Technische Universität München
Lehrstuhl für Nachrichtentechnik
Prof. Dr.sc.tech. Gerhard Kramer

Research Internship

Probabilistic Constellation Shaping with Autoencoders

Vorgelegt von:
David de Andrés Hernández

München, Month Year

Betreut von:
M.Sc. Francesca Diedolo

Research Internship am
Lehrstuhl für Nachrichtentechnik (LNT)
der Technischen Universität München (TUM)
Titel : Probabilistic Constellation Shaping with Autoencoders
Autor : David de Andrés Hernández

David de Andrés Hernández Boschetsriederstr. 55A
81379 München
deandres.hernandez@tum.de

Ich versichere hiermit wahrheitsgemäß, die Arbeit bis auf die dem Aufgabensteller bereits bekannte Hilfe selbständig angefertigt, alle benutzten Hilfsmittel vollständig und genau angegeben und alles kenntlich gemacht zu haben, was aus Arbeiten anderer unverändert oder mit Abänderung entnommen wurde.

München, xx.xx.20xx
. . . .

Ort, Datum (David de Andrés Hernández)

Contents

Abstract

The abstract comes here.

1 Introduction

The constant demand for higher capacity digital links has motivated the development of communication schemes which approach closer and closer the analytical limit of the channel capacity. According to the definition of channel capacity, sending a single bit per time-frequency slot is inefficient. For this reason, higher-order modulations like amplitude shift keying (ASK) or quadrature amplitude modulation (QAM) are used for better efficiency. Under these modulation schemes, the receiver handles more than two signal points per real dimension — the set of spacial signal points is known as constellation. However, these schemes present a constant-width gap to the capacity limit. This is due to the usage of both uniform and discrete probability distributions for the occurrence of the constellation points. While it is not possible to move away from the discrete distributions because of the digital nature of the communication, constellations with specific non-uniform distributions and optimized spatial locations can lead to further capacity improvements.

Constellation shaping thus, is a technique which seeks to optimize the distribution of the transmit symbols. Furthermore, this optimization unfolds into the improvement of the constellation points' location, their occurrence probability or both simultaneously. The first is known as geometric shaping while the latter is known as probabilistic shaping. In both cases, the goal is to maximize the mutual information (MI) of the channel input X and output Y, which we denote with $I(X;Y)$, by optimizing the constellation. This optimization problem arises from the definition of channel capacity C:

$$C = \max_{p(X)} I(X;Y). \tag{1.1}$$

Currently, the optimal $p(x)$ has only be found for specific channels, such as the additive white gaussian noise (AWGN), since knowledge of the channel distribution $p(y|x)$ is required. Still, solving 1.1 can become mathematically intractable despite knowing $p(y|x)$.

Here is where the use of deep-learning is key for finding constellations which maximize $I(X;Y)$, without analytical knowledge of the channel or with very complex $p(y|x)$. As shown in [OH17], the complete communication system can be interpreted as an autoencoder. This approach tackles the physical layer by optimizing the end-to-end performance instead of the performance of the individual components. This is possible thanks to the carefull election of the loss function against which the trainable parameters will be optimized. What is more, geometric shaping under the autoencoder framework has been already performed in [OH17] and [JEY+18]. Still, learning the optimal probability distribution presents an additional challenge — the statistics of the training set are altered during the learning of the probability distribution making numerical approximations unprecise. [SAAH19] and [AC21] address this problem with different proposals. Their results show that the joint application of probabilistic and geometric shaping outperform the PS-QAM scheme from [BSS15] and approach the limit to within 0.1 dB in the AWGN channel. Nonetheless, the existence of multiple and apparently different architectures invites for a comparativfe study. Consequently, in this work we implement and discuss the architectures proposed by [SAAH19] and [AC21].

The rest of this document is organized as follows. Chapter 2, Preliminaries, presents the fundamentals of classical Probabilistic constellation systems as well as of autoencoder systems. Chapter 3 presents a comparative study of two autoencoder architectures which can perform joint porbabilistic and geometric constellation shaping and highlights their limitations. Chapter 4 wraps-up this work.

2 Preliminaries

2.1 Probabilistic Constellation Shaping

In this section our goal is to present the capacity limitations of the commonly used ASK and QAM modulation schemes. These schemes are penalized for two reasons:

1. They use uniform probability densities.
2. The constellation points are equidistant.

In the following we explain the nature of these penalties.

2.1.1 Introduction

We begin with an important result from Information theory. Under a second-moment constraint, also known as power constraint, the probability distribution which maximizes the differential entropy is the Gaussian distribution, denoted with p_G. We thus have

$$h(X) \leq \frac{1}{2} \log \left(2\pi e \sigma^2\right) \tag{2.1}$$

where $\sigma^2 = \mathbb{E}[X^2]$, and with equality if and only if X is Gaussian-distributed. More generally in the multi-dimensional case we have

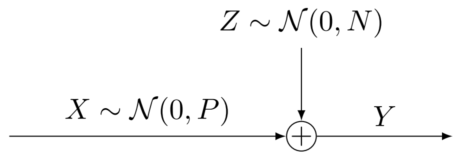
$$h(\underline{X}) \leq \frac{1}{2} \log \left((2\pi e)^n |\mathbf{Q}_{\underline{X}}|\right) \tag{2.2}$$

where we have considered a random column vector $\underline{X}$ of dimension n, mean $\mathbb{E}[\underline{X}] = \underline{m}$ and covariance matrix

$$\mathbf{Q}_{\underline{X}} = \mathbb{E}[(\underline{X} - \underline{m})(\underline{X} - \underline{m})^\mathsf{T}] \tag{2.3}$$

and equality in (2.2) if only if the elements of $\underline{X}$ are jointly Gaussian.

Lets now consider an AWGN channel with Gaussian input X, of zero mean and variance P; Gaussian noise Z, of zero mean and variance N; and output Y; i.e. $Y = X + Z$.



Furthermore, the capacity of the AWGN is

$$\begin{aligned} C(P) &= \max_{P_X : \mathbb{E}[X^2] \leq P} \mathbb{I}(X; Y) && (2.4)\\ &= \max_{P_X : \mathbb{E}[X^2] \leq P} [h(Y) - h(Y|X)] && (2.5)\\ &= \frac{1}{2} \log(2\pi e(P+N)) - \frac{1}{2} \log(2\pi e N) && (2.6)\\ &= \frac{1}{2} \log\left(1 + \frac{P}{N}\right). && (2.7) \end{aligned}$$

We can express the mutual information

$$\mathbb{I}(X; Y) = h(Y) - h(Y|X) \tag{2.8}$$

in two parts, the differential entropy of the output and the conditional differential entropy of the output given the input. We expand the second term as

$$\begin{aligned} h(Y|X) &= h(Y - X|X) && (2.9)\\ &= h(Z|X) && (2.10)\\ &= h(Z) = \frac{1}{2} \log\left(2\pi e \sigma^2\right) && (2.11) \end{aligned}$$

and observe that the term $h(Y|X)$ does not depend on how X is distributed. In contrast, $h(Y)$ does depend on how X is distributed by

$$p_Y(y) = \int_{-\infty}^{\infty} p_X(X) p_Z(y - x)\, dx = (p_X \star p_Z)(y). \tag{2.12}$$

To circumvent the fact that it is difficult to find a closed-form expression of $h(Y)$, we make use of the information divergence as

$$h(Y) \overset{\text{(a)}}{=} h(Y_G) - \mathbb{D}(p_Y \| p_G) \tag{2.13}$$

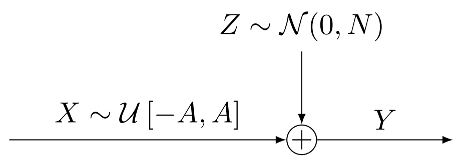
where (a) arises from the fact that $\mathbb{X}(p_X \| p_G) = h(Y_G)$ if and only if p_X has zero mean and variance P as p_G. (2.13) is very useful as it allows us to express the differential entropy of the output in terms of the information divergence between p_G and any other distribution by means of the cross entropy.

Now we can rewrite 2.8 as

$$\mathbb{I}(X;Y) = h(Y) - h(Y|X) \tag{2.14}$$
$$= h(Y) - h(Z) \tag{2.15}$$
$$= h(Y_G) - \mathbb{D}(p_Y \| p_G) - h(Z) \tag{2.16}$$
$$= [h(Y_G) - h(Z)] - \mathbb{D}(p_Y \| p_G) \tag{2.17}$$
$$= C(P/\sigma^2) - \mathbb{D}(p_Y \| p_G). \tag{2.18}$$

This last result indicates that the loss of MI when using a distribution P_X different than P_G is the informational divergence $\mathbb{D}(p_Y \| p_G)$. In other words, if the gaussian distribution is not used, the capacity penalty is characterized by $\mathbb{D}(p_Y \| p_G)$.

2.1.2 Capacity Gap for Uniform Continuous Input



We would like now to understand how far a uniform distribution is from (2.1). To do this, we will follow the approach presented in [Bö18] to lower bound the MI. Start by defining X_u as a uniformly distributed input on the interval $[-A, A]$ where A is carefully chosen so that $\mathbb{E}[{X_u}^2] = P$. The corresponding output is Y_u and we proceed

$$\mathbb{I}(X_u;Y_u) = C(\text{snr}) - \mathbb{D}(p_{Y_u} \| p_{Y_G}) \tag{2.19}$$
$$\geq C(\text{snr}) - \mathbb{D}(p_{X_u} \| p_{X_G}) \tag{2.20}$$
$$= C(\text{snr}) - [h(X_G) - h(X_u)] \tag{2.21}$$
$$= C(\text{snr}) - \frac{1}{2}\log_2\left(\frac{\pi e}{6}\right). \tag{2.22}$$

In Figure 2.1 we display the derived lower bound and observe that the capacity loss, originated from the use of a uniform input density, is never more than $\frac{1}{2}\log_2\frac{\pi e}{6}$ independent of the signal-to-noise ratio (SNR). To show that the shaping gap is tight, it is necessary to proof an upper bound for $\mathbb{I}(X_u;Y_u)$ that approaches 2.22 with increasing SNR. We refer the reader to [Bö18], Section 4.3, for this proof.

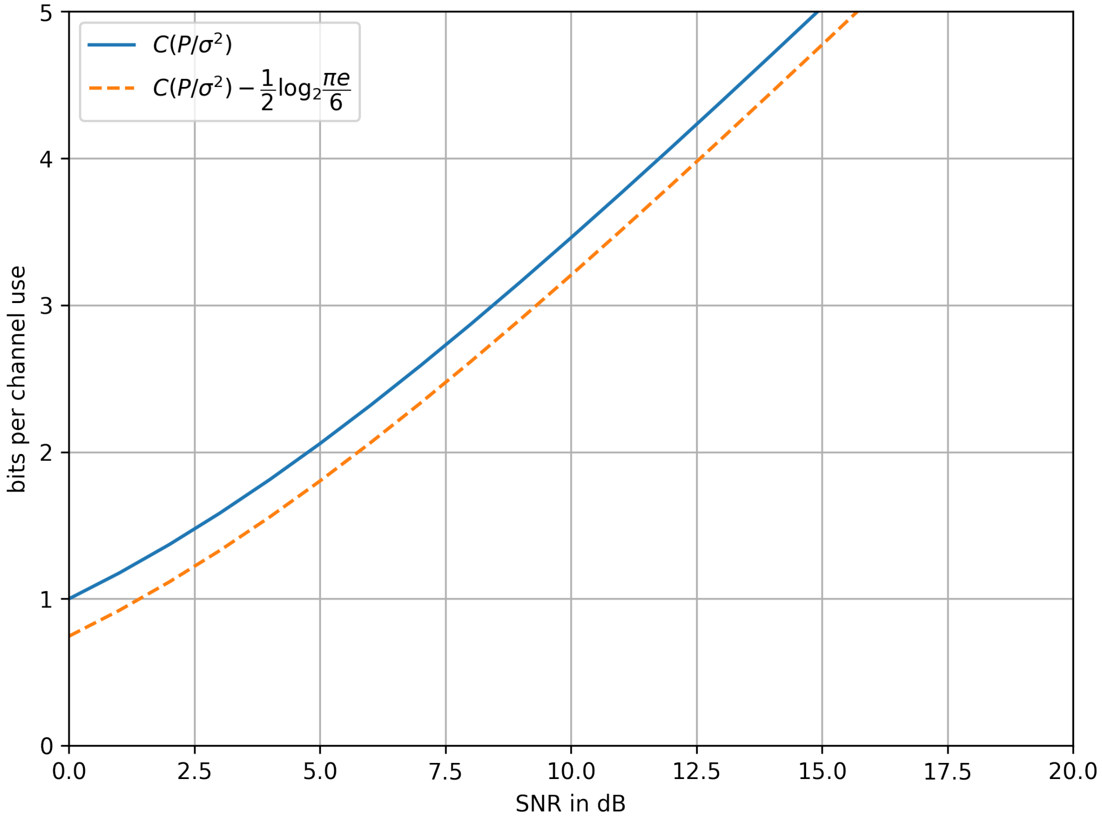


Figure 2.1: AWGN channel capacity gap. The orange line indicates the upper capacity bound for any uniformly distributed constellation.

2.1.3 Uniform Discrete Input Bound

We now show the penalty received for the use of an equidistant M-ASK constellation. We define the constellation points as

$$\mathcal{X} = \{\pm\Delta 1, \pm\Delta 3, \ldots, \Delta(M-1)\}, \tag{2.23}$$

where Δ is the scaling factor of the constellation, so that the channel input is X. If X_M is uniformly distributed, the resulting power is

$$P = \mathbb{E}\left[X_M^2\right] = \Delta^2 \frac{M^2-1}{3}. \tag{2.24}$$

Theorem 1 (Uniform Discrete Input Bound)**.** *The mutual information achieved by X_M is lowered bounded by*

$$\mathbb{I}(X_M; Y_M) \geq \frac{1}{2}\log_2\left(\frac{M^2}{M^2-1}\right) - \frac{1}{2}\log_2\left[2\pi e\left(\frac{P}{M^2-1} + \frac{P}{1+P/\sigma^2}\right)\right] \tag{2.25}$$

$$< C(snr) - \frac{1}{2}\log_2\left(\frac{\pi e}{6}\right) - \frac{1}{2}\log_2\left[1 + \left(\frac{2^{C(snr)}}{M}\right)^2\right] \tag{2.26}$$

$$< C(snr) - Penalty(Uniform\ dist.) - Penalty(Equidistant\ dist.) \tag{2.27}$$

where $snr = P/\sigma^2$.

We refer the reader to [Bö18], section 4.5, for the proof. Theorem 1 shows our goal, namely that both the usage of a uniform distribution and an equidistant constellation penalizes the capacity. We can additionally compute the relation between the constellation size, M, and $C(\mathrm{snr})$ so that the resulting MI is within a constant gap of capacity. To make this result even more attractive, we increase the constraint for the gap to match the order of the distribution loss (0.255 bits). We obtain

$$-\frac{1}{2}\log_2\left[1 + \left(\frac{2^{C(\mathrm{snr})}}{M}\right)^2\right] \leq -\frac{\log_2 e}{2}\left(\frac{2^{C(\mathrm{snr})}}{M}\right)^2 = \frac{1}{4} \tag{2.28}$$

$$\Leftrightarrow M = 2^{C(\mathrm{snr}) + \frac{1}{2} + \frac{1}{2}\log_2\log_2 e} \tag{2.29}$$

by using $\log_e(x) \leq (1-x)$. So if

$$\log_2 M \approx C(\mathrm{snr}) + 0.77, \tag{2.30}$$

then the mutual information is within 0.5 bit of capacity.

2.1.4 Capacity-achieving distributions

We now address the question of finding the discrete probability distribution which maximizes the capacity. Such distribution should be free of the $\frac{1}{2}\log_2\left(\frac{\pi e}{6}\right)$ penalty. We use again an ASK constellation with M signal points (in practice, M is a power of 2) given by

$$\mathcal{X} = \{\pm 1, \pm 3, \ldots, (M-1)\}. \tag{2.31}$$

Let X be a random variable with distribution P_X over $\mathcal{X}$. As before, we scale X by a $\Delta > 0$ and the resulting input/output relation for an AWGN channel becomes

$$Y = \Delta X + Z \tag{2.32}$$

In consequence, the MI of the channel input and output is

$$\begin{aligned} \mathbb{I}(\Delta X; Y) &= \mathbb{I}(\Delta X; \Delta X + Z) && (2.33) \\ &= \mathbb{I}(\Delta X; \Delta X + Z) && (2.34) \end{aligned}$$

where the second equality follows because (ΔX) is a deterministic function of X and vice-versa. Under an input average power constraint P, the scaling Δ and the distribution P_X must be chosen to satisfy

$$\mathbb{E}[(\Delta X)^2] \leq P. \tag{2.35}$$

Formally, our optimization problem is the following

$$C(P/\sigma^2) = \max_{\Delta, P_X : \mathbb{E}[(\Delta X)^2] \leq P} \mathbb{I}(X; \Delta X + Z). \tag{2.36}$$

Maximizing the mutual information $\mathbb{I}(X; \Delta X + Z)$ both over the scaling of the constellation points and the input distribution requires a relatively high amount of power. Instead, as shown in [Bö18], section 5.3, we will use a suboptimal input distribution which follows from maximizing the input entropy.

We expand the mutual information as

$$\mathbb{I}(X, \Delta X + Z) = \mathbb{H}(X) - \mathbb{H}(X | \Delta X + Z) \tag{2.37}$$

and fixing Δ, we select the input distribution P_{X_Δ} that maximizes the input entropy under our power constraint, i.e., we choose

$$P_{X_\Delta} = \underset{P_X:\mathbb{E}[(\Delta X)^2]\leq P}{\arg\max} \ \mathbb{H}(X). \tag{2.38}$$

Without the discrete constraint, the solution would be a Gaussian distribution. For this reason we explore sampled Gaussian distributions, also known as Maxwell-Boltzmann (MB) distributions. For each $\mathcal{X} = \{\pm 1, \pm 3, \ldots, (M-1)\}$, define

$$P_{X_v}(x_i) = A_\nu e^{-\nu {x_i}^2}, \ A_\nu = \frac{1}{\sum\limits_{i=1}^{M} e^{-\nu {x_i}^2}} \tag{2.39}$$

We now show that P_{X_Δ} is given by

$$P_{X_\Delta}(x_i) = P_{X_\nu}(x_i) \text{ with } \nu : \mathbb{E}[(\Delta X_\nu)^2] = P \tag{2.40}$$

Proof. Consider the finite set $\mathcal{X} = x_1, x_2, \ldots, x_n$. Let f be a function that assigns to each $x_i \in \mathcal{X}$ a positive cost $f(x_i) > 0$. Define the MB distribution

$$P_{X_v}(x_i) = A_\nu e^{-\nu f(x_i)}, \ A_\nu = \frac{1}{\sum\limits_{i=1}^{M} e^{-\nu f(x_i)}} \tag{2.41}$$

Let P_X be some distribution on $\mathcal{X}$ with $\mathbb{E}[f(X)] = P$. Choose $\nu : \mathbb{E}[f(X_\nu)] = P$

$$\begin{aligned}
0 &\leq \mathbb{D}(P_X \| P_{X_\nu}) && (2.42)\\
&= \sum_{x\in \text{Support}(P_{X_\nu})} P_X \log\left(\frac{P_X(x)}{P_{X_\nu}(x)}\right) && (2.43)\\
&= -\mathbb{H}(X) - \sum_{x\in \text{Support}(P_{X_\nu})} P_X(x) \log(P_{X_\nu}(x)) && (2.44)\\
&\overset{(*)}{=} -\mathbb{H}(X) - \sum_{x\in \text{Support}(P_{X_\nu})} P_{X_\nu}(x) \log(P_{X_\nu}(x)) && (2.45)\\
&= -\mathbb{H}(X) + \mathbb{H}(X_\nu) && (2.46)\\
\mathbb{H}(X) &\leq \mathbb{H}(X_\nu) && (2.47)
\end{aligned}$$

where the (*) marked step follows since both distributions produce the same moments for $log(P_{X_\nu}(x))$. $\square$

2.2 Autoencoders

2.2.1 Introduction

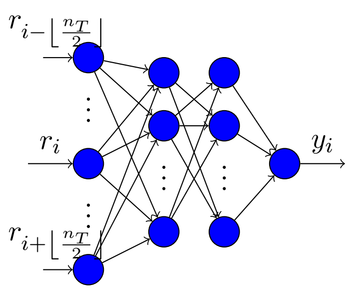
In this section we present the basics of autoencoders in the context of deep learning and communication systems, which was pioneered in [OH17]. The idea behind an autoencoder is to transmit a particular representation of the input data so that at the output, it can be reconstructed with minimal error. This means that the desired representations must be robust with respect to the channel impairments (i.e. noise, fading, distortion, etc.). To find such representations and the corresponding mappings $\mathbf{x}$ to $\mathbf{y}$ we train deep neural networks (NNs). Because NNs are universal function approximators [HSW89], this technique is particularly interesting for training over channels without a mathematically tractable model.

2.2.2 Feed-Forward Neural Networks

Feed-forward neural networkss (FFNNs) are structures capable of transforming the input vector $\mathbf{v_0} = (v_{0,1} \dots v_{0,M})$ into an output vector $\mathbf{v_k} = (v_{K,1} \dots v_{K,n})$, i.e., $\mathbf{v_K} = f_{\mathrm{NN}}(\mathbf{v_0})$. This transformation is accomplished by composing functions, which in turn are computed by layers. Using matrix notation we can express the output of each layer as

$$\mathbf{v_k} = g_{\mathrm{NL},k}(\mathbf{W}_k \mathbf{v}_{k-1} + \mathbf{b}_k), \qquad k = 1, \dots, K, \tag{2.48}$$

where k indicates the layer index, and $g_{\mathrm{NL},k}(\cdot)$ is a nonlinear function applied between layers. Indeed, without the nonlinear function the FFNN lacks the expressive power required to approximate any function [HSW89].



2.2.3 Autoencoders

2.2.4 Stochastic Gradient Descent

To find fitting sets of parameters θ, the most used algorithm is stochastic gradient descent (SGD) which starts with a random set of initial values and then updates θ with each iteration as

$$\theta_{new} = \theta_{old} + \epsilon \frac{\partial}{\partial \theta_{old}} L(\theta_{old}) \tag{2.49}$$

One particular requirement for using SGD to train the autoencoders is that the loss gradient needs to be backpropagated all the way through receiver and channel to the transmitter. Otherwise, the transmitter parameters cannot be updated. This in turn means, that channel and receiver must be available as differentiable functions during training.

We can now set up the problem which we would like to address in this work. Namely, to train a deep NN-based autoencoder system to find a parametric distribution $p_\theta(s)$. By maximizing the MI during training, the output distribution must satisfy 2.18, and thus, approach the channel capacity.

3 Contribution

3.1 Notation

In the following we will use the notation:

$$\mathbb{I}\left(X, Y; D, P_M, C_M\right)$$

which expresses the mutual information between X and Y and D, P_M, C_M, separated by a semi-colon, are the trainable parameters of the system. D stands for the posterior probability distribution learnt by the demapper, P_M stands for the source's probability distribution learnt by the encoder, and C_M stands for the spatial distribution of the constellation points learnt by the mapper. These parameters can be seen as additional input to a function.

3.2 First implementation

In this section we break-down the autoencoder system presented by Stark *et al.* [SAAH19].

3.2.1 Optimization of trainable parameters

As we have seen in Chapter 2, the goal of probabilistic constellation shaping is to maximize the MI. To this end, defining an appropriate loss function is critical. Starting from the demodulator, the categorial cross entropy loss

$$L(D, P_M, C_M) \triangleq \mathbb{X}(P_{X|Y}||Q_{X|Y}; D) = \mathbb{E}\left[-\log_2(Q(X|Y; D))\right] \tag{3.1}$$

is appropriate for training D, but not for P_M and C_M. A modification of this loss function is necessary to ensure that the end-to-end MI is maximized. The following expansions will come handy

$$\mathbb{H}(X) = \mathbb{X}(P_{X|Y}||Q_{X|Y}) - \mathbb{D}(P_{X|Y}||Q_{X|Y}) \tag{3.2}$$

$$\mathbb{H}(X|Y=y) = \mathbb{X}(P_{X|y}||Q_{X|y}|Y=y) - \mathbb{D}(P_{X|y}||Q_{X|y}|Y=y) \tag{3.3}$$

$$\mathbb{H}(X|Y) = \mathbb{E}_y\left[\mathbb{X}(P_{X|y}||Q_{X|y}|Y=y)\right] - \mathbb{E}_y\left[\mathbb{D}(P_{X|y}||Q_{X|y}|Y=y)\right]. \tag{3.4}$$

Using the last expansion we can rewrite the mutual information in terms of the categorical cross entropy

$$\mathbb{I}(X,Y) = \mathbb{H}(X) - \mathbb{X}(P_{X|Y}||Q_{X|Y}) + \mathbb{D}(P_{X|Y}||Q_{X|Y}). \tag{3.5}$$

And the categorical cross entropy loss function becomes

$$L(D, P_M, C_M) \triangleq \mathbb{H}(X) - \mathbb{I}(X,Y) + \mathbb{D}(P_{X|Y}||Q_{X|Y}). \tag{3.6}$$

So, if L is minimized during training, the source entropy is unwantedly minimized. To avoid this effect, Stark *et al.* modify the loss function as

$$\hat{L}(D, P_M, C_M) \triangleq L(D, P_M, C_M) - \mathbb{H}(X). \tag{3.7}$$

With this correction the optimization problem

$$\min_{D,P_M,C_M} \hat{L}(D, P_M, C_M) = \max_{D,P_M,C_M} \{\mathbb{I}(X,Y) - \mathbb{D}(P_{X|Y}||Q_{X|Y})\} \tag{3.8}$$

maximizes the MI.

3.2.2 Autoencoder Architecture

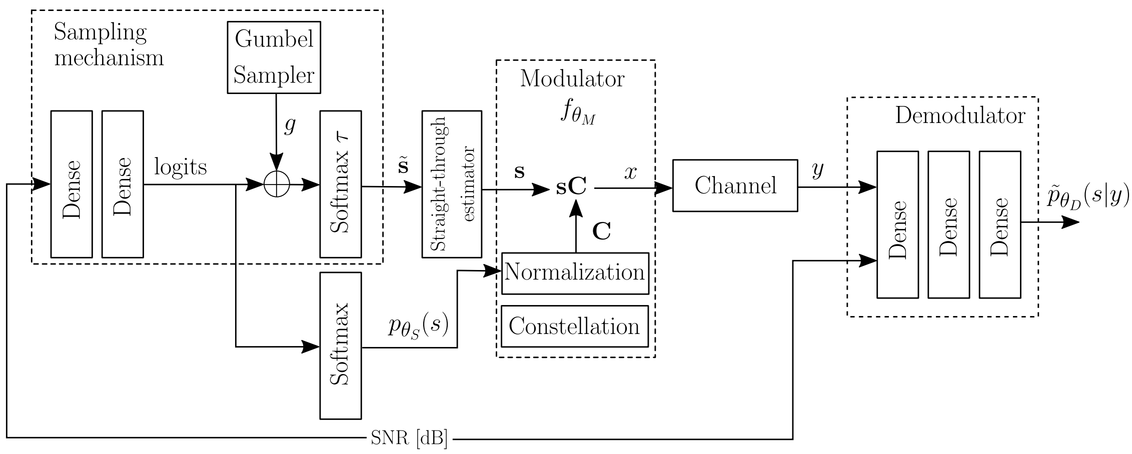


Figure 3.1: Proposed Autoencoder Architecture from [SAAH19]

Stark *et al.*'s autoencoder is made up from three major blocks: sampler, modulator, and demodulator. Fig 3.1 shows the complete architecture of the end-to-end system, where the trainable parameters $p_{\theta_S}(s), f_{\theta_M}$ and $p_{\theta_D}(s|y)$ correspond to our P_M, C_M, and D, respectively. While the modulator and demodulator blocks are similar to the proposal from [OH17], the simultaneous probabilistic shaping is possible thanks to the careful design of the sampler. By ensuring that the sampler mechanism is differentiable, the gradients with respect to each $p_i \in P_M$ are precise when calculated via SGD. In fact, the differentiability is gained by leveraging the so-called *Gumbel-Softmax trick* [JGP16], which circunvents the need for using the arg-max function to sample the discrete distribution P_M.

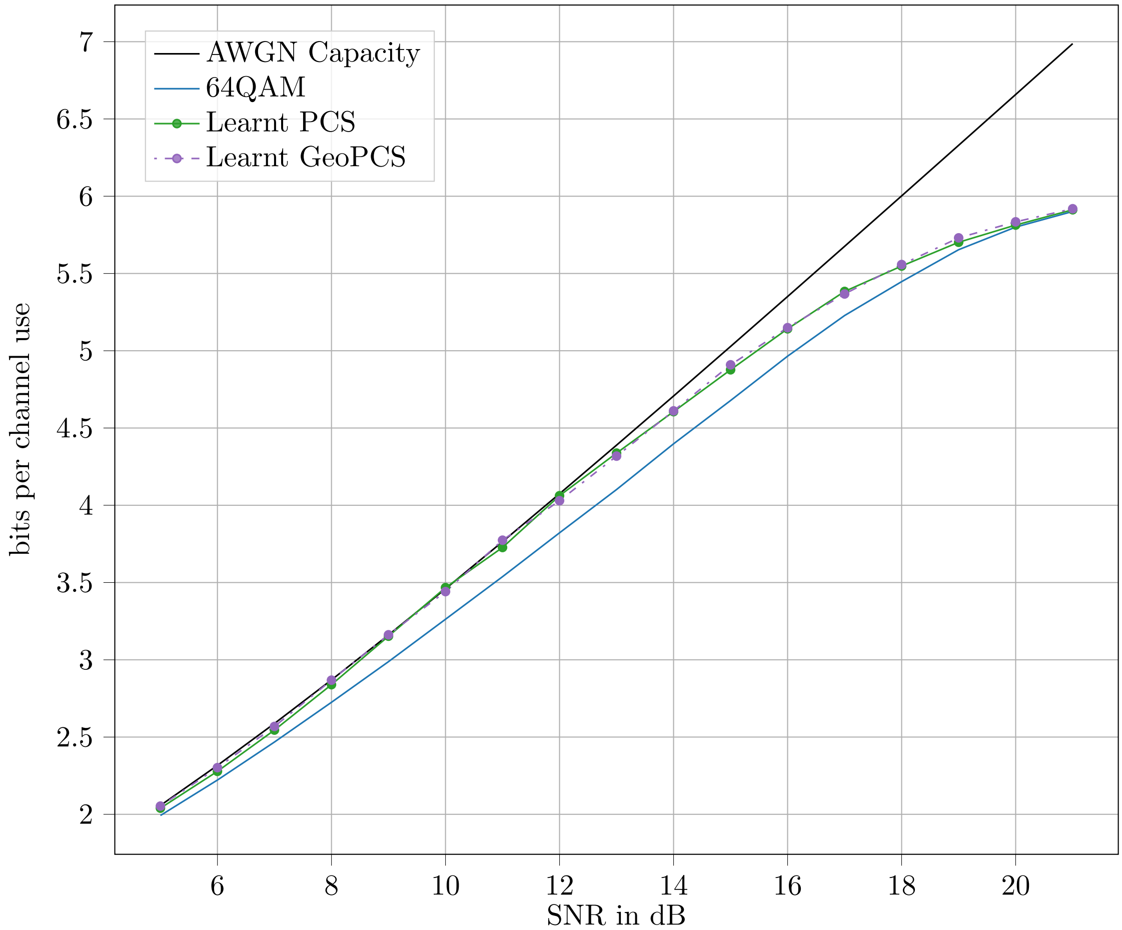
We have implemented the end-to-end system using PyTorch [PGM+19]. The sampler is made out of 2 layers. The first layer is made out of 128 units with ReLU activation, and the second layer of M units with linear activations. In the forward pass, the logit output is then processed through the Gumbel-Softmax trick and then through the straight-through estimator to produce the one-hot-encoded training set. While in the backward pass, the straight-through estimator uses the approximate one-hot vector —the output of the Gumbel-Softmax block. The trainable parameter of the sampler, P_M, is initialized to a uniform distribution. The modulator is made out of a single linear layer of N units followed by a normalization operation to ensure that the energy constraint of the constellation is mantained, i.e.,

$$\sum_{p_i \in P_M} p_i |x_i|^2. \tag{3.9}$$

If only probabilistic shaping is applied, the constellation remains fixed, e.g., M-ASK, is used. On the contrary, when geometric shaping is performed, the parameter C_M corresponds to the unnormalized constellation points. Finally, the demodulator is made out of 3 layers. The first two layers are made out of 128 units with ReLU activations, and the third layer of M units with linear activation. The trainable parameter of the demodulator, D corresponds to the *a posteriori* probabilities $p(x|y)$.

3.2.3 Autoencoder Performance

Training was performed with the Adam optimizer and the hyper-parameters of the training are: learning-rate=0.001, batch-size=10000, and number of epochs=4000. The resulting M-ASK constellations for both only probabilistic and joint probabilistic and geometric shaping are presented in Figure 3.2. Moreover the respective achieved MI for the corresponding M-QAM scheme, i.e., 64-QAM, are available in Figure 3.3 for SNR values ranging from 5dB to 22dB.



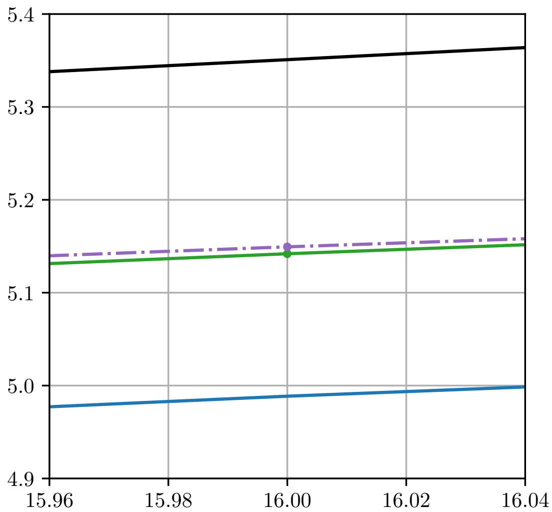


Figure 3.3: (a) Mutual information learned by the PCS and GeoPCS for constellation size M=64 on the AWGN channel

.

3.3 Second implementation

In this section we break-down the autoencoder system presented by Aref and Chagnon [AC21]. This novel approach calls to remove the risk of numerical instabilities present in [SAAH19]. Such instabilities are introduced by the Gumbel-Softmax trick, required to make the sampler differentiable, and the sensitive extra hyper-parameters. The outstanding feature of this autoencoder is the hability to sample from the constellation probabilities without any approximation.

3.3.1 Optimization of trainable parameters

As we have seen in Chapter 2, the goal of probabilistic constellation shaping is to maximize the mutual information

$$\max_{D,P_M,C_M} \mathbb{I}(X,Y;D,P_M,C_M) = \mathbb{H}(X) - \mathbb{X}(P_{X|Y} \| Q_{X|Y}; D, P_M, C_M) \tag{3.10}$$

where the entropy is maximized when the symbols probabilities follow a MB distribution; and the cross-equivocation is minimum when $Q_{X|Y} = P_{X|Y}$.
Typically, the gradient descent (or ascent, as we intent to maximize) allows us to solve the optimization problem by adjusting the trainable parameters as:

$$\theta_{new} = \theta_{old} + \epsilon \frac{\partial}{\partial \theta_{old}} \mathbb{I}(X,Y;\theta_{old}) \tag{3.11}$$

for all trainable parameters $\theta \in P_M, C_M, D$. And the MI can be numerically approximated by

$$\mathbb{I}(X,Y) \approx \mathbb{I}(X,Y)_{\text{num}} = \frac{1}{B}\sum_{i=1}^{B} -\log_2(P(x_i)) + \log_2(Q_{X|Y}(x_i|y_i)) \tag{3.12}$$

$$= \frac{1}{B}\sum_{i=1}^{B} L(x_i,y_i). \tag{3.13}$$

Next, the following approximation usually allows to adjust the trainable parameters:

$$\frac{\partial}{\partial \theta}\mathbb{I}(X,Y;\theta) \approx \frac{\partial}{\partial \theta}\mathbb{I}(X,Y)_{\text{num}} = \frac{1}{B}\sum_{i=1}^{B} L(x_i,y_i). \tag{3.14}$$

However, Aref claims that although this is true for the constellation locations ($\theta \in C_M$) and the demapper parameters ($\theta \in D$), it does not hold for the constellation probabilities

$\{p_1, p_2, \ldots, p_M\} = P_M$

$$\frac{\partial}{\partial p_j}\mathbb{I}\left(X, Y; P_M\right) \not\approx \frac{1}{B}\sum_{i=1}^{B}\frac{\partial}{\partial p_j}L(x_i, y_i) \tag{3.15}$$

as $\{p_1, p_2, \ldots, p_M\}$ changes the statistics of the training set.

For this reason, (3.15) must be computed differently. On the one hand, to compute the derivative of the cross-equivocation, the following expansions are useful

$$\mathbb{X}\left(P_{X|Y}\|Q_{X|Y}|Y=b\right) = \sum_{a\in Supp(P_{X|Y}(\cdot|b))} P_{X|Y}(a|b)\log_2(Q_{X|Y}(a|b)) \tag{3.16}$$

$$\mathbb{X}\left(P_{X|Y}\|Q_{X|Y}\right) = \sum_{b\in Supp(P_Y)} P_Y(b)\mathbb{X}\left(P_{X|Y}\|Q_{X|Y}|Y=b\right) \tag{3.17}$$

as combined together and applying Bayes' theorem they yield

$$\mathbb{X}\left(P_{X|Y}\|Q_{X|Y}\right) = \sum_{(a,b)\in Supp(P_{XY})} P_X(a)P_{Y|X}(b|a)\log_2(Q_{X|Y}(a|b)). \tag{3.18}$$

And so, the derivative results

$$\frac{\partial}{\partial p_j}\mathbb{X}\left(P_{X|Y}\|Q_{X|Y}\right) = \sum_{b \text{ if } x=j} P_{Y|X}(b|j)\log_2 Q_{X|Y}(j|b) \tag{3.19}$$

$$+ \sum_{(a,b)\in Supp(P_{XY})} P_{XY}(a,b)\frac{\partial}{\partial p_j}\log_2 Q_{X|Y}(a|b) \tag{3.20}$$

which can be rewritten using the expectation operator as

$$\frac{\partial}{\partial p_j}\mathbb{X}\left(P_{X|Y}\|Q_{X|Y}\right) = \mathbb{E}_{Y|X}[\log_2 Q_{X|Y}(j|b)|X=j] \tag{3.21}$$

$$+ \mathbb{E}_{XY}[\frac{\partial}{\partial p_j}\log_2 Q_{X|Y}(a|b)]. \tag{3.22}$$

The terms can now be numerically computed as

$$\mathbb{E}_{Y|X}[\log_2 Q_{X|Y}(j|b)|X=j] \approx \frac{1}{Bp_j}\sum_{b \text{ if } x=j}\log_2 Q_{X|Y}(j|b) \tag{3.23}$$

$$\mathbb{E}_{XY}[\frac{\partial}{\partial p_j} \log_2 Q_{X|Y}(a|b)] \approx \frac{1}{B} \sum_{(a,b)\in Supp(P_{XY})} \log_2 Q_{X|Y}(a|b). \tag{3.24}$$

On the other hand, the derivative of the entropy w.r.t. p_j is

$$\frac{\partial}{\partial p_j}\mathbb{H}(X) = \frac{\partial}{\partial p_j} \sum_{i=1}^{B} -p_i \log_2(p_i) = -\log_2(p_j) - log_2(e). \tag{3.25}$$

Now, combining (3.23), (3.24), and (3.25) the derivative of the mutual information w.r.t. p_j, (3.15), can be computed as

$$\frac{\partial}{\partial p_j}\mathbb{I}(X,Y;P_M) \approx -\log_2(p_j) - \log_2(e) + \frac{1}{Bp_j} \sum_{b \text{ if } x=j} \log_2 Q_{X|Y}(j|b) + \frac{1}{B}\sum_{(a,b)} \log_2 Q_{X|Y}(a|b) \tag{3.26}$$

Aref now indicates that the following terms can be computed via backpropagation

$$-\log_2(p_j) + \frac{1}{Bp_j} \sum_{b \text{ if } x=j} \log_2 Q_{X|Y}(j|b) = \frac{1}{B}\sum_{i=1}^{B} \frac{\partial}{\partial p_j} L(x_i, y_i) \tag{3.27}$$

while the remaining ones must be explicitely computed and added to the gradient after backpropagating. We call this step *gradient correction* and it is due to the change of statistics in the sampled batch.

3.3.2 Autoencoder Architecture

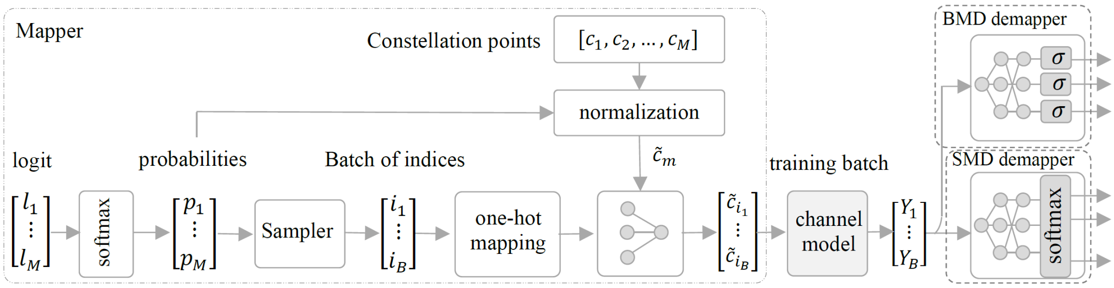
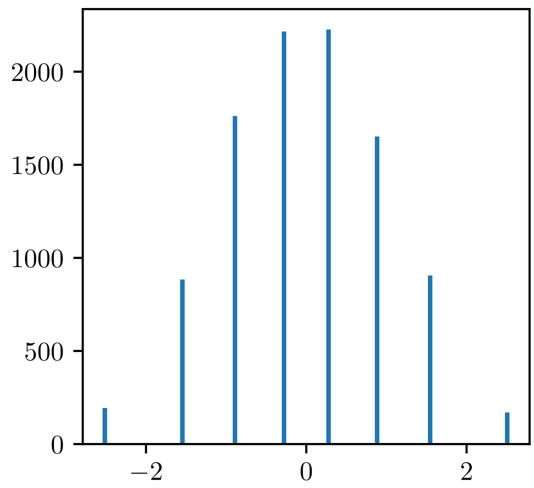


Figure 3.4: Proposed Autoencoder Architecture from [AC21]

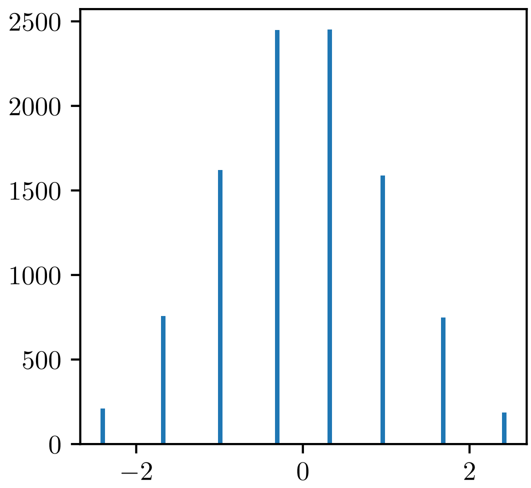
Aref and Chagnon's autoencoder is made up from two major blocks: mapper and demapper. Fig 3.4 shows the complete architecture of the end-to-end system, where the trainable parameters are P_M, C_M, and D. Furthermore, the mapper breaks down into the sampling and the modulation mechanism. In order to sample, a single linear layer followed by a softmax layer trains the $p_i \in P_M$. Next, to produce a training symbols batch of size B, each index i is drawn about Bp_i times, for $i = 1, \dots, M$. After, the indeces are randomly permuted. Note that this sampling mechanism is not differentiable and consequently, the derivatives of the loss function w.r.t to the p_i will not be accurate. However, using the *gradient correction* factor described in Section 3.3.1, the gradient is adjusted during the backpropagation step. Similar to the previously presented architecture, the modulation mechanism is made of a single linear layer of M units and trainable parameters $c_i \in C_M$. It also includes a normalization layer to ensure that the power contstraints are met. Finally, the Demapper is also made of a single linear layer followed by a softmax layer. The demapper's trainable parameter, D correspond to the *a posteriori* probability distribution $p(x|y)$ depending on the channel model.

3.3.3 Autoencoder Performance

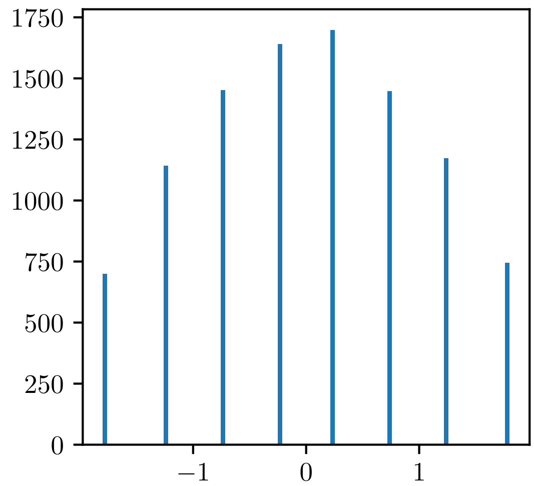
Training was performed with the Adam optimizer and the hyper-parameters of the training are: learning-rate=0.1, batch-size=10000, and number of epochs=4000. The resulting M-ASK constellations for both only probabilistic and joint probabilistic and geometric shaping are presented in Figure **??**. Moreover the respective achieved MI for the corresponding M-QAM scheme, i.e., 64-QAM, are available in Figure 3.5 for SNR values ranging from 5dB to 22dB.



(a) SNR = 5dB



(b) SNR = 12dB



(c) SNR = 18dB

Figure 3.2: (a-c) Learnt ASK constellation for M=8.

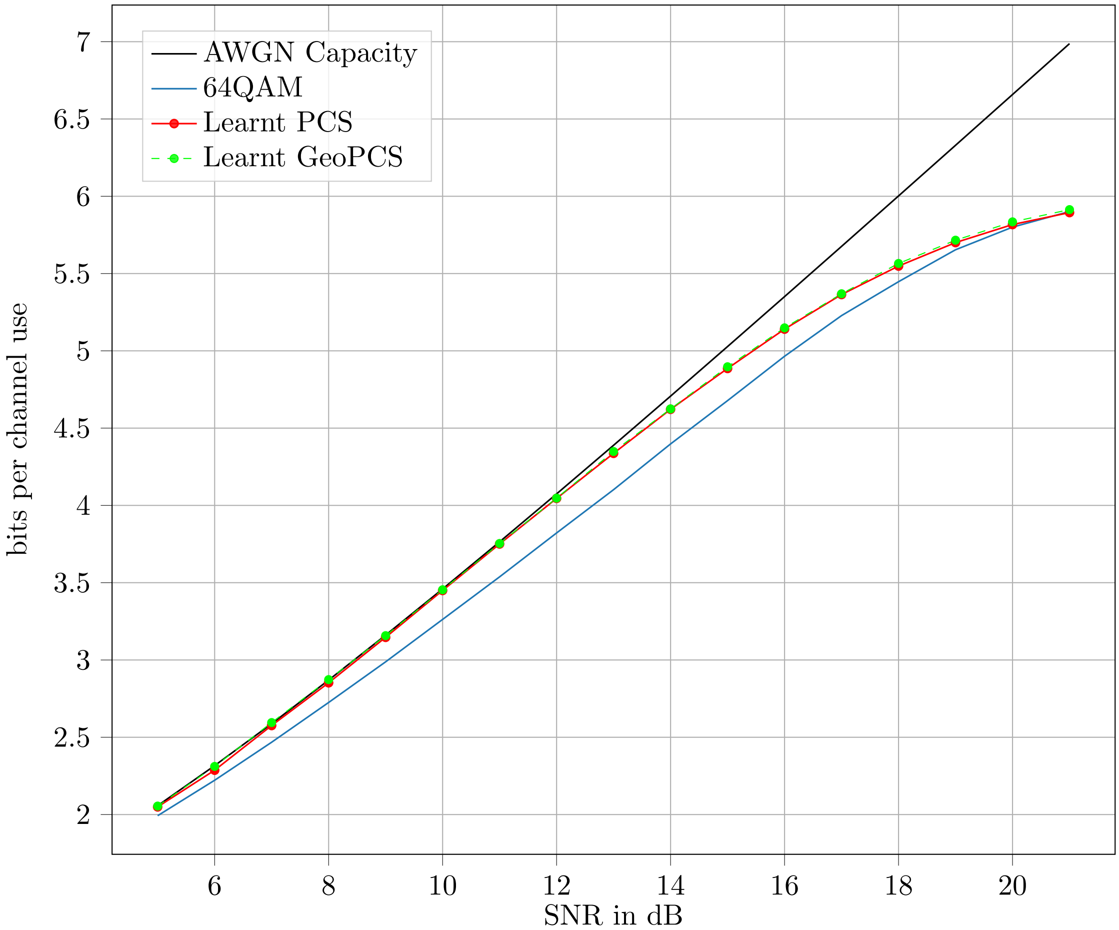


Figure 3.5: Mutual information learned by the probabilistic constellation shpaing on the AWGN channel.

4 Conclusions

[WIP] Include the access to the prob. distribution through the entropy branch of the DCG of the Entropy loss factor. Include the drawback of the Aref? method which minimizes the possible effects that the channel can have on the probability distribution, e.g. the entropy path is principal. Include other possible channels different than AWGN. Implementing complex-based numbers approaches require delicate conversion to real an imaginary parts.

Appendix A

First appendix goes here.

Appendix B

Second appendix goes here.

List of Abbreviations

ASK	amplitude shift keying.
AWGN	additive white gaussian noise.
MB	Maxwell-Boltzmann.
MI	mutual information.
NN	neural network.
QAM	quadrature amplitude modulation.
SGD	stochastic gradient descent.
SNR	signal-to-noise ratio.

Bibliography

[AC21] V. Aref and M. Chagnon, *End-to-end learning of joint geometric and probabilistic constellation shaping*, 2021. DOI: 10.48550/ARXIV.2112.05050. [Online]. Available: https://arxiv.org/abs/2112.05050.

[BSS15] G. Böcherer, F. Steiner, and P. Schulte, "Bandwidth efficient and rate-matched low-density parity-check coded modulation," *IEEE Transactions on Communications*, vol. 63, no. 12, pp. 4651–4665, 2015. DOI: 10.1109/TCOMM.2015.2494016.

[Bö18] G. Böcherer, *Principles of coded modulation. [online] available*, 2018. [Online]. Available: http://www.georg-boecherer.de/bocherer2018principles.pdf.

[HSW89] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, no. 5, pp. 359–366, 1989, ISSN: 0893-6080. DOI: https://doi.org/10.1016/0893-6080(89)90020-8. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0893608089900208.

[JEY+18] R. T. Jones *et al.*, *Geometric constellation shaping for fiber optic communication systems via end-to-end learning*, 2018. DOI: 10.48550/ARXIV.1810.00774. [Online]. Available: https://arxiv.org/abs/1810.00774.

[JGP16] E. Jang, S. Gu, and B. Poole, *Categorical reparameterization with gumbel-softmax*, 2016. DOI: 10.48550/ARXIV.1611.01144. [Online]. Available: https://arxiv.org/abs/1611.01144.

[OH17] T. O'Shea and J. Hoydis, "An introduction to deep learning for the physical layer," *IEEE Transactions on Cognitive Communications and Networking*, vol. 3, no. 4, pp. 563–575, 2017. DOI: 10.1109/TCCN.2017.2758370.

[PGM+19] A. Paszke *et al.*, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds., Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[SAAH19] M. Stark, F. Ait Aoudia, and J. Hoydis, "Joint learning of geometric and probabilistic constellation shaping," in *2019 IEEE Globecom Workshops (GC Wkshps)*, 2019, pp. 1–6. DOI: 10.1109/GCWkshps45667.2019.9024567.