package [13] can be used to solve the following convex optimization problem:

Minimize:

$$\sum_{j=1}^{K} (A^T\mathbf{p}_X)_j \log (A^T\mathbf{p}_X)_j - \sum_{i=1}^{N}\sum_{j=1}^{K} p_i A_{ij} \log A_{ij}, \tag{5}$$

subject to:

$$\begin{cases} \mathbf{p}_X \succeq \mathbf{0} \\ \mathbf{1}^T\mathbf{p}_X = 1. \end{cases}$$

By plugging the optimal input distribution back into (4), the corresponding mutual information is produced.

B. Finding the optimal threshold vector for a given input distribution

Step 2 in Algorithm 1 employs a dynamic programming algorithm to find the optimal threshold vector. Since $I(X;V) = H(X) - H(X|V)$ and the input distribution $\mathbf{p}_X$ is fixed, then designing the optimal thresholds for maximizing $I(X;V)$ is equivalent to finding the thresholds for minimizing $H(X|V)$. To apply the dynamic programming algorithm, we first discretize U into M intervals $\{u_1, u_2, \ldots, u_M\}$. Follow [14], we define $D(i,j,k)$ as the minimum (optimal) value of $H(X|V)$ by clustering $(u_i, u_j]$ into k clusters where $0 \leq i \leq j \leq M$ and $0 \leq k \leq K$. The goal is to find $D(1,M,K)$.

Recall that $\mathbf{p}_X = \{p_1, p_2, \ldots, p_N\}$ denotes the input distribution vector and $\phi_i(u) = p_{U|X}(u|x_i)$. Let $\mathbf{q}_V = \{q_1, q_2, \ldots, q_K\}$ denote the quantized output distribution vector, then $H(X|V)$ can be computed by:

$$H(X|V) = -\sum_{j=1}^{K} q_j \sum_{i=1}^{N} p(x_i|v_j)\log(p(x_i|v_j)) \tag{6}$$

where:

$$p(v_j|x_i) = \int_{u=h_{j-1}}^{h_j} \phi_i(u)du, \tag{7}$$

$$q_j = \sum_{i=1}^{N} p_i p(v_j|x_i). \tag{8}$$

$$p(x_i|v_j) = \frac{p_i p(v_j|x_i)}{\sum_{i=1}^{N} p_i p(v_j|x_i)}, \tag{9}$$

The dynamic programming algorithm is based on the following recursion:

$$D(1,j,k) = \min_{1 \leq q \leq j-1} D(1,q,k-1) + D(q+1,j,1). \tag{10}$$

By definition, $D(1,j,k) = 0$ if $j = 0$ or $k = 0$. From these initial values, using (10), one can compute $D(1,j,k)$, $\forall j,k$. The desired optimal solution is $D(1,M,K)$. To track the optimal threshold, one can use the backtracking method, for example, by storing the index that results in the optimal value in each recursion step. Particularly, define:

$$H_k(j) = \arg\min_{q} D(1,q,k-1) + D(q+1,j,1). \tag{11}$$

Then, $H_k(j)$ saves the position of $k-1^{th}$ threshold. By starting from $h_K^* = M$, for each $i = \{K-1, K-2, \ldots, 1\}$, all of the other optimal thresholds can be found by backtracking:

$$h_i^* = H_{i+1}(h_{i+1}^*). \tag{12}$$

The pseudo-code of the dynamic programming can be viewed in Algorithm 2. Compare to the original one in [14], Algorithm 2 does not involve the constraint on the output distribution and some typos have been fixed. Finally, the optimal mutual information for given $\mathbf{p}_X$ is:

$$\begin{aligned} I^*(X;V) &= H(X) - H^*(X|V) \\ &= -\sum_{i=1}^{N} p_i \log(p_i) - D(1,M,K). \end{aligned} \tag{13}$$

Algorithm 2 Dynamic programming to find $D(1,M,K)$

Input: $\mathbf{p}_X$, N, K, M, $p_{U|X}(u|x_i)$, $i = 1,2,\ldots,N$.
Initialization: $D(1,j,k) = 0$ for $\forall j = 0$ or $k = 0$.
Recursion step:
  For $k = 1,2,\ldots,K$
    For $j = 1,2,\ldots,M$

$$D(1,j,k) = \min_{i \leq q \leq j-1} D(1,q,k-1) + D(q+1,j,1).$$

$$H_k(j) = \arg\min_{q} D(1,q,k-1) + D(q+1,j,1).$$

    End For
  End For
Backtracking step: Let $h_K^* = M$, for $i = \{K-1, K-2, \ldots, 1\}$: $h_i^* = H_{i+1}(h_{i+1}^*)$.
Output: $D(1,M,K)$, $\mathbf{h}^* = \{h_1^*, h_2^*, \ldots, h_{K-1}^*\}$.

References

[1] Thomas M Cover and Joy A Thomas. *Elements of information theory*. John Wiley & Sons, 2012.

[2] Jiadong Wang, Thomas Courtade, Hari Shankar, and Richard D Wesel. Soft information for ldpc decoding in flash: Mutual-information optimized quantization. In *Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE*, pages 1–6. IEEE, 2011.

[3] Jaspreet Singh, Onkar Dabeer, and Upamanyu Madhow. On the limits of communication with low-precision analog-to-digital conversion at the receiver. *IEEE Transactions on Communications*, 57(12), 2009.

[4] Brian M Kurkoski and Hideki Yagi. Quantization of binary-input discrete memoryless channels. *IEEE Transactions on Information Theory*, 60(8):4544–4552, 2014.

[5] Thuan Nguyen and Thinh Nguyen. Structure of optimal quantizer for binary-input continuous-output channels with output constraints. In *2020 IEEE International Symposium on Information Theory (ISIT)*, pages 1450–1455. IEEE, 2020.

[6] Thuan Nguyen and Thinh Nguyen. Optimal quantizer structure for maximizing mutual information under constraints. *IEEE Transactions on Communications*, 69(11):7406–7413, 2021.

[7] Rudolf Mathar and Meik Dörpinghaus. Threshold optimization for capacity-achieving discrete input one-bit output quantization. In *Information Theory Proceedings (ISIT), 2013 IEEE International Symposium on*, pages 1999–2003. IEEE, 2013.

[8] Berkan Dulek. On mutual information-based optimal quantizer design. *IEEE Communications Letters*, 26(5):1008–1011, 2022.

[9] Thuan Duc Nguyen and Thinh Nguyen. On binary quantizer for maximizing mutual information. *IEEE Transactions on Communications*, 68(9):5435–5445, 2020.

[10] Thuan Nguyen, Yu-Jung Chu, and Thinh Nguyen. On the capacities of discrete memoryless thresholding channels. In *2018 IEEE 87th Vehicular Technology Conference (VTC Spring)*, pages 1–5. IEEE, 2018.

[11] Brian M Kurkoski and Hideki Yagi. Finding the capacity of a quantized binary-input dmc. In *2012 IEEE International Symposium on Information Theory Proceedings*, pages 686–690. IEEE, 2012.

[12] Thuan Nguyen and Thinh Nguyen. Capacity achieving quantizer design for binary channels. *IEEE Communications Letters*, 25(3):759–763, 2020.

[13] Michael Grant and Stephen Boyd. Cvx: Matlab software for disciplined convex programming, version 2.1, 2014.

[14] Thuan Nguyen and Thinh Nguyen. Thresholding quantizer design for mutual information maximization under output constraint. In *2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring)*, pages 1–5. IEEE, 2020.

[15] Brendan Mumey and Tomáš Gedeon. Optimal mutual information quantization is np-complete. In *Neural Information Coding (NIC) workshop poster, Snowbird UT*, pages 1932–4553, 2003.