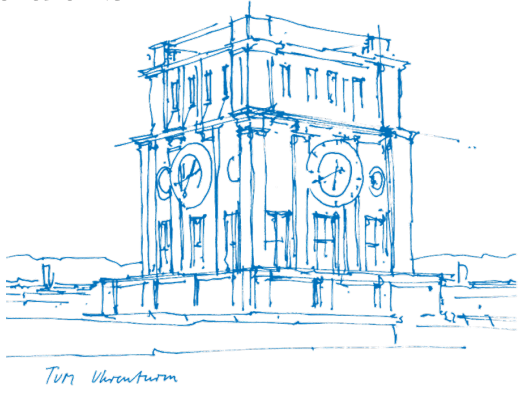
Geometric and Probabilistic Constellation Shaping with Autoencoders

Research Internship



David de Andrés Hernández
Technical University of Munich
Institute for Communications Engineering

October 24, 2022

Agenda

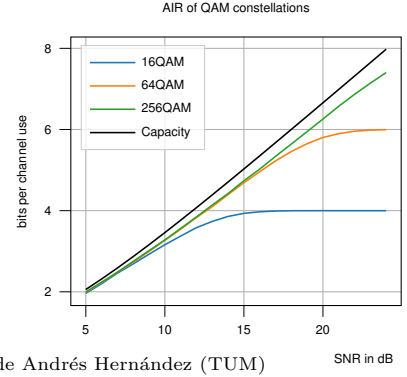
1. Introduction
   - SNR Gap
   - Probabilistic Constellation Shaping
   - Geometric Shaping
2. Autoencoders
   - Challenges
3. Contribution
   - First Implementation
   - Second Implementation
4. Conclusions

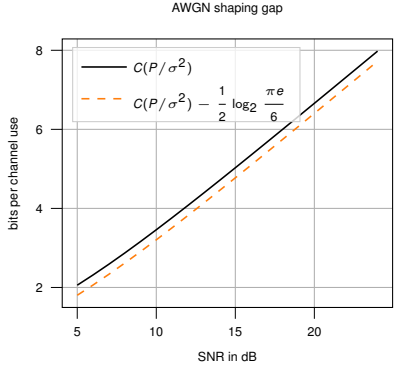
Introduction

- We want to make use of each channel's capacity as efficiently as possible

$$C = \max_{p(X):\mathbb{E}[X^2]\leq P} \mathbb{I}(X;Y)$$

- The optimal $p(x)$ has only been found for specific channels, such as the AWGN, since knowledge of the channel distribution $p(y|x)$ is required.





- Simple problem definition: maximize the MI under a power constraint
- Sending a single bit is inefficient. For this reason, QAM and ASK schemes are used. So depending on the SNR you are targetting you might choose the cardinality of your scheme.
- However, if we are to improve the AIR of these schemes we will soon find out that there are some barriers.

Closing the SNR Gap

- ASK and QAM modulation schemes are penalized for two reasons:
  1. They use uniform probability densities
  2. The constellation points are equidistant
- Solution 1: Shape the probability of occurrence of the constellation points — Probabilistic Constellation Shaping
- Solution 2: Shape the spatial location of the constellation points — Geometric Constellation Shaping

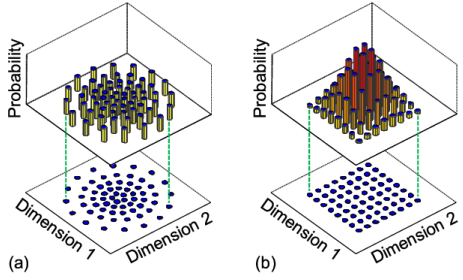


Figure: (a) Probabilistic Constellation Shaping, (b) Geometric Constellation Shaping; [CW19]

Autoencoders

- O'Shea and Hoydis [OH17] pioneered the idea of interpreting the complete communication system as an autoencoder
- Idea: transmit a particular representation of the input data so that at the output, it can be reconstructed with minimal error
- This representations must be robust with respect to the channel impairments (i.e. noise, fading, distortion, etc.) — bottleneck in the autoencoder jargon

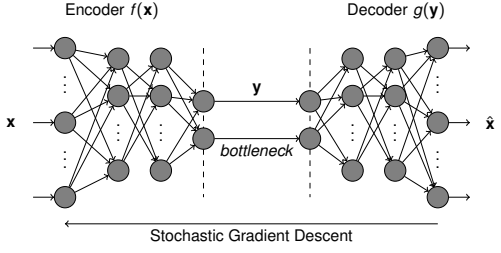
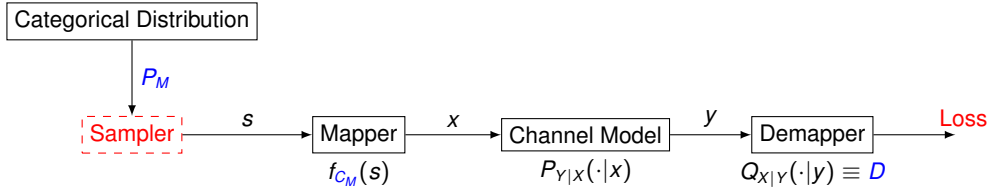


Figure: Representation of the NN of an autoencoder.

Why does it makes sense to use ML for this problem?

- Finding the constellation parameters when $p(y|x)$ is very complex or unknown can be mathematically untractable
- NN have the property of being universal function approximators [HSW89]
- The autoencoder is implemented using Feed-Forward Neural Networks (FFNN), and the parameters are learned using Stochastic Gradient Descent (SGD)
- — —
- An autoencoder is a type of neural network used to learn efficient codings of unlabeled data (unsupervised learning).
- The autoencoder learns a representation (encoding) for a set of data, typically for dimensionality reduction, by training the network to ignore insignificant data ("noise").
- The chosen encoding is validated and refined by attempting to regenerate the input from the encoding.

Challenges



- To find fitting sets of parameters $\{P_M, C_M, D\}$ we define a loss function, $L(P_M, C_M, D)$, and update the parameters using SGD.
- Geometric shaping using autoencoders is a solved problem [OH17], [JEY+18].
- Probabilistic shaping is different because of the added stochastic node (sampler).

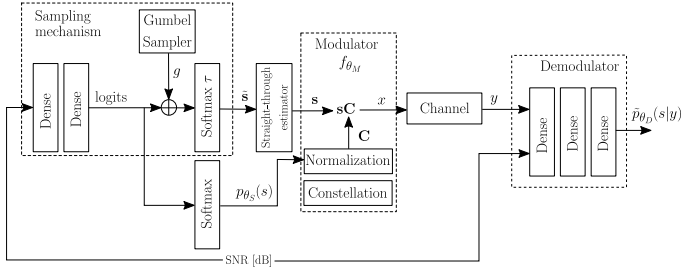
- To find fitting sets of parameters $\{P_M, C_M, D\}$ we define a loss function, $L(P_M, C_M, D)$, that compares the current output of the autoencoder with the desired output from the training set.
- To compute the gradient efficiently a computational graph stores the transformations to the factors which influenced the loss function
- This requires **every block** to be **differentiable**
- Automatic differentiation (pytorch cannot handle complex numbers)

First implementation [SAAH19]

Trainable parameters:

- P_M, source's probability distribution learnt by the encoder.
- C_M, spatial distribution of the constellation points learnt by the mapper.
- D, posterior probability distribution learnt by the demapper.

Autoencoder architecture:



Gumbel-Softmax trick [JGP16]

- Solves the problem of backpropagating through stochastic nodes by reparametrizing the samples to avoid breaking the dependency between the samples and the trainable parameters.
- Relaxes the argmax function using a softmax instead, which is smooth in $\tau > 0$. The parameter τ, controls the degree of approximation to the expected value of the categorical distribution.

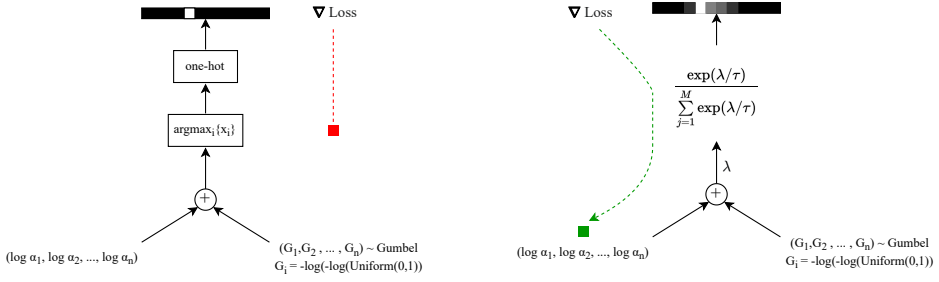


Figure: Non-differentiable path vs. differentiable path using the Gumbel-Softmax trick.

Loss Function

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 categorical cross entropy loss

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

is appropriate for training D and C_M, but not P_M.

Loss Function

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 categorical cross entropy loss

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

is appropriate for training D and C_M, but not P_M.To see why, we rewrite the MI as

$$\mathbb{I}(X, Y) = \mathbb{H}(X) - \mathbb{H}(X|Y)$$

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

Loss Function

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 categorical cross entropy loss

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

is appropriate for training D and C_M, but not P_M.To see why, we rewrite the MI as

$$\mathbb{I}(X, Y) = \mathbb{H}(X) - \mathbb{H}(X|Y)$$

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

And the Loss becomes

$$L(D, P_M, C_M) = \underbrace{\mathbb{H}(X)}_{!} - \mathbb{I}(X, Y) + \mathbb{D}(P_{X|Y}||Q_{X|Y}).$$

So, if L is minimized during training, the source entropy is unwantedly minimized.

Loss Function (cont'd)

To avoid this effect, Stark *et al.* modify the loss function as

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

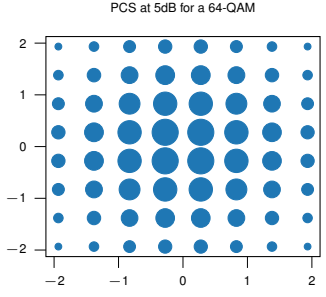
With this correction the optimization problem

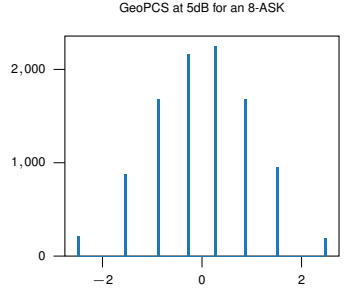
$$\min_{D,P_M,C_M} \hat{L}(D, P_M, C_M) = \max_{D,P_M,C_M} \{\mathbb{I}(X, Y) - \mathbb{D}(P_{X|Y}||Q_{X|Y})\}$$

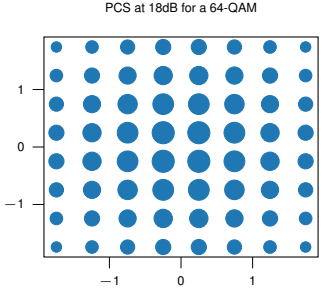
maximizes the MI.

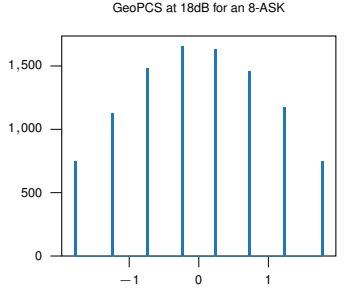
- Training the E2E system by minimizing L corresponds to miximizing the MI, while minimizing the ID between the true posterior distribution and the one learned by the receiver.

Learnt Constellations

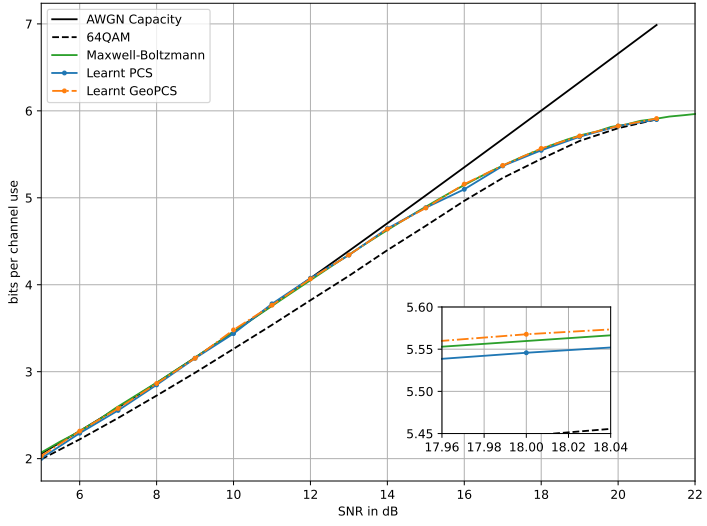








Overall Performance



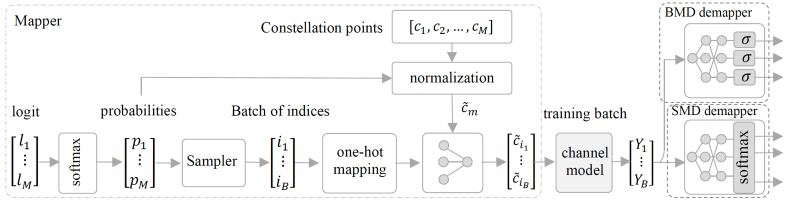
Second implementation [AC21]

Motivation: *The Gumbel-Softmax trick is complex and numerically unstable.*

Claim: The sampler does not need to be differentiable as long as we have other means of computing the gradient w.r.t P_M

Trainable parameters:

- P_M, source's probability distribution learnt by the sampler.
- C_M, spatial distribution of the constellation points learnt by the mapper.
- D, posterior probability distribution learnt by the demapper.



Loss Function

The goal is to to maximize the achievable transmission rate

$$\max_{D,P_M,C_M} \mathbb{R}_{PS}(X,Y;D,P_M,C_M) = \max_{D,P_M,C_M} \left\{ \mathbb{H}(X) - \mathbb{X}(P_{X|Y} \| Q_{X|Y}; D, P_M, C_M) \right\}. \tag{1}$$

And the rate can be numerically approximated by

$$\mathbb{R}_{PS}(X,Y) \approx \mathbb{R}_{PS}(X,Y)_{\text{num}} = \frac{1}{B} \sum_{i=1}^{B} \underbrace{-\log_2(P(x_i)) + \log_2(Q_{X|Y}(x_i|y_i))}_{L(x_i,y_i)} \tag{2}$$

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

$$\frac{\partial}{\partial\theta}\mathbb{R}_{PS}(X,Y;\theta) \approx \frac{\partial}{\partial\theta}\mathbb{R}_{PS}(X,Y)_{\text{num}} = \frac{1}{B}\sum_{i=1}^{B}\frac{\partial}{\partial\theta}L(x_i,y_i). \tag{3}$$

Loss Function (Cont'd)

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

$$\frac{\partial}{\partial p_j} \mathbb{R}_{PS}(X, Y; P_M) \approx \frac{1}{B} \sum_{i=1}^{B} \frac{\partial}{\partial p_j} L(x_i, y_i) \tag{4}$$

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

Loss Function (Cont'd)

- The derivative of the rate w.r.t. p_j, (4), results to be

$$\frac{\partial}{\partial p_j}\mathbb{R}_{PS}\left(X, Y; P_M\right) \approx \underbrace{-\log_2(p_j)}_{\text{backpropagation}} - \log_2(e) + \underbrace{\frac{1}{Bp_j}\sum_{b \text{ if } x=j} \log_2 Q_{X|Y}(j|b)}_{\text{backpropagation}} + \frac{1}{B}\sum_{(a,b)} \log_2 Q_{X|Y}(a|b) \tag{5}$$

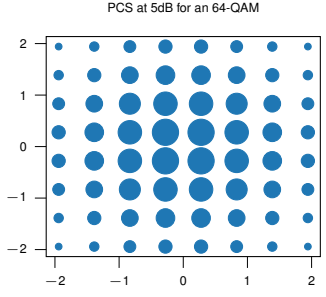
- The following terms can be computed via backpropagation

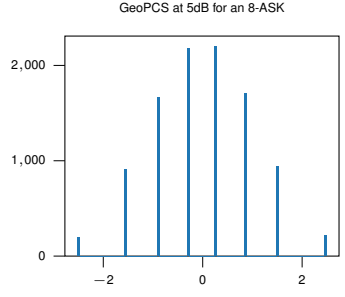
$$-\log_2(p_j) + \frac{1}{Bp_j}\sum_{b \text{ if } x=j} \log_2 Q_{X|Y}(j|b) = \frac{1}{B}\sum_{i=1}^{B} \frac{\partial}{\partial p_j} L(x_i, y_i) \tag{6}$$

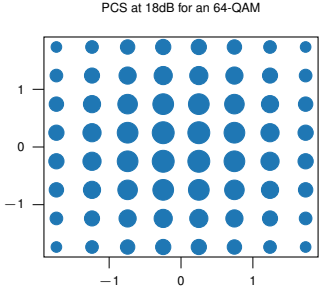
  while the remaining ones must be explicitly computed and added to the gradient after backpropagating.

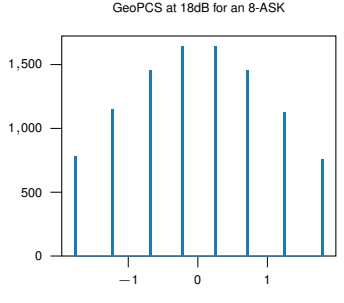
- We call this step *gradient correction* and it is due to the change of statistics in the sampled batch.

Learnt Constellations

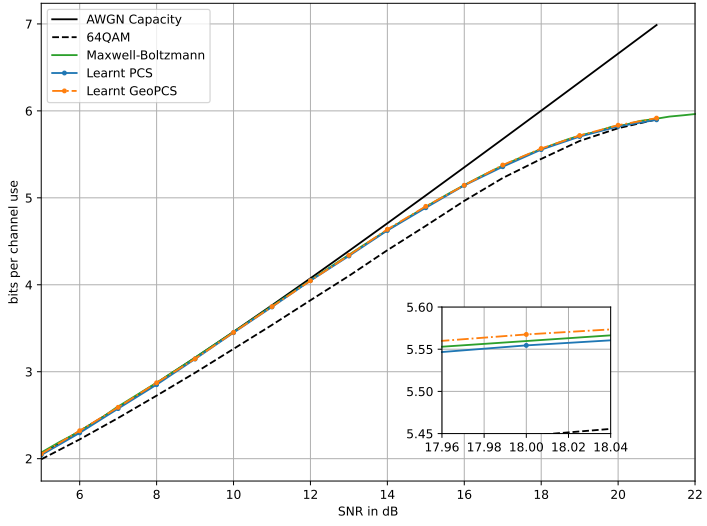








Overall Performance



Conclusions

- Both autoencoder proposals show close-to-optimal performance over the AWGN channel.

Over other channels we expect that they will exhibit different performance.

Conclusions

- Both autoencoder proposals show close-to-optimal performance over the AWGN channel.
- Common keys for success:
  1. the choice of the loss function
  2. and correct computation of the gradient w.r.t P_M

Over other channels we expect that they will exhibit different performance.

Conclusions

- Both autoencoder proposals show close-to-optimal performance over the AWGN channel.
- Common keys for success:
  1. the choice of the loss function
  2. and correct computation of the gradient w.r.t P_M
- Main differences:
  1. Stark *et al.* use the Gumbel-Softmax Trick to allow backpropagation through the sampler
  2. Aref and Chagnon use a non-differentiable sampling mechanism but compute the gradient manually

Over other channels we expect that they will exhibit different performance.

Conclusions

- Both autoencoder proposals show close-to-optimal performance over the AWGN channel.
- Common keys for success:
  1. the choice of the loss function
  2. and correct computation of the gradient w.r.t P_M
- Main differences:
  1. Stark *et al.* use the Gumbel-Softmax Trick to allow backpropagation through the sampler
  2. Aref and Chagnon use a non-differentiable sampling mechanism but compute the gradient manually
- The potential of the autoencoder approach is for training over complex channels such as optical fiber.

Over other channels we expect that they will exhibit different performance.

Conclusions

- Both autoencoder proposals show close-to-optimal performance over the AWGN channel.
- Common keys for success:
  1. the choice of the loss function
  2. and correct computation of the gradient w.r.t P_M
- Main differences:
  1. Stark *et al.* use the Gumbel-Softmax Trick to allow backpropagation through the sampler
  2. Aref and Chagnon use a non-differentiable sampling mechanism but compute the gradient manually
- The potential of the autoencoder approach is for training over complex channels such as optical fiber.
- Both implementations introduce $\mathbb{H}(X)$ into the loss function. For the 2nd implementation this has the effect of adding a complementary path to the computational graph for altering the tensor's gradient w.r.t P_M without backpropagating through the channel model.

Over other channels we expect that they will exhibit different performance.

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.

[CW19] J. Cho and P. J. Winzer, "Probabilistic constellation shaping for optical fiber communications," *Journal of Lightwave Technology*, vol. 37, pp. 1590–1607, 2019.

[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.

[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.

Back-up Slides

Why does it makes sense to use ML for this problem?

- Finding the constellation parameters when $p(y|x)$ is very complex or unknown can be mathematically untractable
- NN have the property of being universal function approximators [HSW89]
- O'Shea and Hoydis [OH17] pioneered the idea of interpreting the complete communication system as an autoencoder

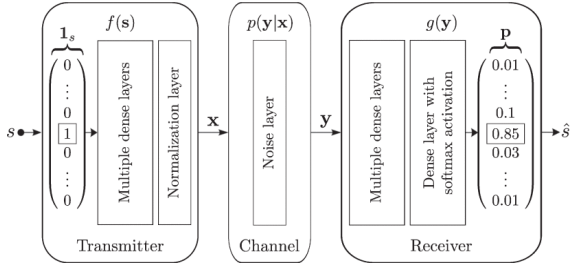


Figure: Autoencoder architecture proposed by [OH17].

- An autoencoder is a type of neural network used to learn efficient codings of unlabeled data (unsupervised learning).
- The autoencoder learns a representation (encoding) for a set of data, typically for dimensionality reduction, by training the network to ignore insignificant data ("noise").
- The chosen encoding is validated and refined by attempting to regenerate the input from the encoding.

Autoencoders

- Idea: transmit a particular representation of the input data so that at the output, it can be reconstructed with minimal error
- This representations must be robust with respect to the channel impairments (i.e. noise, fading, distortion, etc.) — bottleneck in the autoencoder jargon
- The autoencoder is implemented using Feed-Forward Neural Networks (FFNN), and the parameters are learned using Stochastic Gradient Descent (SGD)

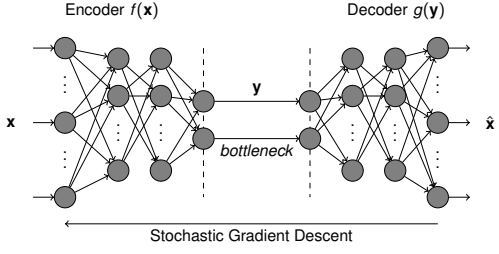
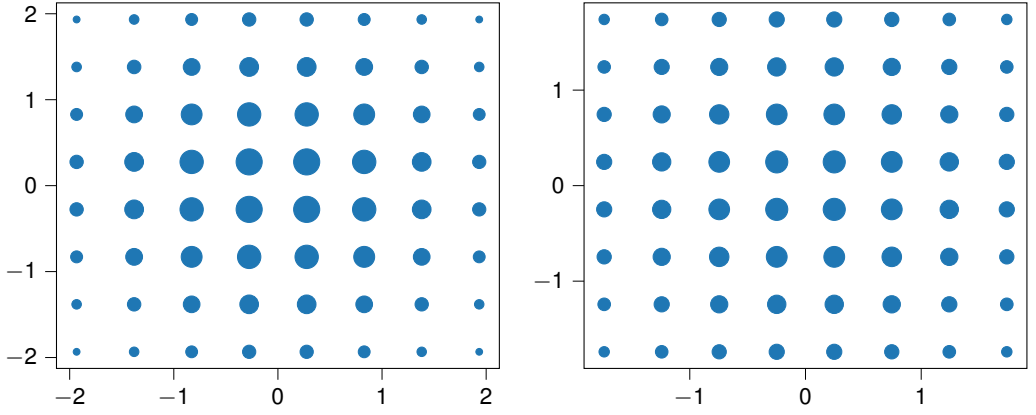


Figure: Representation of the NN of an autoencoder.

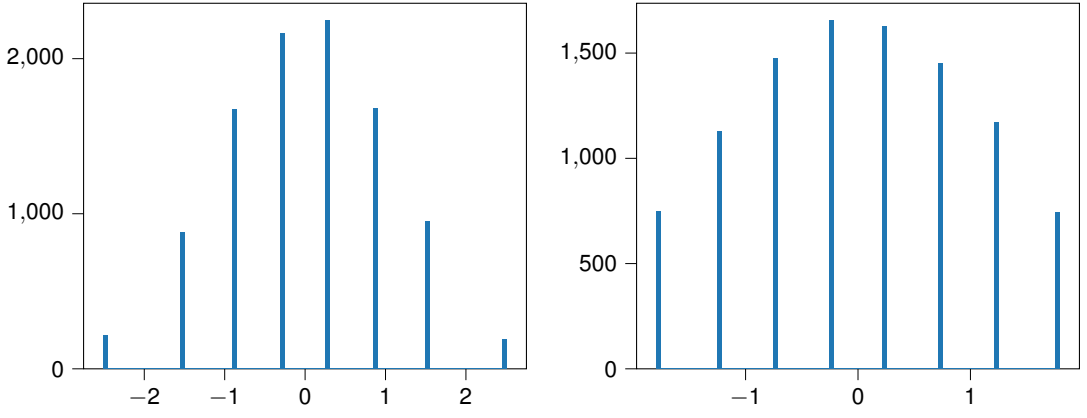
Probabilistic Constellation Shaping (1st Imp)



(a) 5dB (b) 18db

Figure: Learnt probabilistic constellation shaping for M = 64. The size of the markers is proportional to the transmission probability of the symbol. When trained under 5dB, the probabilistic shaping approaches a Gaussian. While under 18dB it approaches a uniform distribution.

Joint Probabilistic and Geometric Shaping (1st Imp)

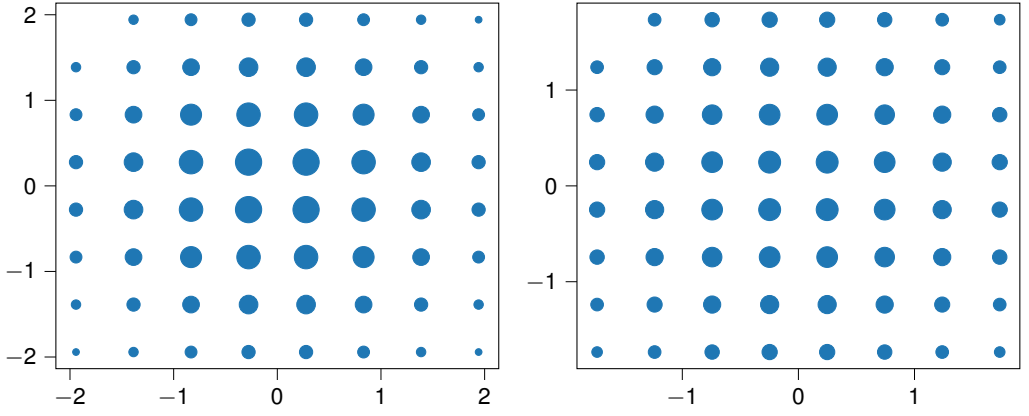


(a) SNR = 5dB

(b) SNR = 18dB

Figure: Learnt joint geometric and probabilistic ASK constellations for M=8.

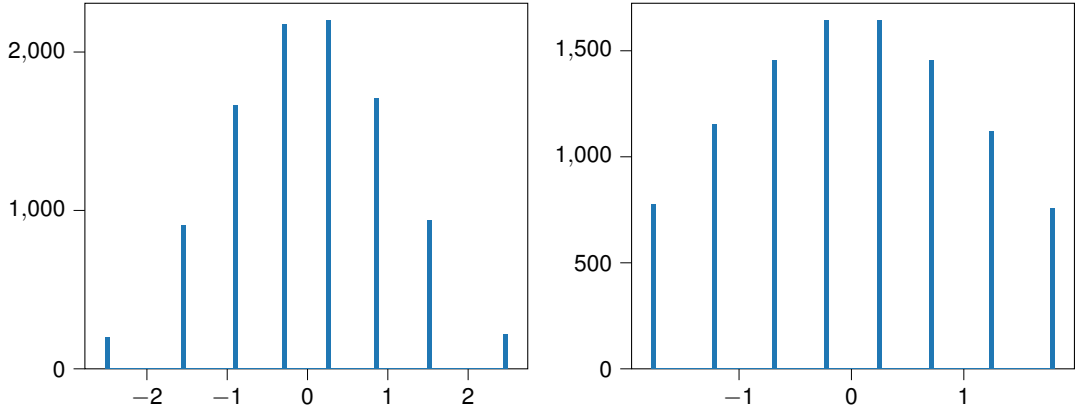
Probabilistic Shaping (2nd Imp)



(a) SNR = 5dB (b) SNR = 18dB

Figure: Learnt probabilistic constellation shaping for M = 64. The size of the markers is proportional to the transmission probability of the symbol. When trained under 5dB, the proabilistic shaping approaches a gaussian. While under 18dB it approaches a uniform distribution.

Joint Geometric and Probabilistic Shaping (2nd Imp)



(a) SNR = 5dB

(b) SNR = 18dB

Figure: Learnt joint geometric and probabilistic ASK constellations for M=8.