Machine Learning for Massive MIMO Communications

Wei Yu

joint work with
Foad Sohrabi, Kareem M. Attiah, and Hei Victor Cheng

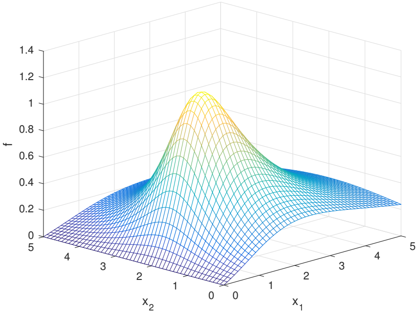
University of Toronto

July 2020

Why is Machine Learning so Powerful?

- Universal functional mapping – either by supervised or reinforcement learning
- Incorporating vast amount of data over poorly defined problems
- Highly parallel implementation architecture

- Mathematical optimization requires highly structured models over well defined problems.
- Finding solution efficiently relies on specific and often convex optimization landscape.



- Traditional approach for communication engineering is to model-then-optimize.
- Machine learning approach allows us to be data driven thereby skipping models altogether!

Role of Machine Learning for Communication System Design

- Traditionally, communication engineers have invested heavily on channel models.
  - However, models are inherently only an approximation of the reality;
  - Moreover, model parameters need to be estimated – with inherent estimation error.
- Machine learning approach allows us to skip channel modeling altogether!
  - End-to-end communication system design
  - Implicitly accounting for channel uncertainty
- This talk will provide two examples in massive MIMO design for mmWave communications
  - Multiuser channel estimation and feedback for FDD massive MIMO
  - Constellation design for symbol-level precoding in TDD massive MIMO

Massive MIMO for mmWave Communication

- **Motivation:** mmWave massive MIMO for enhanced mobile broadband in the downlink.
- **Key problem:** How to obtain channel state information (CSI)?
- **Time-Division Duplex (TDD) Massive MIMO:**
  - Channel reciprocity can be assumed.
  - Uplink pilot transmission followed by CSI estimation at BS and downlink transmission.
- **Frequency-Division Duplex (FDD) Massive MIMO:**
  - Channel reciprocity does not necessarily hold in different frequencies
  - Downlink pilot transmission followed by CSI estimation and feedback at the users.

Part I

Channel Estimation and Feedback for FDD Massive MIMO

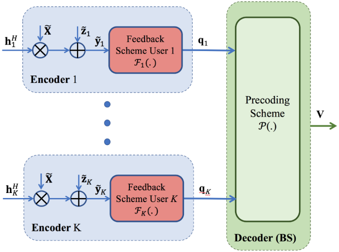
Channel Estimation, Feedback and Precoding in FDD Massive MIMO

Conventional downlink FDD wireless system design involves:

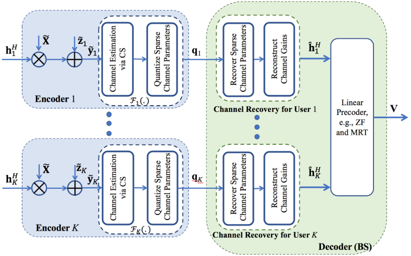
- Independent channel estimation at each UE based on downlink pilot.
- Independent quantization and feedback of each user's channel to the BS.
- Multiuser precoding at the BS based on channel feedback from ALL the users.

Channel Estimation and Feedback

Key Observation: Single-user channel feedback for multiuser precoding is NOT optimal.



FDD downlink precoding as a DSC problem.

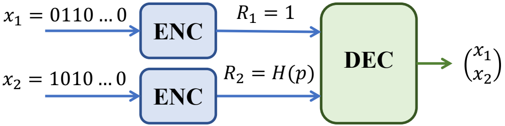


The conventional scheme amounts to a separate source coding strategy.

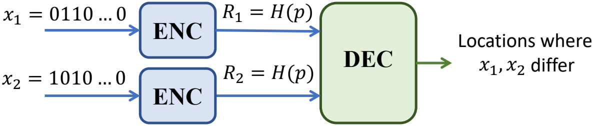
- The FDD feedback/precoding problem is a distributed source coding (DSC) problem.
- Much more efficient distributed feedback scheme can be designed.

Distributed Source Coding

- The information theoretic study of distributed source coding originated in the 1970's.
- Recovering correlated sources with separate encoders and joint decoder:
  - [Slepian and Wolf, 1973] shows that optimal lossless DSC of correlated sources can be much more efficient than independent encoding/decoding.
    Example: $x_1, x_2 \in Ber(0.5)$ but differing with probability p in each position.



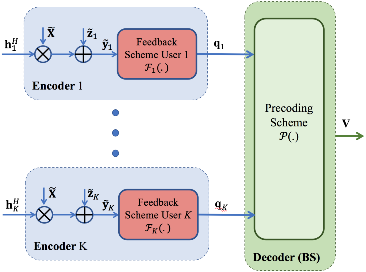
  - [Wyner and Ziv, 1976] extends the results to lossy compression.
- Computing a function of multiple sources:
  - [Korner and Marton, 1979] shows how to compute mod-2 sum of two correlated sequences.



  - [Nazer and Gastpar, 2007] shows DSC has benefit even when the sources are independent.

Channel Estimation and Feedback as Distributed Source Coding

- We recognize that the end-to-end design of a downlink FDD precoding system can be regarded as a DSC problem of computing a function (the downlink precoding matrix) of independent sources (channels) under finite feedback rate constraints.



- The design of the optimal DSC strategy is, however, a difficult problem in general.
  - Statistics of the source needs to be known.
  - Optimal distributed source coding method needs to be designed.
- **This motivates us to propose a deep-learning methodology to jointly design:** (i) the pilot; (ii) a deep neural network (DNN) at each UE for channel feedback, and (iii) a DNN at the BS for precoding to achieve much better performance without explicitly channel estimation.

Deep Learning Approach to Distributed Source Coding

Why is deep learning well suited to tackle the DSC design problem?

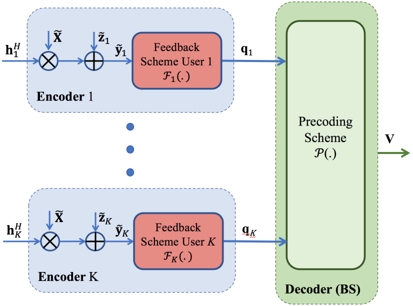
- Different from the convectional design methodology, deep learning can jointly design all the components for end-to-end performance optimization.
- Deep learning implicitly learns the channel distributions in a data-driven fashion without requiring tractable mathematical channel models.
- Computation using trained DNN can be highly parallelized, so that the computational burden of DNN is manageable.

Some recent work on the use of DNNs for FDD system design:

- Single-user scenario with no interference:
  - [Wen, Shih, and Jin, 2018] and [Jang, Lee, Hwang, Ren, and Lee, 2020].
- Channel reconstruction at the BS under perfect CSI assumption:
  - [Lu, Xu, Shen, Zhu, and Wang, 2019] and [Guo, Yang, Wen, Jin, and Li, 2020].

This work:

- Considers the multiuser case and take the channel estimation process into account.
- Provides end-to-end training, including pilot design, channel estimation process and precoder design, to directly maximize the system throughput.



- K-user FDD downlink precoding system involves two phases:
  1. Downlink training and Uplink feedback phase:

$$\begin{aligned} \tilde{\mathbf{y}}_k &= \mathbf{h}_k^H \tilde{\mathbf{X}} + \tilde{\mathbf{z}}_k, \\ \mathbf{q}_k &= \mathcal{F}_k(\tilde{\mathbf{y}}_k), \end{aligned}$$

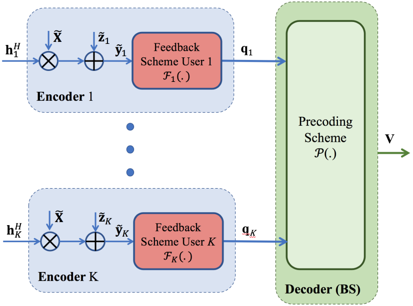
► BS broadcasts L downlink pilots.
► Each user feedbacks B bits.

  2. Downlink precoding for data transmission:

$$\mathbf{V} = \mathcal{P}(\mathbf{q}_1, \ldots, \mathbf{q}_K),$$

► BS maps KB bits to precoder on M antennas.

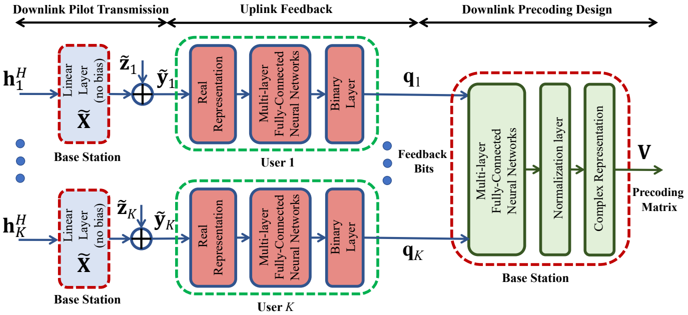
- **Goal:** Designing training pilots, feedback scheme at the users, and precoding scheme at the BS to maximize throughput.



- **Problem of Interest:** Sum rate maximization problem under power constraint P:

$$\begin{array}{ll} \underset{\widetilde{\mathrm{X}},\ \{\mathcal{F}_k(\cdot)\}_{k=1}^K,\ \mathcal{P}(\cdot)}{\text{maximize}} & \displaystyle\sum_{k=1}^{K} \log_2\left(1+\frac{|\mathrm{h}_k^H \mathrm{v}_k|^2}{\sum_{j\neq k}|\mathrm{h}_k^H \mathrm{v}_j|^2+\sigma^2}\right) \\ \text{subject to} & \mathrm{V}=\mathcal{P}\left(\mathcal{F}_1(\mathrm{h}_1^H\widetilde{\mathrm{X}}+\widetilde{\mathrm{z}}_1),\ldots,\mathcal{F}_K(\mathrm{h}_K^H\widetilde{\mathrm{X}}+\widetilde{\mathrm{z}}_K)\right), \\ & \mathrm{Tr}(\mathrm{V}\mathrm{V}^H)\leq P, \\ & \|\widetilde{\mathrm{x}}_\ell\|^2\leq P, \end{array}$$

Proposed DNN Architecture



- **Downlink Pilot Transmission:** Modelled by a linear neural layer followed by additive noise.
- **Uplink Feedback:** Modelled by an R-layer DNN with B binary activation neurons at the last layer: $\mathrm{q}_k = \mathrm{sgn}\left(\mathrm{W}_R^{(k)}\sigma_{R-1}\left(\cdots\sigma_1\left(\mathrm{W}_1^{(k)}\bar{\mathrm{y}}_k + \mathrm{b}_1^{(k)}\right)\cdots\right) + \mathrm{b}_R^{(k)}\right)$.
- **Downlink Precoding Design:** Modelled by a T-layer DNN with normalization activation function at the last layer: $\mathrm{v} = \widetilde{\sigma}_T\left(\widetilde{\mathrm{W}}_T\widetilde{\sigma}_{T-1}\left(\cdots\widetilde{\sigma}_1\left(\widetilde{\mathrm{W}}_1\mathrm{q} + \widetilde{\mathrm{b}}_1\right) + \cdots\right) + \widetilde{\mathrm{b}}_T\right)$.
- Sum rate maximization can be cast as the following learning problem:

$$\max_{\widetilde{\mathrm{X}}, \left\{\Theta_{\mathrm{R}}^{(k)}\right\}, \Theta_{\mathrm{T}}} \mathbb{E}_{\mathrm{H},\widetilde{\mathrm{z}}}\left[\sum_k \log_2\left(1 + \frac{|\mathrm{h}_k^H \mathrm{v}_k|^2}{\sum_{j\neq k}|\mathrm{h}_k^H \mathrm{v}_j|^2 + \sigma^2}\right)\right], \tag{2}$$

- DNN training is performed using stochastic gradient descent (SGD) via back-propagation.
- **Challenge:** The gradient of the binary hidden layer is always zeros.
- **Solution:** Approximate sgn(u) in back-propagation phase with a differentiable function, $f(u)$.
- Straight-through (ST) [Hinton's Lectures]:

$$f(u) = u.$$

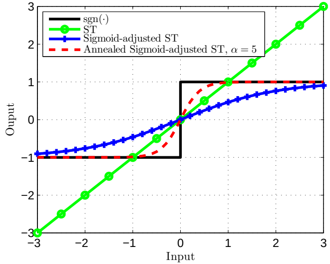
- Sigmoid-adjusted ST [Bengio, Léonard, and Courville, 2013]:

$$f(u) = 2\,\mathrm{sigm}(u) - 1.$$

- Annealed Sigmoid-adjusted ST [Chung, Ahn, and Bengio, 2016]:

$$f(u) = 2\,\mathrm{sigm}(\alpha^{(i)}u) - 1, \text{ where } \alpha^{(i)} \geq \alpha^{(i-1)}.$$

- In this work, we adopt sigmoid-adjusted ST with the annealing trick.



Generalizability

Robustness:

- The DNNs are trained under varying different channel models to ensure robustness.

Enhancing generalizability for arbitrary K:

- All different users adopt a common set of DNN parameters.
- The DNN parameters and the pilot sequences are designed by end-to-end training of a single-user system.
- The BS-side DNN are obtained by training a K-user system with the user-side DNNs fixed.

Enhancing generalizability for arbitrary B:

- **Goal:** Design a common user-side DNN to operate over a wide range of feedback rates.
- Modify user-side DNN to output soft information (which can be quantized later at different values of B) by using a `tanh()` function at the output layer.
- Train the modified user-side DNN to obtain its parameter and the pilot sequences.
- Apply different quantization resolutions to the user-side DNN, then conduct another round of training to design the BS-side DNN

Simulation Settings: Channel Model and DNN Parameters

Channel Model:

- We consider a limited-scattering propagating environment, e.g., mmWave channels:

$$\mathrm{h}_k = \frac{1}{\sqrt{L_p}} \sum_{\ell=1}^{L_p} \alpha_{\ell,k} \mathrm{a}_t\left(\theta_{\ell,k}\right),$$

- L_p is the number of propagation paths,
- $\alpha_{\ell,k} \sim \mathcal{CN}(0,1)$ is the complex gain of the ℓ^{th} path,
- $\theta_{\ell,k} \sim \mathcal{U}(-30^\circ, +30^\circ)$ is the AoD of the ℓ^{th} path,
- $\mathrm{a}_t(\cdot)$ is the array response vector, e.g., $\mathrm{a}_t(\theta) = \left[1, e^{j\pi \sin(\theta)}, \ldots, e^{j\pi(M-1)\sin(\theta)}\right]$.

DNN Implementation:

- **Implementation platform:** TensorFlow and Keras.
- **Optimization method:** Adam optimizer with an adaptive learning rate initialized to 0.001.
- **# hidden layers:** $T = 4$ and $R = 4$.
- **# hidden neurons/layer:** $[1024, 512, 256, B]$ for the user-side DNNs,
  [1024,512,512,2KM] for the BS-side DNN.
- **Activation function of the hidden layers:** Rectified linear units (ReLUs).

Numerical Results: Performance Comparison

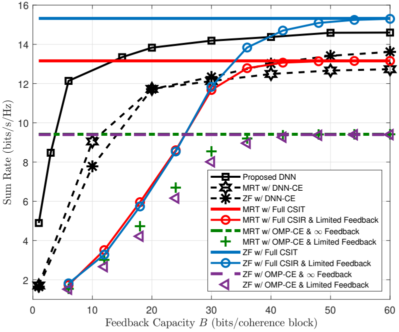


Figure: Sum rate achieved by different methods in a 2-user FDD system with $M = 64$, $L = 8$, $L_p = 2$, and $\text{SNR} \triangleq 10\log_{10}(\frac{P}{\sigma^2}) = 10\text{dB}$.

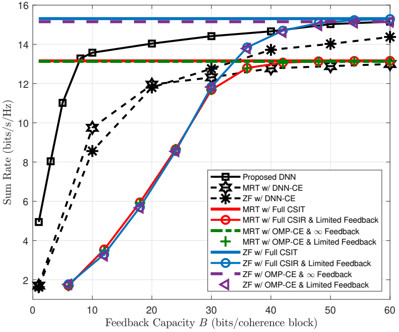


Figure: Sum rate achieved by different methods in a 2-user FDD system with $M = 64$, $L = 64$, $L_p = 2$, and $\text{SNR} \triangleq 10\log_{10}(\frac{P}{\sigma^2}) = 10\text{dB}$.

Numerical Results: Generalizability in L_p

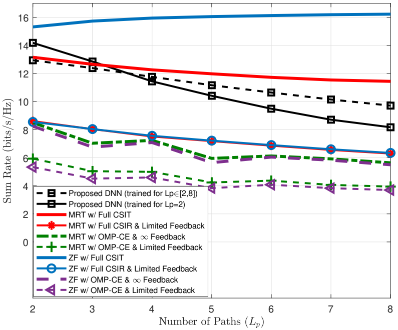


Figure: Sum rate achieved by different methods in a 2-user FDD system with $M = 64$, $L = 8$, $B = 30$, and SNR $= 10$dB.

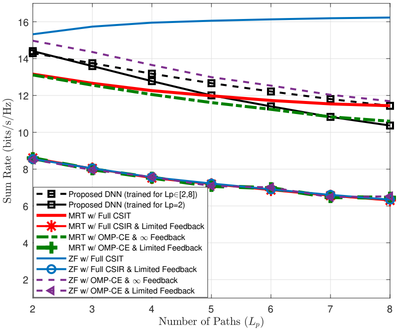


Figure: Sum rate achieved by different methods in a 2-user FDD system with $M = 64$, $L = 64$, $B = 30$, and SNR $= 10$dB.

Numerical Results: Generalizability in B

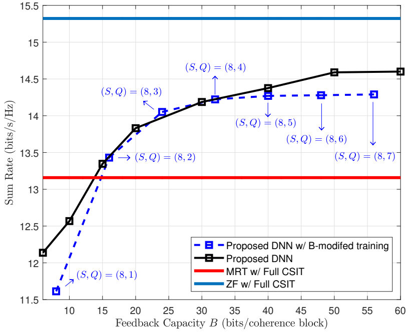


Figure: Sum rate achieved by different methods in a 2-user FDD system with $M = 64$, $L = 8$, $L_p = 2$, and SNR $= 10$dB.

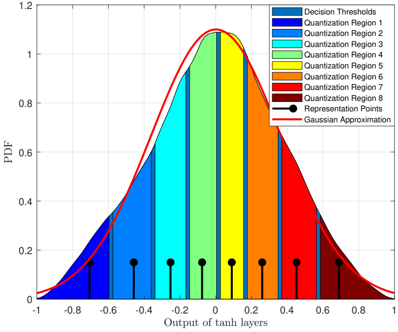


Figure: The empirical PDF of the soft output layer in the modified user-side DNN, trained for $M = 64$, $K = 2$, and $L = 8$. This figure also indicates the quantization regions and the corresponding representation points for the optimal 3-bit quantizer.

Numerical Results: Generalizability in K

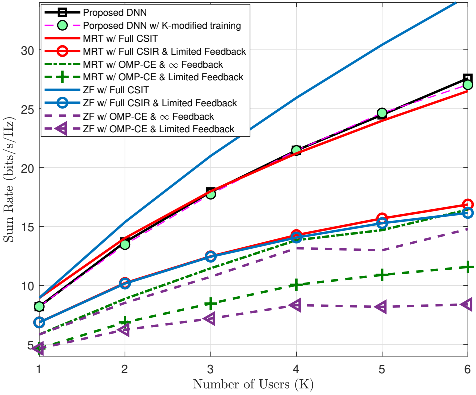


Figure: Sum rate achieved by different methods in a K-user FDD system with $M = 64$, $L = 8$, $B = 30$, $L_p = 2$, and SNR = 10dB.

- As the input dimension of the decoding DNN is KB, for larger values of K we need to increase the capacity of the BS's DNN in order to fully process the input signals.
- In this simulations, we employ a 4-layer DNN at the BS with $[2048, 1024, 512, 2MK]$ number of neurons per layer.

Summary of Part I

- This work shows that the design of a downlink FDD massive MIMO system with limited feedback can be formulated as a DSC problem.
- To solve such a challenging DSC problem, we propose a novel deep learning framework.
- In particular, we represent an end-to-end FDD downlink precoding system, including the downlink training phase, the uplink feedback phase, and the downlink precoding phase, using a user-side DNN and a BS-side DNN.
- We propose a machine learning framework to jointly design:
  - The pilots in the downlink training phase,
  - The channel estimation and feedback strategy adopted at the users,
  - The precoding scheme at the BS.
- We also investigate how to make the proposed DNN architecture more generalizable to different system parameters.
- Numerical results show that the proposed DSC strategy for FDD precoding, which bypasses explicit channel estimation, can achieve an outstanding performance.

Part II

Symbol-Level Precoding for TDD Massive MIMO

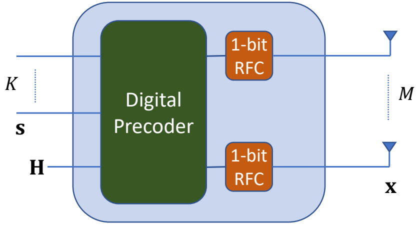
Motivation

In TDD systems, CSI can be estimated in the uplink for downlink beamforming due to reciprocity.

- **Fully Digital Beamforming**
  - Requires one high-resolution RF chain per antenna element.
  - Has high power consumption and hardware complexity.
- **Lower-Complexity Architectures:**
  - Analog Beamforming
  - Antenna Switching
  - Hybrid Beamforming
  - One-Bit Precoding ✓

Beamforming design is a challenging problem. Further, how to take CSI uncertainty into account?

One-Bit Precoding Architecture



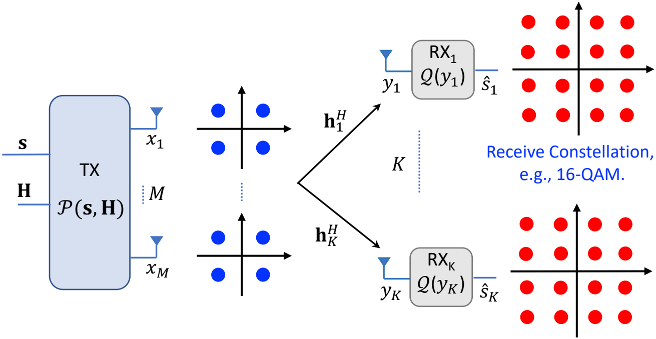
Precoder $\mathcal{P}(\mathbf{s}, \mathbf{H})$

- One RF chain is dedicated to each antenna but with only 1-bit resolution per dimension.
- The transmitted signal of each antenna is chosen from: $\mathcal{X} = \left\{ \frac{1}{\sqrt{2}} (\pm 1 \pm \imath) \right\}$.
- Power saving due to low-resolution digital-to-analog converter.

How to Perform One-Bit Precoding?

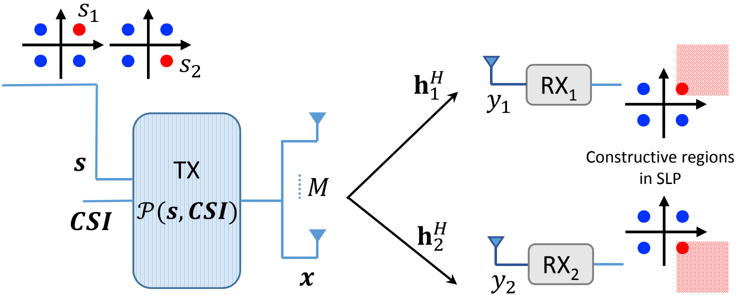
- Quantized-ZF one-bit precoding: [Saxena, Fijalkow, and Swindlehurst, 2016].
  - Performance at moderate-to-high SNRs is limited by quantization noise.
- One-bit beamforming at both transmitter and receivers: [Usman, Jedda, Mezghani, and Nossek, 2016].
  - Restricted to the QPSK constellation.
- One-bit precoding for higher order modulations:
  - Examples: POKEMON [Castañeda, Goldstein, and Studer, 2017], SQUID [Jacobsson, Durisi, Coldrey, Goldstein, and Studer, 2016], and Greedy-exhaustive one-bit precoding [Sohrabi, Liu, and Yu, 2018].
  - Restricted to the conventional QAM and PSK constellations.

- We can actually *jointly* design the receive constellation and one-bit precoder.
  - Machine learning, specifically the concept of autoencoder, allows us to do this efficiently.

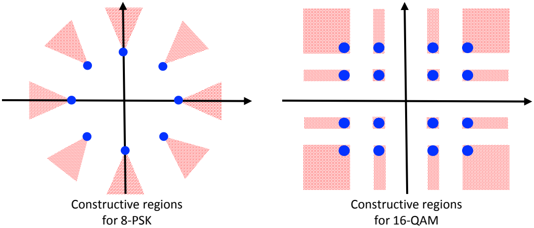


- Target constellation point s is taken from a constellation conventionally QAM or PSK.
- Symbol-by-symbol precoding: $\mathrm{x} = \mathcal{P}(\mathrm{s}, \mathrm{H})$, where $\mathrm{x} \in \mathcal{X}^M = \left\{\frac{1}{\sqrt{2}}(\pm 1 \pm \imath)\right\}^M$.
- Received signal at the k^{th} user: $y_k = \sqrt{\frac{P}{M}}\mathrm{h}_k^H \mathrm{x} + z_k$.
- Signal recovery at the receiver: $\hat{s}_k = \mathcal{Q}(y_k)$.
- **Goal:** Design the receive constellation and precoder $\mathcal{P}(\mathrm{s}, \mathrm{H})$ to minimize average SER.

- The one-bit precoding architecture is an example of symbol-level precoding.
- **Traditional Multiuser Precoding:**
  - Focuses on eliminating interference between different users.
  - Designs precoders only based on channel state information (CSI).
- **Symbol-Level Precoding (SLP):**
  - Exploits constructive interference for enhancing received signal power.
  - Designs precoders by exploiting the knowledge of users' data symbol, in addition to CSI.



- **Symbol-level Precoding Main Idea:**
  - Design precoders such that received symbols for all users lie in the constructive regions.
  - Such a precoding design involves formulating/solving non-trivial optimization problems.
  - The idea of SLP is pioneered in [Alodeh, Chatzinotas, Ottersten, 2015] and [Masouros, G. Zheng, 2015].



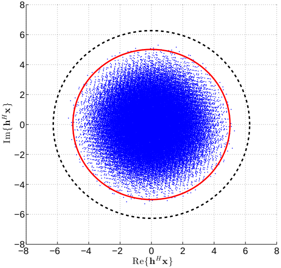
- Most previous works on SLP focus on PSK modulations.
  - This is because the decision boundaries in PSK are easier to characterize.
  - Examples: [Li and Masouros, 2018] and [Law and Masouros, 2018].
- Some recent works consider SLP design for QAM modulations.
  - Examples: [Kalantari et al., 2018] and [Li, Masouros, Li, Vucetic, and Swindlehurst, 2018].

Constellation Design: Single-User MISO Case

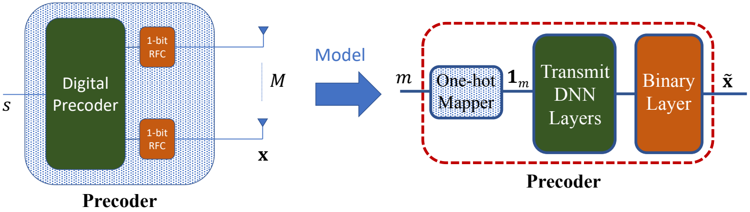
- Precoder design problem given the constellation point s^i:

$$\mathrm{x}_i^* = \underset{\mathrm{x}_i \in \mathcal{X}^M}{\operatorname{argmin}} \left| \sqrt{\frac{P}{M}} \mathrm{h}^H \mathrm{x}_i - s^i \right|. \tag{3}$$

- **Observation:** For a fixed channel, the possible realizations of $\mathrm{h}^H\mathrm{x}$ when $\mathrm{x} \in \mathcal{X}^M$ are distributed densely close to the origin, e.g.,



- [Sohrabi, Liu, Yu '08]: Set the range to be $\sqrt{\frac{2}{\pi}}$, or 80% of the infinite resolution case
- Can we use a neural network to "discover" the optimal constellation and precoder?



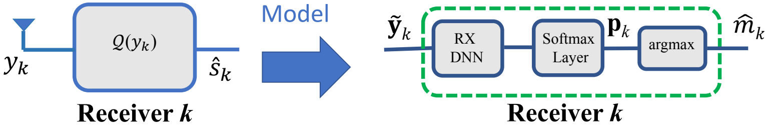
- The real-valued received signal model:

$$\underbrace{\begin{bmatrix} \Re\{y_k\} \\ \Im\{y_k\} \end{bmatrix}}_{\triangleq \atop \widetilde{\mathrm{y}}_k} = \rho \underbrace{\begin{bmatrix} \Re\{\mathrm{h}_k^H\} & -\Im\{\mathrm{h}_k^H\} \\ \Im\{\mathrm{h}_k^H\} & \Re\{\mathrm{h}_k^H\} \end{bmatrix}}_{\triangleq \atop \widetilde{\mathrm{H}}_k} \underbrace{\begin{bmatrix} \Re\{\mathrm{x}\} \\ \Im\{\mathrm{x}\} \end{bmatrix}}_{\triangleq \atop \widetilde{\mathrm{x}}} + \underbrace{\begin{bmatrix} \Re\{z_k\} \\ \Im\{z_k\} \end{bmatrix}}_{\triangleq \atop \widetilde{\mathrm{z}}_k}.$$

- The precoder is modeled by a DNN with T dense layers followed by a binary layer:

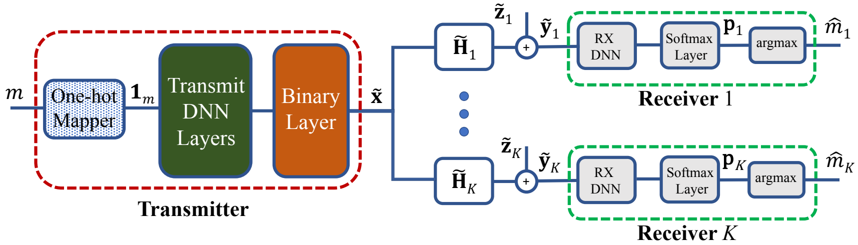
$$\widetilde{\mathrm{x}} = \mathrm{sgn}\left(\mathrm{W}_\mathrm{T}\sigma_{\mathrm{T-1}}\left(\cdots \mathrm{W}_2\sigma_1\left(\mathrm{W}_1 1_m + \mathrm{b}_1\right) + \cdots \mathrm{b}_{\mathrm{T-1}}\right) + \mathrm{b}_\mathrm{T}\right),$$

  - $m \in \{1, \ldots, |\mathcal{C}|\}$ denotes the index of the intended symbol.
  - $1_m \in \mathbb{R}^{|\mathcal{C}|}$ denotes the one-hot representation of m.
  - σ_t is the activation function for the t^{th} layer.
- Binary layer ensures that the one-bit constraints on the elements of $\widetilde{\mathrm{x}}$ are met.



- The receivers' operations are modeled by another DNN with R dense layers.
- Softmax activation function in the last layer:
  - To generate $\mathrm{p}_k \in (0,1)^{|\mathcal{C}|}$, where its i^{th} element indicates the probability that the index of the intended symbol is i.
- Receiver k declares $\hat{m}_k$, which corresponds to the index of largest p_k.
- We consider one common DNN to represent the decoding procedure of different users.
  - Reduces dimensions of the receivers' trainable parameters.
    $\Longrightarrow$ Faster training procedure.
  - The BS needs to broadcast the common constellation parameters to all the users.
    $\Longrightarrow$ Reduction in amount of required feedback.

End-to-End Autoencoder Representation



- As proof of concept, consider the case that a common symbol is sent to multiple users.
- **Input**: Index of the intended symbol.
- **Outputs**: Index of the intended symbol decoded at the receivers.
- After the network being trained for a fixed $\{\widetilde{\mathrm{H}}_k\}_{k=1}^{K}$, we obtain:
  - The precoding procedure at the transmitter.
  - The constellation design and decision boundaries at the receivers.
- How to train this network?
  - SGD-based training via back-propagation.
  - The binary layer is approximated by annealed sigmoid-adjusted straight-through.

Implementation Details

- **Implementation platform:** TensorFlow.
- **Optimization method:** Adam optimizer with an adaptive learning rate initialized to 0.001. .
- $\#$ **hidden layers:** $Tx = 12$ and $Rx = 5$.
- $\#$ **hidden neurons/layer:** $6M$ for the transmitter and $2M$ for the receiver.
- **Activation function of the hidden layers:** Exponential linear units (ELUs).
- **Loss Function:** Cross entropy between 1_m and the probability vectors, p_k:

$$\mathcal{L}_{\text{CE}} = -\mathbb{E}_{\text{training samples}}\left[\frac{1}{K|\mathcal{C}|}\sum_{k=1}^{K}\sum_{m=1}^{|\mathcal{C}|}\log p_{k,m}\right]. \tag{4}$$

- **Annealing parameter update rule:**

$$\alpha^{(i)} = 1.002\alpha^{(i-1)} \tag{5}$$

  with $\alpha^{(0)} = 1$ such that $1.002^{2000} \approx 55$.
- In the training stage, the noise variance is randomly generated so that:

$$\text{SNR} \triangleq 10\log_{10}\left(\frac{P}{2\sigma^2}\right) \in [4dB, 16dB]. \tag{6}$$

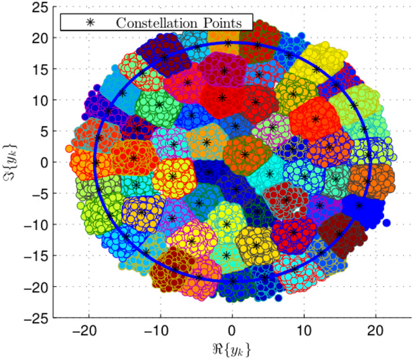


Figure: The receive constellation points and their corresponding decision boundaries obtained from a trained autoencoder in a system with $M = 128$, $K = 4$, and $|C| = 64$.

- The furthest constellation points are located at the following distance from the origin:

$$d^\star = \sqrt{\frac{\frac{2}{\pi}P}{\mathbf{1}^T\left(\mathbf{H}\mathbf{H}^H\right)^{-1}\mathbf{1}}}, \tag{7}$$

matching the heuristic 0.8 constellation range result in [Sohrabi, Liu, Yu '08].

Numerical Results with Varying Channels

- **Constellation range needs to adapt to the channel:**
  - Consider the constellation designed for one particular H.
  - Rescale that constellation for other H so that the constellation range becomes $d^\star$.

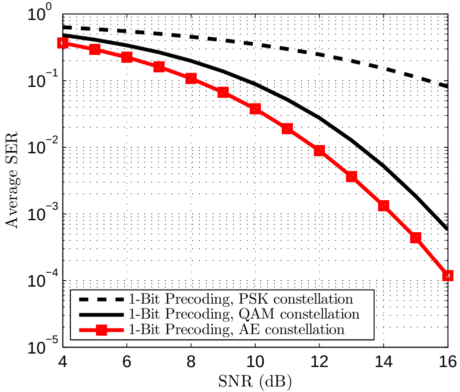
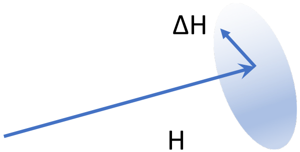


Figure: Average SER versus SNR in a system with $M = 128$, $K = 4$, and $|C| = 64$ using the greedy plus exhaustive search based one-bit precoding algorithm of [Sohrabi, Liu, and Yu, 2018].

Robust Symbol-Level Precoding

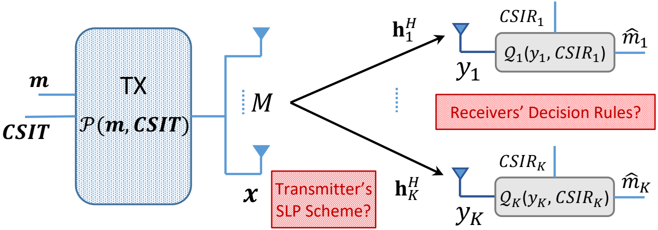
- CSI is never perfect in practice due to several reasons such as:
  - Imperfect channel estimation,
  - Limited/delayed feedback in FDD systems,
  - Mismatch in channel reciprocity in TDD systems.

  $\Longrightarrow$ Robust symbol-level precoding design is crucial.



- A robust SLP scheme has recently been proposed in [Haqiqatnejad, Kayhan, and Ottersten, 2019]:
  - Restricted to spherical bounded model and stochastic Gaussian model.
  - Based on the assumption that CSI uncertainty model is accurate.
- In contrast, a data-driven robust SLP design can implicitly account for channel uncertainty.

Symbol-Level Precoding with CSI Uncertainty



- Target message m_k of B-bits for each user is uniformly taken from $\{1, \ldots, 2^B\}$.
- Symbol-by-symbol precoding: $\mathrm{x} = \mathcal{P}(\mathrm{m}, \mathrm{CSIT})$, satisfying $\|\mathrm{x}\|^2 \leq P$.
- Received signal at the k^{th} user: $y_k = \mathrm{h}_k^H \mathrm{x} + z_k$.
- Message recovery at the k^{th} user: $\hat{m}_k = \mathcal{Q}_k(y_k, CSIR_k)$.
- **Goal:** Design the precoder function $\mathcal{P}(\cdot)$ and the receivers' decision rules $\mathcal{Q}_k(\cdot), \forall k$, to minimize average SER.

CSI Model

- We consider a propagating environment with sparse channels, e.g., mmWave channels:

$$\mathrm{h}_k = \frac{1}{\sqrt{L}} \sum_{\ell=1}^{L} \alpha_{\ell,k} \mathrm{a}_t\left(\theta_{\ell,k}\right),$$

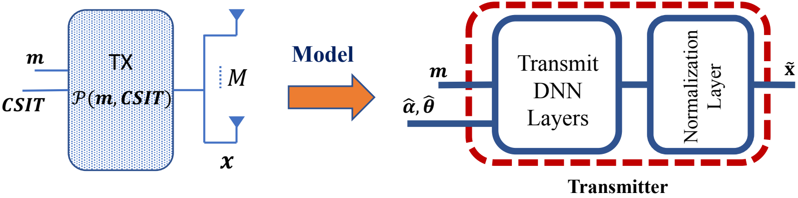
  - L is the number of propagation paths,
  - $\alpha_{\ell,k}$ is the complex gain of the ℓ^{th} path,
  - $\theta_{\ell,k}$ is the AoD of the ℓ^{th} path,
  - $\mathrm{a}_t(\cdot)$ is the array response vector, e.g., $\mathrm{a}_t(\theta) = \left[1, e^{j\pi \sin(\theta)}, \ldots, e^{j\pi(M-1)\sin(\theta)}\right]$.

- We assume that the available CSI is in the form of imperfect estimation of the sparse channel parameters as:

$$\hat{\alpha}_{\ell,k} = \alpha_{\ell,k} + \Delta\alpha_{\ell,k},$$
$$\hat{\theta}_{\ell,k} = \theta_{\ell,k} + \Delta\theta_{\ell,k},$$

  where $\Delta\alpha_{\ell,k} \sim \mathcal{CN}\left(0, \sigma^2_{\Delta\alpha}\right)$ and $\Delta\theta_{\ell,k} \sim \mathcal{U}\left(-\Delta\theta_{\max}, \Delta\theta_{\max}\right)$.

- Summary of the CSI model: $\mathrm{CSIT} = \{\hat{\alpha}_{\ell,k}, \hat{\theta}_{\ell,k}\}_{\forall \ell,k} = \{\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\theta}}\}$
  $CSIR_k = \{\hat{\alpha}_{\ell,k}, \hat{\theta}_{\ell,k}\}_{\forall \ell}$



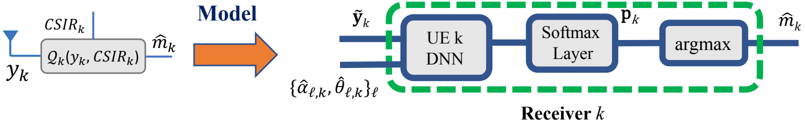
- The real-valued received signal model:

$$\underbrace{\begin{bmatrix} \Re\{y_k\} \\ \Im\{y_k\} \end{bmatrix}}_{\widetilde{y}_k} = \underbrace{\begin{bmatrix} \Re\{h_k^H\} & -\Im\{h_k^H\} \\ \Im\{h_k^H\} & \Re\{h_k^H\} \end{bmatrix}}_{\widetilde{H}_k} \underbrace{\begin{bmatrix} \Re\{x\} \\ \Im\{x\} \end{bmatrix}}_{\widetilde{x}} + \underbrace{\begin{bmatrix} \Re\{z_k\} \\ \Im\{z_k\} \end{bmatrix}}_{\widetilde{z}_k}.$$

- The precoder is modeled by a DNN with T dense layers followed by a normalization layer:

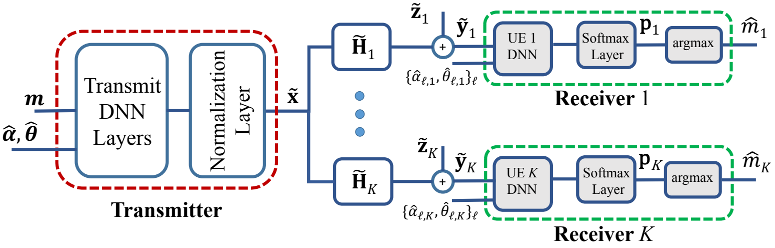
$$\widetilde{x} = \sigma_T \left(W_T \sigma_{T-1} \left(\cdots W_2 \sigma_1 \left(W_1 v + b_1 \right) + \cdots \right) + b_T \right),$$

  - σ_t, W_t, and b_t are the activation function, the weights, and the biases in the t^{th} layer.
  - $v = [\hat{\alpha}, \hat{\theta}, m]$ is the input vector to the DNN.
- Normalization layer, $\sigma_T(x) = \min(\sqrt{P}, \|x\|) \frac{x}{\|x\|}$, ensures that the power constraint is met.



- The receivers' operations are modeled by another DNN with R dense layers.
- Softmax activation function in the last layer:
  - To generate $\mathrm{p}_k \in (0,1)^{|\mathcal{C}|}$, where its i^{th} element indicates the probability that the index of the intended symbol is i.
- Receiver k declares $\hat{m}_k$, which corresponds to the index of largest p_k.

End-to-End Autoencoder Representation



- The BS aims to send independent messages to multiple users.
- **Inputs**: Intended messages and estimated channel parameters.
- **Outputs**: Intended messages recovered at the users.
- After the network is trained for a fixed $\{\widetilde{\mathrm{H}}_k\}_{k=1}^K$, we obtain:
  - The precoding procedure at the transmitter.
  - The decision boundaries at the receivers.
- End-to-End SGD-based training with cross-entropy loss.

Implementation Details

- **Implementation platform:** TensorFlow.
- **Optimization method:** Adam optimizer with an adaptive learning rate initialized to 0.001.
- # **hidden layers:** $T = 4$ and $R = 4$.
- # **hidden neurons/layer:** $[1024, 512, 512, 2M]$ for the transmitter,
  $[256, 128, 64, 2^B]$ for the receivers.
- **Activation function of the hidden layers:** Rectified linear units (ReLUs).
- In the training stage, the noise variance is generated so that:

$$\text{SNR} \triangleq 10\log_{10}\left(\frac{P}{\sigma^2}\right) \in \mathcal{U}(5, 30)dB.$$

- We use 10^5 channel realizations for training and set the CSI parameters as:
  - Linear array with $M = 128$.
  - Single-path, i.e., $L_k = 1, \forall k$.
  - $\alpha_k \sim \mathcal{CN}(0.5 + 0.5\imath, 1)$,
  - $\theta_k \sim \mathcal{U}(\phi_k - 5^\circ, \phi_k + 5^\circ), \forall k$, with $\{\phi_1, \phi_2, \phi_3\} = \{-30^\circ, 0^\circ, +30^\circ\}$,
  - $\sigma_{\Delta\alpha} = 0.001$ and $\Delta\theta_{\max} = 1^\circ$.

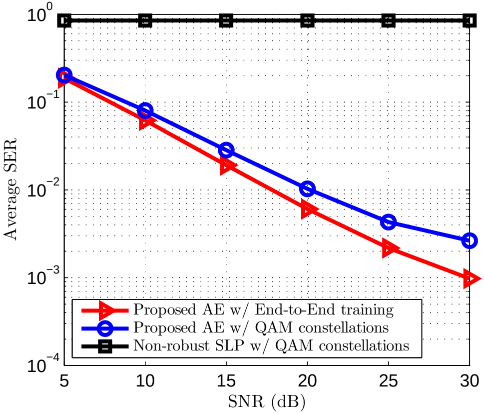


Figure: Avg. SER versus SNR in a system with $M = 128$, $K = 3$, $B = 4$bits, $\Delta\theta_{\text{max}} = 1^\circ$ and $\sigma_{\Delta\alpha} = 0.001$. "Non-robust SLP" corresponds to the SLP algorithm in [Li, Masouros, Li, Vucetic, and Swindlehurst, 2018].

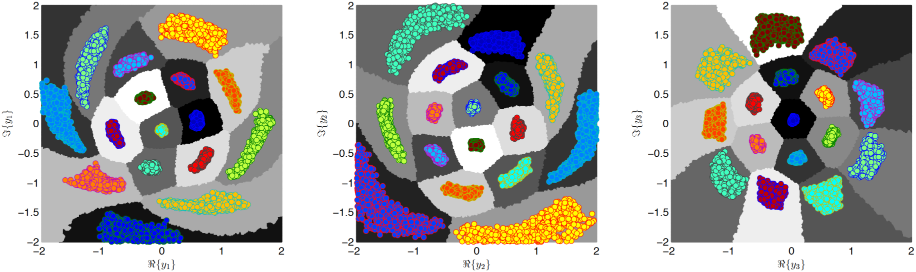


Figure: The decision boundaries (in grey scale) designed by the autoencoder together with the noiseless received signal (as circles) for a robust SLP with $K = 3$ users.

CSI Uncertainty is Explicitly Accounted for in Constellation Design!

Numerical Results: SER Performance vs CSI Uncertainty

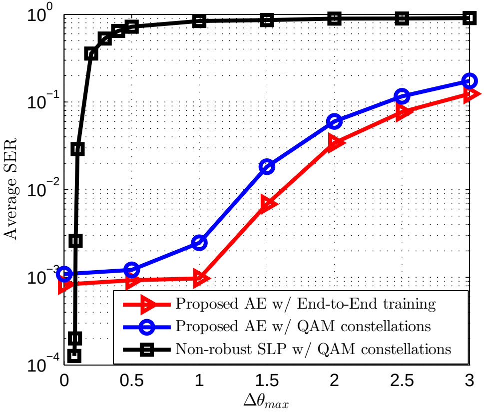


Figure: Avg. SER versus $\Delta\theta_{\max}$ in a system with $M = 128$, $K = 3$, $B = 4$bits, SNR $= 30$dB and $\sigma_{\Delta\alpha} = 0.001$. "Non-robust SLP" corresponds to the SLP algorithm in [Li, Masouros, Li, Vucetic, and Swindlehurst, 2018].

Conclusion and Summary

Summary of Part II

- We propose an end-to-end design for one-bit precoding and for symbol-level precoding.
- We use an DNN autoencoder to jointly design the transciever and the constellation.
- The design account for channel estimation and leads to a more robust receive constellation in a limited scattering environment.

Concluding Remarks:

- Traditional paradigm for communication system design is to model-then-optimize.
- Machine learning allows a data-driven approach that
  - Perform channel estimation, feedback and precoding without explicit channel model;
  - Perform robust precoding and detection without explicit channel uncertainty model.
- Key future issues are: generalizability, training and computational complexity

Further Information

Foad Sohrabi, Kareem M. Attiah, and Wei Yu,
"Deep Learning for Distributed Channel Feedback and Multiuser Precoding in FDD Massive MIMO",
submitted to *IEEE Transactions on Wireless Communications*.

Foad Sohrabi and Wei Yu,
"One-Bit Precoding Constellation Design via Autoencoder-Based Deep Learning",
Asilomar Conference on Signals, Systems, and Computers, Pacific Grove, CA, November 2019.

Foad Sohrabi and Hei Victor Cheng and Wei Yu,
"Robust Symbol-Level Precoding via Autoencoder-Based Deep Learning",
IEEE International Conference on Acoustic, Speech and Signal Processing (ICASSP), 2020.

Foad Sohrabi, Ya-Feng Liu, Wei Yu
" One-Bit Precoding and Constellation Range Design for Massive MIMO with QAM Signaling",
IEEE Journal on Selected Topics in Signal Processing, vol. 12, no. 3, pp. 557-570, June 2018.