A New Fast Algorithm for Finding Capacity of Discrete Memoryless Thresholding Channels

Thuan Nguyen
School of Electrical and
Computer Engineering
Oregon State University
Corvallis, OR, 97331
Email: nguyeth9@oregonstate.edu

Yu-Jung Chu
School of Electrical and
Computer Engineering
Oregon State University
Corvallis, OR, 97331
Email: chuy@oregonstate.edu

Thinh Nguyen
School of Electrical and
Computer Engineering
Oregon State University
Corvallis, 97331
Email: thinhq@eecs.oregonstate.edu

***Abstract*—This paper proposes an efficient algorithm for finding the channel capacity of discrete memoryless thresholding channels (DMTCs) that are typically used in Pulse Amplitude Modulation (PAM). While there are efficient algorithms for determining capacity of a discrete memoryless channel (DMC), it is difficult to obtain the capacity of a DMTC. Unlike a typical DMC channel whose the capacity is a convex function of the input distribution, the capacity of a DMTC is a non-convex function of both the input distribution and decision thresholds. To resolve this problem, we propose an efficient algorithm for approximating the channel capacity of a DMTC using a novel modified k-means algorithm whose computational complexity is reduced by a factor of $\frac{M}{\log M}$ over the standard k-means algorithm, where M relates to the precision resolution of the solution. Both theoretical and numerical results are provided to verify the proposed algorithm.**

Keyword: channel quantization, algorithm, mutual information, threshold, partition, optimization.

I. Introduction

A communication system can be modeled by an abstract channel with a set of inputs at the transmitter and a set of corresponding outputs at the receiver. Often times the transmitted symbols (inputs) are different from the receiving symbols (outputs), i.e., errors occur due to many factors such as the physics of signal propagation through a medium or thermal noise. Thus, the goal of a communication system is to transmit the information reliably at the fastest rate. The fastest achievable rate with vanishing error for a given channel is defined by its channel capacity which is the maximum mutual information between input and output random variables. For an arbitrary discrete memoryless channel (DMC) that is specified by a given channel matrix, the mutual information is a concave function of the input probability mass function [1]. Thus, many efficient algorithms/closed-form expressions exist to find the channel capacity of DMC [2], [3]. On the other hand, in many real-world scenario, the channel matrix is not given. Rather, the channel matrix is designed under the consideration of many factors such as power consumption, encoding/decoding speeds, and so on. As a result, the mutual information is no longer a concave function of the input distribution alone, but is a possibly non-concave/convex function

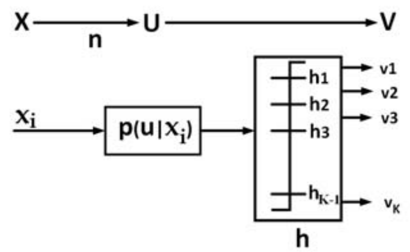

Figure 1: A DMTC K inputs and K outputs using $K-1$ thresholds.

in both input distribution and the parameters of the channel matrix.

In fact, many real-world communication scenarios can be modeled as a channel with discrete inputs, additive continuous noise, the discrete outputs as a result of quantizing the sum of continuous noise and discrete inputs. In such cases, each quantization scheme produces a different channel matrix which ultimately determines the channel capacity. Thus, designing an optimal quantizer is critical. Many quantizers are based on some intuitive objectives such as minimizing the MSE distortion and error rate [4] or maximizing the mutual information (capacity) between the inputs and outputs [5], [6]. Recently, designing quantizers that maximize the mutual information is also important in the design of Polar code and LDPC code decoders [7]. We also note that these quantizers assume the input distribution is given and the output is discrete while only in [8] and [9], the quantization of continuous output channel is investigated. Specifically, only in [9], a near optimal algorithm is constructed for maximizing the mutual information of continuous output channel as a function of both quantizer and input distribution.

Discrete Memoryless Thresholding Channel. One important class of quantizer is the thresholding quantizer which maps a continuous value $u \in \mathbb{R}$ to a discrete value v such that every distinct v corresponds to one single continuous interval covering u. Such quantizer is suitable for the decoder that uses Pulse Amplitude Modulation (PAM) in which the output symbols are recovered based on the magnitude of received signals. Fig. 1 shows the setup of a discrete memoryless thresholding channel (DMTC). In this setup, the input $x_i \in X = \{x_1, \dots, x_K\}$ are assumed to be discrete. The received signal $u \in \mathbb{R}$ is drawn from the conditional density $p_{U|X}(u|x_i)$. $p_{U|X}(u|x_i)$ models the effect of noise. The noise can have different characteristics for different transmitted signals. u is then quantized into v_i, $v_i \in V = \{v_1, \dots, v_K\}$ as the output using a quantizer Q:

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

with $h_0 = -\infty$ and $h_K = +\infty$. A quantizer $Q(.)$, therefore, is equivalent to a threshold vector $h = \{h_1, \dots, h_{K-1}\}$.

The capacity is found by selecting the optimal input p.m.f p_X^* and the optimal thresholds h^* that maximize the mutual information $I(X;V)$. To our knowledge this problem is difficult and not well-studied [8].

In this paper, we propose an efficient algorithm for approximating the channel capacity of this DMTC using a novel modified k-means algorithm with the Kullback-Leibler (KL) metric. We show that the computational complexity of the proposed algorithm is reduced by factor of $\dfrac{M}{\log M}$ over the standard k-means algorithm.

II. Problem Formulation and Solution Approach

Fig. 1 shows the setup of a DMTC. The input set consists of K discrete transmitted symbols $x_1 < x_2 < \cdots < x_K$. Due to a continuous noise, the received signal is $u \in \mathbb{R}$ and is modeled via the conditional density $p_{U|X}(u|x_i)$. We note that $p_{U|X}(u|x_i)$ can have different statistics associated with each transmitted signals x_i. In the special case where $u_i = x_i + n_i$ with n_i's are i.i.d, then $p_{U|X}(u|x_i)$ is simply a shifted version of $p_{U|X}(u|x_j), \forall i, j$. u is then quantized into K discrete outputs $v_i \in V = \{v_1, \dots, v_K\}$. Since the thresholding quantization is used, there are $K-1$ thresholds $h_1 \leq h_2 \leq \cdots \leq h_{K-1}$. The channel capacity is the maximum mutual information, therefore it is a solution to the following optimization problem:

$$C = \max_{p_X, Q} I(X;V), \tag{2}$$

where p_X is the input p.m.f and $Q(.) = h = \{h_1, \dots, h_{K-1}\}$ is the quantizer with,

$$Q(u) = v_i, \text{ if } h_{i-1} \leq u < h_i. \tag{3}$$

We note that this type of quantizer is not optimal in general. Depending on the characteristics of noise, for a DMTC with a given K output symbols, the optimal quantizer might consists more than $K-1$ thresholds. The reason is that there might be multiple distinct intervals consisting u that maps to a v_i. On the other hand, it can be shown that this thresholding quantizer structure is actually optimal for many real-world scenarios [6]. In this paper, we will focus on finding capacity for this thresholding quantizer.

Unfortunately, $I(X;V)$ is not a convex/concave function in p_X and h, making this problem difficult to solve. To this end, we propose the Algorithm 1 that efficiently finds the capacity by solving two simpler sub-problems iteratively. The main idea is to use the alternating direction algorithm that (1) maximizes $I(X;V)$ with respect to p_X while keeping h fixed, which then (2) maximizes $I(X;V)$ with respect to h while given p_X. The steps (1) and (2) are repeated until a convergence is reached. In particular, let $I(X;V)$ be written as a function $I(p_X, h)$ of p_X and h, then sub-problem 1 is:

P1: **Maximize:**

$$I(p_X, h)$$

s.t:

$$\mathbf{0} \preceq p_X \preceq \mathbf{1},$$

$$\mathbf{1}^T p_X = 1.$$

P1 is a constrained convex optimization problem since for a fixed h, $I(p_X, h)$ is a concave function [1], [2], [3], and can be solved efficiently using gradient descent algorithms [1]. The constraints are imposed to make p_X is a valid p.m.f.

The sub-problem 2 is:

P2: **Maximize:**

$$I(p_X, h)$$

s.t:

$$h_i \leq h_{i+1}, i = 1, 2, \dots, K-2.$$

In solving **P2**, p_X is fixed while h is the optimization variable. In general, **P2** is not as easy to solve as **P1** since $I(p_X, h)$ is not a convex/concave function in relation to h. However, there are methods for solving **P2** effectively, e.g., using the k-means algorithm. In this paper, we propose a modified version of k-means algorithm that takes advantage of the structure of the noise to speed up the time complexity significantly. Also, since **P1** is readily solved, the rest of the paper will be focused on the algorithm and analysis for solving **P2** using the modified k-means algorithm.

Algorithm 1 Maximizing the mutual information. ϵ is a small given value for convergence; t is the iteration.

```
1: t = 0, E = 1                      ▷ E is any number larger than ε
2: Choose initial random vectors p_X^(t) and h^(t)
3: While E > ε
4:   p_X^(t+1) = argmax_{p_X} I(p_X, h^(t))            ▷ fixed h^(t)
5:   h^(t+1) = argmax_h I(p_X^(t+1), h)               ▷ fixed p_X^(t+1)
6:   E = |I(p_X^(t+1), h^(t+1)) − I(p_X^(t), h^(t))|
7:   t = t + 1
8: End
9: return I(p_X^(t), h^(t))
```

III. Mutual Information Maximization Given p_X

We discuss the algorithm and its analysis for the sub-problem **P2**. The proposed algorithm is based on the modified k-means algorithm using KL-divergence as the distance metric. Before discussing the rationale for using the KL-divergence and the k-means algorithm, let us define the notations.

Notation. Let x_u denote the conditional pmf:

$$x_u = (p_{X|U}(x{=}x_1|u), p_{X|U}(x{=}x_2|u), \dots, p_{X|U}(x{=}x_K|u)),$$

where

$$p_{X|U}(x|u) = \frac{p_{U|X}(u|x)p_X(x)}{\sum_{x' \in X} p_{X|U}(u|x')p_X(x')}.$$

Define

$$\phi_i(u) = p_{U|X}(u|x_i)$$

as the conditional noise density of u given the transmitted signal x_i, then

$$x_u = [\frac{p_1\phi_1(u)}{\sum_{j=1}^{K} p_j\phi_j(u)}, \frac{p_2\phi_2(u)}{\sum_{j=1}^{K} p_j\phi_j(u)}, \dots, \frac{p_K\phi_K(u)}{\sum_{j=1}^{K} p_j\phi_j(u)}], \quad (4)$$

where p_j denotes j^{th} vector component of p_X.

Similarly, let x_{v_i} denote the conditional p.m.f

$$x_{v_i} = (p_{X|V}(x{=}x_1|v{=}v_i), \dots, p_{X|V}(x{=}x_K|v{=}v_i)), \quad (5)$$

where

$$p_{X|V}(x|v) = \frac{p_{V|X}(v|x)p_X(x)}{\sum_{x'} p_{X|V}(V|x')p_X(x')}.$$

Also,

$$p_{V|X}(v|x) = \int_{u:Q(u)=v} p_{U|X}(u|x).$$

A. k-means Algorithm With KL-Divergence Using A General Quantizer

In this section, we provide justification for using k-means algorithm with KL-divergence for maximizing $I(X;V)$.

Kullback-Leibler Divergence. KL divergence of two probability vectors $a = (a_1, a_2, \dots, a_J)$ and $b = (b_1, b_2, \dots, b_J)$ of the same outcome set is defined as

$$D(a||b) = \sum_{i=1}^{J} a_i \log(\frac{a_i}{b_i}). \quad (6)$$

For a given u and a given quantizer that produces $v_i = Q(u)$, the KL-divergence between the conditional pmfs x_u and x_{v_i} is denoted as $D(x_u||x_{v_i})$. If the expectation is taken over U, i.e., over the noise distribution, and $V = Q(U)$ for any quantizer, then from Lemma 1 [5], we have:

$$\begin{aligned} \mathbb{E}_U[D(x_u||x_{v_i})] &= H(X|V) - H(X|U) \\ &= I(X;U) - I(X;V). \end{aligned}$$

Since for a fixed p_X and the conditional noise density $\phi_i(x)$, $I(X;U)$ is fixed and independent of the quantizer Q. Thus, maximizing $I(X;V)$ over Q is equivalent to minimizing $\mathbb{E}_U[D(x_u||x_{v_i})]$ with optimal quantizer:

$$Q^* = \min_{Q} \mathbb{E}_U[D(x_u||x_{v_i})]. \quad (7)$$

k-means Algorithm. Now, the optimal Q^* in Eq. (7) can be found effectively using a k-means algorithm. A generic k-means algorithm is conceptually a clustering algorithm that classifies a cloud of M discrete points into K clusters (sets) $C_1, C_2, \dots, C_K$ such that if a point $x \in C_i$ then $d(x, C_i) \le d(x, C_j), \forall j$, where $d(x, C)$ denotes some distance metric of the point x to the set C. Often, $d(x, C)$ is defined as the distance from x to the centroid of the set C. k-means algorithm works as follows. First, K centroids are randomly selected. Next, each point x is assigned to the a cluster whose centroid is closest the point x. After all the points have been assigned, the centroids of each cluster are recomputed. The assignment of the points to the clusters then starts again with the new centroids, then new centroids based on the new assignment are again recomputed. The process keeps on until there is no change in cluster membership.

Since the k-means algorithm minimizes the sum of distances of every point to its centroid, therefore, based on our discussion of KL divergence above, to find Q^* we can treat x_u (function of u) as a K-dimensional point, x_{v_i} as the centroid of the cluster C_i, and $D(x_u||x_{v_i})$ as the distance of the point x_u to to the centroid x_{v_i}.

Because u is a continuous value and k-means algorithm is used only for discrete points, we first discretize the range of u into M bins of equal width ϵ, with $u_1, u_2, \dots, u_M$ as the center values. The k-means algorithm finds $v_i = Q^*(u_j)$, for $\forall\ j = 1, 2, \dots, M$, $i = 1, 2, \dots, K$. Note that this is a general quantizer which requires a lookup table consisting of M entries. If the actual u value is not in one of the u_i then we pick the nearest u_i to feed to the quantizer. As seen, the larger M results in a better approximation, but also results in a larger lookup table for $Q^*(u_j)$. Algorithm 2 shows the k-means algorithm using KL divergence as the distance metric.

We note that the Cluster assignment step (Line 4) is similar to classical k-means algorithm, however, the distance metric is KL divergence. In the Computing centroids step (Line 5), the centroids are updated using Eq. (9). The proof can be viewed in Proposition 1 of [10] for all the Bregman divergences whose special case is the KL divergence. We note that the proof in [10] is for discrete domain, however, a similar result can be easily established for continuous domain. Due to the limitation of space, please see the proof in our extension version.

Computational Complexity of Algorithm 2. As described, there are two main iterative operations in a generic k-means algorithm to cluster M points into K clusters, namely, the computing centroid operation and the cluster assignment operation. Since each point in Algorithm 2 is a vector of K-dimensional space, the computational complexities of the computing centroid operation is $O(KM)$ and of the cluster assignment operation is $O(K^2M)$. Since, the cluster assignment operation is most expensive, the overall complexity of the k-means algorithm is $O(K^2MT) = O(\frac{K^2}{\epsilon}T)$ where ϵ is the width of the discretized interval, i.e., the precision of the solution, and T is the number of iterations. In the new k-means algorithm using a thresholding quantizer (rather than a general quantizer), we show that the complexity is significantly reduced to $O(TK^2 \log M)$.

B. k-means Algorithm with KL Divergence Using Thresholding Quantizer

In this section, we present a modified version of the k-means algorithm in III-A using a thresholding quantizer. By taking advantage of the structure of a thresholding quantizer,

Algorithm 2 k-means algorithm with KL divergence as a distance metric

1: **Input**: p_X, $\phi_i(u)$, K, M.
2: **Ouput**: Cluster C_j, $j = 1, 2, \ldots, K$. Given C_j, the quantizer $Q(u_i) \to v_j$ if $x_{u_i} \in C_j$.
3: **Initialization**: Discretize u into to M values: $u_1, u_2, \ldots, u_M$, and compute x_{u_i} for each u_i. Pick an arbitrary quantizer $Q(.)$, i.e., pick K arbitrary x_{v_j}, $j = 1, 2, \ldots K$.
4: **Step 1 (Cluster assignment)**: Cluster $x_{u_1}, x_{u_2}, \ldots, x_{u_M}$ into one of the cluster C_j with the centroid x_{v_j}.

$$C_j = \{u_i | D(x_{u_i} || x_{v_j}) \leq D(x_{u_i} || x_{v_s}), \forall s \neq j\}, \forall j. \quad (8)$$

5: **Step 2 (Computing centroids)**: Computing centroids for each cluster C_j:

$$(x_{v_j})_i = \frac{\int_{C_j} (p_X)_i \phi_i(u) du}{\sum_{k=1}^{K} \int_{C_j} (p_X)_k \phi_k(u) du}, \forall i, j, \quad (9)$$

where $(p_X)_i$ and $(x_{v_j})_i$ denote i^{th} component of p_X and x_{v_j}.
6: **Step 3**: Go to Step 1 until all clusters stop changing or the maximum number of iterations is reached.

Algorithm 3 k-means Algorithm Using Thresholding Quantizer

1: **Input**: p_X, $\phi_i(u) = p_{U|X}(u|x_i)$, K, M.
2: **Output**: $h = \{h_1, h_2, \ldots, h_{K-1}\}$
3: **Initialization**: Randomly choose $K-1$ thresholds $h_k \in \mathbb{R}$ for $k \in \{1, 2, \ldots, K-1\}$ such that $h_k < h_{k+1}$.
4: **Step 1 (Computing Centroids)**: Updating centroids for each cluster:

$$x_{v_k} = [(x_{v_k})_1, (x_{v_k})_2, \ldots, (x_{v_k})_K], \quad (11)$$

$$(x^*_{v_k})_i = \frac{\int_{h_{k-1}}^{h_k} p_i \phi_i(u) du}{\sum_{j=1}^{K} \int_{h_{k-1}}^{h_k} p_j \phi_j(u) du}. \quad (12)$$

5: **Step 2 (Updating Thresholds)**
For every $k = 1, 2, \ldots K-1$, find u_k such that:

$$D(x_{u_k} || x_{v_{k-1}}) = D(x_{u_k} || x_{v_k}), \quad (13)$$
$$h_{k-1} = u_k.$$

6: **Step 3**: Go to Step 1 until all h_k's stop changing or the maximum number of iterations is reached.

the new algorithm has a much lower complexity. Unlike a general quantizer that needs a lookup table consisting of M entries, a thresholding quantizer has only $K-1$ thresholds $h = \{-\infty, h_1, \ldots, h_{K-1}, +\infty\}$ that follows Eq. (3) which decodes a continuous value u to K contiguous regions such that if $u \in (h_{k-1}, h_k)$, the receiver will decode u to $v_k \in V = \{v_1, v_2, \ldots, v_K\}$ [9]. Therefore, a thresholding quantizer lends itself to a simple circuit implementation consisting of a few comparators. That said, the optimal thresholding quantizer is:

$$h^* = \underset{h}{\operatorname{argmax}}\, I(X; V). \quad (10)$$

Algorithm 3 finds $h^* = \{h^*_1, h^*_2, \ldots, h^*_{K-1}\}$.

Similar to Algorithm 2, Algorithm 3 is a k-means algorithm with a twist. The computing centroids step (Line 4) in Algorithm 3 is equivalent to computing centroids step (Line 5) in Algorithm 2 using $C_k = [h_{k-1}, h_k]$. In particular, instead of performing the most time consuming classic Clustering assignment step (Line 4) in Algorithm 2, where the distances from every point to every centroid are compared, this step is replaced with step 2 of the Algorithm 3. In this step 2, only the boundaries of the clusters, specifically, h is updated. This is possible because the structure of the thresholding quantizer requires that all values of u's that maps to the same v_i, must be in a same contiguous region. To update h, Eq. (13) is solved for every u_k. We note that x_{u_k} and x_{v_k} are defined in Eqs. (4) and (5), respectively. We also note that the Eq. (13) effectively groups u into the right cluster since the group membership decision occurs when either $D(x_{u_k}||x_{v_{k-1}}) > D(x_{u_k}||x_{v_k})$ or $D(x_{u_k}||x_{v_{k-1}}) < D(x_{u_k}||x_{v_k})$. The root of $D(x_{u_k}||x_{v_{k-1}}) = D(x_{u_k}||x_{v_k})$ acts as the boundaries between clusters, i.e., h. We will show that with a mild assumption on the noise density, $\phi_i(u)$, this Step 2 can be solved effectively with complexity of $O(K^2 \log M)$, where M, similar to Algorithm 2, is the parameter that controls the precision of solution as the continuous noise is discretized. The complexity of Algorithm 3 is therefore much lower than Algorithm 2.

The following Theorem 1 states a mild condition that guarantees Step 2 of Algorithm 3 can be computed in $O(K^2 \log M)$.

Theorem 1. Let

$$F_k(u) = D(x_u || x_{v_{k-1}}) - D(x_u || x_{v_k}) \quad (14)$$

be $K-1$ functions of u, $k = 2, \ldots, K$. If the conditional density $\phi_i(u) = p_{U|X}(u|x_i)$ satisfies:

$$\frac{\phi_i(u)}{\phi_j(u)} \geq \frac{\phi_i(u')}{\phi_j(u')}, \quad (15)$$

for $\forall\ i \leq j$ and $u \leq u'$, then

1) $F_k(u)$ is an increasing function.
2) $F_k(u) = 0$ has a unique root.

Proof. Please see our extension version. □

Corollary 1. If the inequality (15) holds, then the computational complexity of Step 2 in Algorithm 3 is $O(K^2 \log M)$.

Proof. The key to the proof is based on Theorem 1. Since $F_k(u)$ is an increasing function and has a unique root, one can employ a bisection algorithm to find the unique root for $F_k(u)$, which is h_{k-1}. In particular, if for some $u_1 < u_2$ and if $F_k(u_1) < 0$ and $F_k(u_2) > 0$, one can evaluate $F_k(\frac{u_1+u_2}{2})$ to determine whether it is larger or smaller than 0. If it is larger than 0, we repeat the process on the interval $[u_1, \frac{u_1+u_2}{2}]$. Otherwise, we repeat the process on the interval $[\frac{u_1+u_2}{2}, u_2]$.

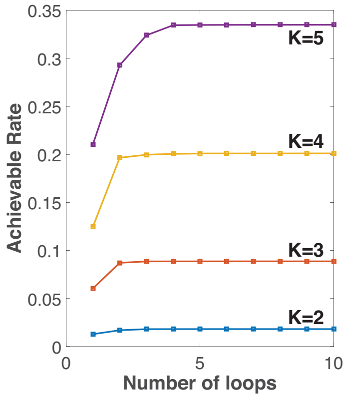

Figure 2: Achievable rates for $K = 2, 3, 4, 5$ as a function of the number of iterations

The process repeats until the solution is found, i.e., within some ϵ away from zero. As a result, there are $O(\log M)$ divisions where $M \sim O(1/\epsilon)$. Since there are $K-1$ roots for $K-1$ functions $F_k(u)$ and the KL-divergence are computed over K-dimensional vector, the running time complexity is therefore $O((K-1)K \log M) \sim O(K^2 \log M)$. We note that using the Newton's method (rather than bisection method) is even more efficient if the derivative of $F_k(u)$ is available. $\square$

In practice, the inequality (15) usually holds. For example, in typical communication scenarios where noise is additive, i.e., $u = x_i + n_i$, then we can show that the inequality (15) hold for a variety of common noise distributions such as normal distribution, exponential distribution, gamma distribution, uniform distribution, and more generally, all log-concave, log-convex distributions (please see our extended version). Therefore, Algorithm 3 is quite useful in real-world scenarios.

As for the computing centroid step in Eq. (12), it can be done in $O(K^2)$. The key is that, often times Eq. (12) can be computed in closed form, and thus can be efficiently computed with the computational complexity of $O(K^2)$. If even the closed form expression for Eq. (12) is not available, one can store the results of evaluating the integral $\int_{h_{k-1}}^{h_k} p_i \phi_i(u) du$ in Eq. (12) for M possible values of h_k ahead of time in a lookup table since these values do not change. Eq. (12) is then performed with appropriate values of h_k found from the updating threshold steps. We note that in the regular k-means algorithm, it is not possible to perform the computing centroids step in $O(K^2)$ since there is no linear structure produced by h_k to allow us exploit the computation of the integral $\int_{h_{k-1}}^{h_k} p_i \phi_i(u) du$ ahead of time for efficient computations.

IV. Simulations and Numerical Results

In this section, we compare the performances in terms of run-time and accuracies of the proposed Algorithm 1 against those of an exhaustive search. Algorithm 1 employs Algorithm 3 in Line 5, while the exhaustive search performs a grid search through all the possible p_X and h. With sufficiently small grid, the exhaustive search returns the correct capacity at the expense of significantly large run-time. Also, we simulate a communication system with additive noise, i.e.,

$$u = x_i + n_i,$$

where n_i are i.i.d $N(0, 1)$. $x_i = \{-1, 1, \ldots, 2(K-1) - 1\}$. As a result, $\phi_i(u) = N(\mu_i, \sigma_i)$ where $\mu_i = -1 + 2(i-1)$ and $\sigma_i = 1$ for $\forall\ i = 1, 2, \ldots, K$.

Fig. 2 illustrates the convergence rates of the proposed Algorithm 1. As seen, the proposed algorithm converges very quickly to the correct values of the channel capacity as indicated in the "actual capacity" column after 10 iterations. We note that the actual capacity is computed using an exhaustive search with the resolution of $\epsilon = 0.1$ for $K = 2, 3$ and $\epsilon = 0.2$ for $K = 4, 5$ while the resolution ϵ of Algorithm 3 is always 0.1. As K increases, the proposed algorithm converges a slightly slower rate, but overall the proposed algorithm is still a very fast and converges to the right values. In fact, the average running time of Algorithm 1 and exhaustive searching using $K = 5$ are 10.22 and 319725.21 seconds, respectively. This is because the computational complexity of Algorithm 3 is $O(TK^2 \log M)$ while the computational complexity of the exhaustive search is $O(M^{K-1})$.

V. Conclusion

In this paper, we proposed a fast algorithm to find the sub-optimal quantizer of a discrete input continuous output channel which maximizes the mutual information between discrete input and quantized output. Both theoretical and numerical results are presented to verify our approach.

References

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

[2] Richard Blahut. Computation of channel capacity and rate-distortion functions. *IEEE transactions on Information Theory*, 18(4):460–473, 1972.

[3] Thuan Nguyen and Thinh Nguyen. On closed form capacities of discrete memoryless channels. In *2018 IEEE 87th Vehicular Technology Conference (VTC Spring)*, pages 1–5. IEEE, 2018.

[4] Joel Max. Quantizing for minimum distortion. *IRE Transactions on Information Theory*, 6(1):7–12, 1960.

[5] Jiuyang Alan Zhang and Brian M Kurkoski. Low-complexity quantization of discrete memoryless channels. In *Information Theory and Its Applications (ISITA), 2016 International Symposium on*, pages 448–452. IEEE, 2016.

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

[7] Francisco Javier Cuadros Romero and Brian M Kurkoski. Decoding ldpc codes with mutual information-maximizing lookup tables. In *Information Theory (ISIT), 2015 IEEE International Symposium on*, pages 426–430. IEEE, 2015.

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

[9] 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.

[10] Arindam Banerjee, Srujana Merugu, Inderjit S Dhillon, and Joydeep Ghosh. Clustering with bregman divergences. *Journal of machine learning research*, 6(Oct):1705–1749, 2005.