Efficient Deep Learning for Massive MIMO Channel State Estimation

PhD Thesis (draft)

Mason del Rosario

Department of Electrical and Computer Engineering

University of California, Davis

Davis, CA 95616, U.S.A.

September, 2022

Contents

Mason del Rosario
September, 2022
Electrical and Computer Engineering

Abstract

Future wireless communications networks will rely heavily on massive MIMO technologies where a base station (BS) with large multiantenna arrays serve a large number of user equipment (UE) terminals. Such multiantenna arrays enable high capacity communications via beamforming, as evidenced by work in information theory [1]. To achieve capacity in massive MIMO networks, the base station requires accurate estimates of the channel state information (CSI) in order to precode (decode) transmitted (received) messages [2].

CSI can be acquired using pilot signals, and in time-division duplex (TDD) mode, channel reciprocity allows the BS to estimate the downlink CSI via pilots in uplink transmissions. However, in frequency division duplex (FDD) mode, channel reciprocity between uplink and downlink channels is weak, and the BS must rely on feedback from the UE to estimate downlink CSI. Specifying an appropriate CSI feedback scheme is a key issue and involves a reducing feedback bandwidth while maintaining accurate downlink CSI estimates.

Conventional methods for CSI feedback compression typically rely on compressed sensing (CS), which seeks to reconstruct high-dimensional data based on low-dimensional measurements (see [3] for a survey of CS methods). Many CS methods rely on convex relaxations of an underdetermined least-squares problem, and such methods rely on iterative solvers (e.g., the proximal gradient method). When using an iterative solver, reconstruction can consume an undue amount of time even when measurements are available, making faster methods for reconstruction desirable.

Recent work in deep learning for compressed CSI estimation has presented viable alternatives to CS methods [4]. Such work typically employs convolutional neural networks (CNNs) to learn compressed representations of high-dimensional CSI, and the architectures

used in CNN-based works can be placed in one of two categories. The first category, CNN-based autoencoders, consists of networks which utilize two subnetworks: an encoder network which learns a low-dimensional representation with the original data as an input and a decoder network which estimates the original data as a function of the low-dimensional representation. The second category, unrolled optimization networks, draws inspiration from the aforementioned CS methods by structuring the CNN as a finite number of repeated blocks, each of which imitates an iteration of a given CS algorithm.

This thesis explores both CNN autoencoders and unrolled optimization networks for CSI estimation while focusing domain knowledge to improve the performance of these CSI estimation networks with respect to accuracy, feedback rate, or network efficiency. Prior works have demonstrated superior performance over these architectures over conventional CS methods [4, 5], and this thesis investigates the myriad ways that domain knowledge in wireless channels and communications protocols can be leveraged to improve CSI estimation.

Chapter 1

Introduction

This dissertation details work in improving the accuracy and efficiency of deep learning methods for MIMO channel state information estimation. This chapter provides the necessary background to understand the contributions of the dissertation. Section 1.1 provides an overview of the MIMO channel and the importance of CSI estimation in MIMO-based communications networks. Section 1.3 discusses MIMO channel models and introduces the primary channel model used in this work, the COST2100 model. Section 1.4 discusses prior work in compressed sensing for CSI estimation. Section 1.5 provides a generic overview of deep learning. Boldface lowercase (uppercase) letters indicate vectors (matrices). Unless otherwise specified, the norm $\|\cdot\|$ indicates the Frobenius norm. Superscripts ^{T} (^{H}) indicate the transpose (Hermitian transpose).

1.1 MIMO Channel Overview

In this work, we consider a MIMO channel with a multiple antennas ($N_B \gg 1$) at the transmitter (gNodeB or gNB) servicing a single user equipment (UE) with a single antenna. Under orthogonal frequency division multiplexing (OFDM) with N_f subcarriers, the received symbols on the m-th subcarrier for the downlink and the uplink at the receiver are

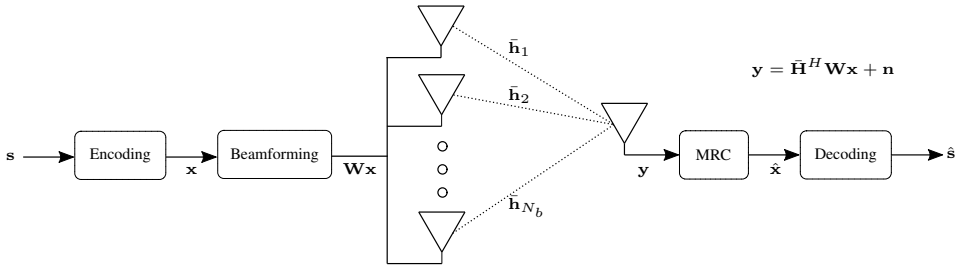


Figure 1.1: Example multi-antenna transmitter (BS, gNB) and single-antenna user equipment (UE) and relevant system values.

Table 1.1: MIMO system variables considered in this work.

Symbol	**Dimension**	**Description**
$y_{d,m}$	$\mathbb{C}^1$	Received downlink symbol on m-th subcarrier
$\mathbf{h}_{d,m}$	$\mathbb{C}^{N_b \times 1}$	Downlink channel on m-th subcarrier
$\bar{\mathbf{H}}_d$	$\mathbb{C}^{N_f \times N_b}$	Downlink CSI (spatial-frequency domain)
$\mathbf{w}_{t,m}$	$\mathbb{C}^{N_b \times 1}$	Transmitter precoding vector for m-th subcarrier
$x_{d,m}$	$\mathbb{C}^1$	Trasmitted symbol on m-th subcarrier
$n_{d,m}$	$\mathbb{C}^1$	Downlink noise on m-th subcarrier
$\tilde{\mathbf{H}}_d$	$\mathbb{C}^{N_f \times N_b}$	Downlink CSI (angular-delay domain)
$\mathbf{H}_d$	$\mathbb{C}^{R_d \times N_b}$	Truncated downlink CSI (angular-delay domain)

given as

$$y_{d,m} = \mathbf{h}_{d,m}^H \mathbf{w}_{t,m} x_{d,m} + n_{d,m}.$$

where the individual system values are defined in Table 1.1, and a representative system model is viewable in Figure 1.1. The resulting downlink and uplink channel state information (CSI) matrices are given as

$$\bar{\mathbf{H}}_d = \begin{bmatrix} \mathbf{h}_{d,1} & \dots & \mathbf{h}_{d,N_f} \end{bmatrix}^H \in \mathbb{C}^{N_f \times N_b}.$$

To achieve near-capacity transmission rates, the transmitter needs access to an appropriate estimate of $\bar{\mathbf{H}}_d$ [1]. Such estimates enable the use of linear precoding techniques (e.g., con-

Table 1.2: Parameters used for COST2100 simulations for both Indoor and Outdoor datasets.

Symbol	Value	Description
N_b	32	Number of antennas at gNB
N_f	1024	Number of subcarriers for OFDM link
R_d	32	Number of delay elements kept after truncation
N	10^6	Total number of samples per dataset
T	10	Number of timeslots
δ	40 ms	Feedback delay interval between consecutive CSI timeslots

jugate beamforming or zero-forcing beamforming) to realize appreciable spectral and power efficiency gains [6]. Downlink CSI estimation can be performed in time division duplex (TDD) by using uplink pilots due to channel reciprocity [7–9]. In contrast, frequency domain duplex (FDD) does not admit channel reciprocity due to frequency-selective channels, meaning CSI estimates must be acquired at the UE using pilot signals, and these estimates must be compressed then fed back to the BS.

1.2 Pilot-based Channel Estimates

1.3 Channel Model

For all CSI tests, we mainly rely on the COST2100 MIMO channel model [10]. We use two datasets with a single base station (gNB) and a single user equipment (UE) in the following scenarios:

(1) **Indoor** channels using a 5.3GHz downlink at 0.001 m/s UE velocity, served by a gNB at center of a $20\text{m}\times20\text{m}$ coverage area.

(2) **Outdoor** channels using a 300MHz downlink at 0.9 m/s UE velocity served by a gNB at center of a $400\text{m}\times400\text{m}$ coverage area.

In both scenarios, we use the parameters listed in Table 1.2.

1.4 Classical CSI Estimation

Works in compressive feedback for CSI estimation in MIMO networks can be placed in three broad categories. The first category includes works which use direct quantization of continuous CSI elements to discrete levels. The quantized CSI are encoded and fed back to the transmitter [11, 12]. The second category includes works which use compressed sensing, a technique which applies a random measurement matrix at the transmitter and the receiver [13, 14]. Compressed sensing assumes matrices to be encoded and fed back meet certain sparsity requirements, and compressed sensing algorithms require iterative solvers [15] for decoding, resulting in undesired latency.

The last category of work in compressive CSI feedback uses deep learning (DL), neural networks with numerous layers which are trained on large datasets using backpropagation. Before describing these works, we first describe a few pertinent concepts from deep learning.

1.5 Deep Learning Background

This section provides a brief overview of relevant deep learning concepts employed in this work, including convolutional neural networks (CNNs), autoencoders, and unsupervised learning.

Deep learning (DL) is a subset of machine learning (ML), a broad class of algorithms which use data to "fit" models for prediction or classification tasks. The three predominant learning frameworks are supervised learning, unsupervised learning, and reinforcement learning. In the works proposed, we focus on *unsupervised learning*, which seeks to find a compressed representation of the data without labels (see Chapter 14 of [16] for an overview).

Convolutional Neural Networks: A neural network is a machine learning algorithm with multiple *layers* of parameterized linear functions followed nonlinear functions (typically referred to as 'activation' functions). The parameters for these layers can be updated

via a stochastic optimizer (e.g., [17]), and given enough layers, such networks can achieve arbitrarily accurate functional approximation [18]. In recent years, neural networks with convolutional layers have established state-of-the-art performance in computer vision tasks such as image classification [19] and segmentation [20].

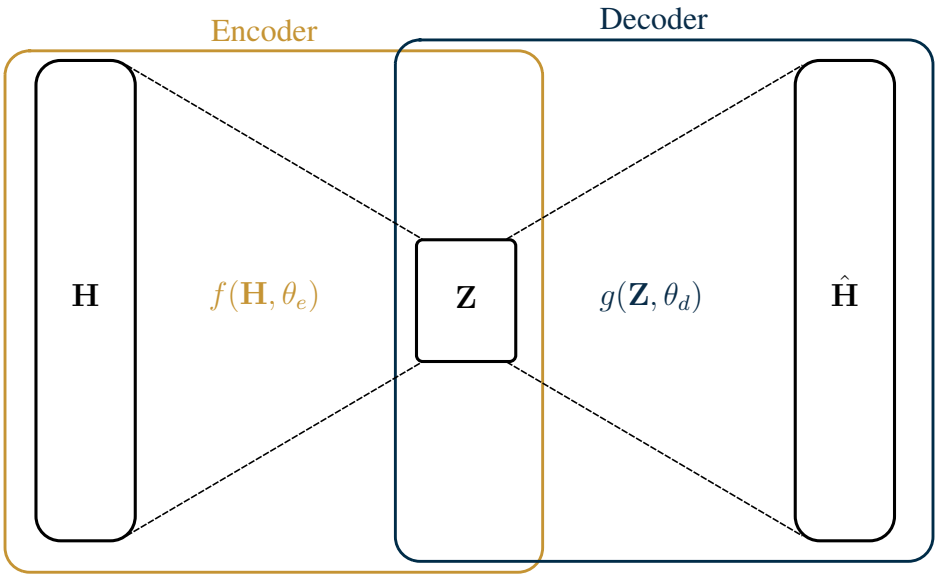


Figure 1.2: Abstract schematic for an autoencoder operating on CSI matrices $\mathbf{H}$. The encoder learns a latent representation, $\mathbf{Z}$, while the decoder learns to reconstruct estimates $\hat{\mathbf{H}}$.

A common architecture for deep unsupervised learning is the *autoencoder* (see Fig. 1.2 for a generic example). Trained end-to-end on input data, an autoencoder is comprised of an encoder and a decoder which jointly learn a compressed latent representation ($\mathbf{Z}$) and an estimate of the input ($\hat{\mathbf{H}}$). By choosing $\mathbf{Z}$ to have lower dimension than the input, the network is forced to learn a "useful" summary of the input data. The typical objective function for such a network is the mean squared error (MSE),

$$\underset{\theta_e, \theta_d}{\operatorname{argmin}} \frac{1}{N} \sum_{i=1}^{N} \|\mathbf{H}_i - g(f(\mathbf{H}_i, \theta_e), \theta_d)\|^2.$$

We optimize network parameters $\vec{\theta}_e, \vec{\theta}_d$ by backpropagation and a stochastic optimization algorithm (e.g., stochastic gradient descent, ADAM).

1.6 Objective and Contributions

Successful efforts in DL for CSI estimation have typically utilized convolutional neural networks (CNNs) in an autoencoder structure [4]. Variations on the CNN-based autoencoder have investigated different network architectures [21], variational training frameworks [22], and denoising modules [23]. These architectural changes are largely inspired by successful application of DL in image compression [24–26].

While they can continue to push the state-of-the-art in CSI reconstruction accuracy, architectural optimizations may ultimately follow the same trends of fields such as language modeling, where state-of-the-art performance requires prohibitively massive compute [27]. In this proposal, we take a different approach seek to improving compressive channel feedback by focusing on domain knowledge and physical insight.

This qualifying exam proposal details our attempts to use domain knowledge to enhance the performance and the efficiency of neural networks for CSI estimation (for a visual summary, see Figure 1.3). Section 2 details our work in power-based normalization, which leverages CSI sparsity. Section 3 describes our work in differential encoding, which exploits temporal coherence of CSI. Section 4 describes our work in pilot-based delay domain CSI estimation.

Figure 1.3: Venn diagram highlighting different aspects of domain knowledge in CNN-based CSI compressive feedback, relevant convolutional networks, and our contributions.

Chapter 2

Data Pre-processing and Normalization

In this chapter, we will discuss the data pre-processing techniques and the applications of domain knowledge that have enabled successful application of deep learning to MIMO CSI estimation (Section 2.1), including our proposed pre-processing technique, spherical normalization (Section 2.3).

2.1 Data Pre-processing for CSI Data

The success of machine learning tasks relies on proper *data pre-processing*, a sequence of transformations used on the input data before fitting a model. In any machine learning task, data pre-processing is necessary to ensure that the scales of input features are similar. In deep learning, three important pre-processing techniques are domain transformations, truncation, or normalization, and here we will explore the choices in pre-processing that different authors have made based on domain knowledge of MIMO CSI data.

2.1.1 Sparse Basis for CSI

The first type of data pre-processing we consider is a domain transformation, the discrete Fourier transform in particular. While the **spatial-frequency** representation $\bar{\mathbf{H}}$ is used for

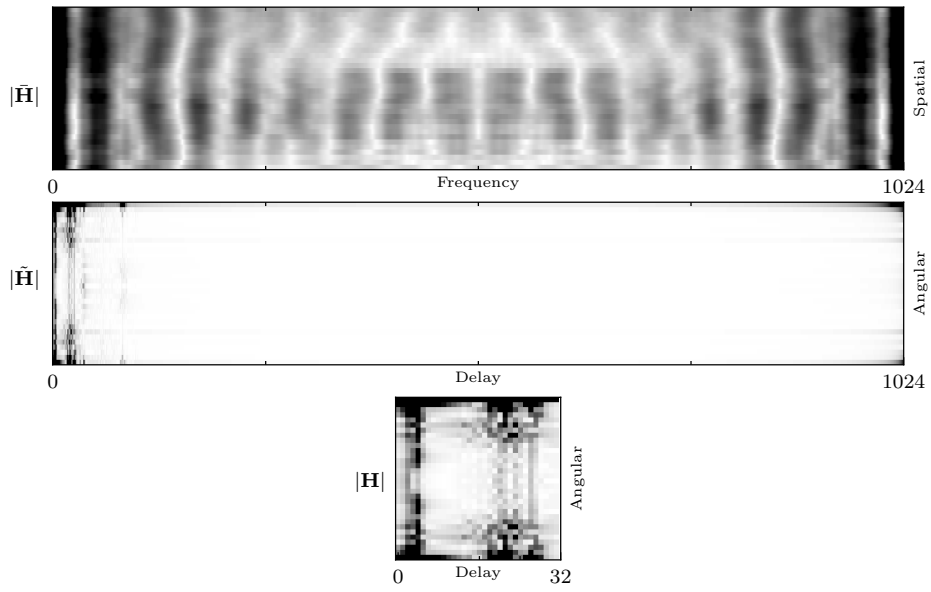


Figure 2.1: Magnitude of spatial-frequency ($\bar{\mathbf{H}}$), angular-delay ($\tilde{\mathbf{H}}$), and truncated angular-delay ($\mathbf{H}$) representations for a single random channel from the outdoor COST2100 dataset.

beamforming at the transmitter, the number of non-zero elements is comparatively large. Given the dimension of $\bar{\mathbf{H}}$, feeding back entire CSI matrices is impractical. Instead, we seek a compressed representation of a sparse transformation. The sparse representation we consider is the angular-delay representation of CSI matrices [28]. Denote the unitary inverse DFT for the spatial (frequency) axis as $\mathbf{F}_a \in \mathbb{C}^{N_b \times N_b}$ ($\mathbf{F}_d^H \in \mathbb{C}^{N_f \times N_f}$), and denote the spatial-frequency CSI matrix as $\bar{\mathbf{H}}$. The angular-delay domain representation $\tilde{\mathbf{H}}$ is given as

$$\tilde{\mathbf{H}} = \mathbf{F}_d^H \bar{\mathbf{H}} \mathbf{F}_a.$$

The delay spread of the resulting $\tilde{\mathbf{H}}$ can typically be captured with a small number of delay elements (see Figure 2.2), so we restrict our attention to the first R_d elements of $\tilde{\mathbf{H}}$, resulting in a truncated angular-delay matrix which we denote as $\mathbf{H} \in \mathbb{C}^{(R_d \times N_b)}$ for the downlink channel state. An illustrative example of this truncation can be seen at the bottom of Figure 2.1.

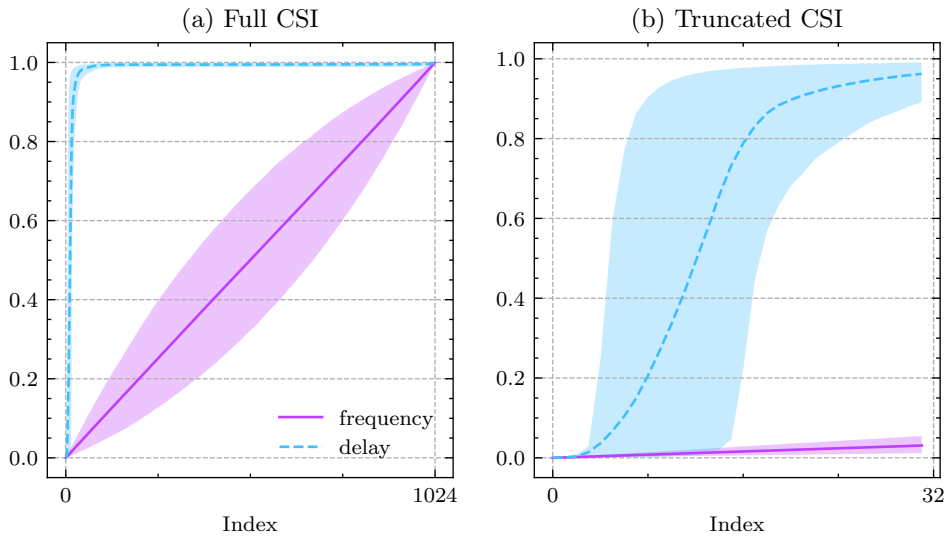


Figure 2.2: Energy CDF for 5000 CSI samples of 32 antennas and 1024 subcarriers, generated from COST2100 outdoor models described in Section 1.3. Mean percentage of energy in CSI matrix up to index is shown with 90% confidence intervals. The index denotes the amount of energy accounted for up to the corresponding frequency/delay element. The truncated angular-delay CSI contains a mean energy of 96.2% (c.i. 89.2%, 99.1%), while the truncated frequency-spatial CSI only contains a mean energy of 3.1% (c.i. 1.2%, 5.4%).

2.1.2 Bidirectional Reciprocity in FDD Networks

The next type of pre-processing under consideration is a change of coordinates. Specifically, rather than utilizing a Cartesian representation (i.e., real-imaginary channels), we can consider a polar representation (i.e., magnitude-phase). As discussed in Section 1.1, the reciprocity of downlink and uplink channels is weak in FDD wireless networks when compared to TDD. Despite this, DL CSI estimation techniques have used uplink CSI to improve the reconstruction accuracy of downlink CSI at gNB. In [29], the authors demonstrate that the correlation between the magnitude of uplink and downlink CSI elements is strong. To exploit magnitude reciprocity, they propose DualNet, a CNN autoencoder which learns a feedback encoding for the downlink CSI magnitude and decodes the feedback with the magnitude of uplink CSI as side information. The downlink phase is separately quantized and fed back to gNB via magnitude-dependent phase quantization (MDPQ). The authors demonstrate that exploiting bidirectional reciprocity can substantially improve CSI estimation accuracy.

2.1.3 Minmax Normalization

The last pre-processing technique we discuss is normalization. Typical deep autoencoders require normalized data to ensure that the range of the input data matches the range of the autoencoder's output function, which is typically chosen as `sigmoid` or `tanh` as pictured in Figure 2.3. To accommodate such output functions, most works in both image

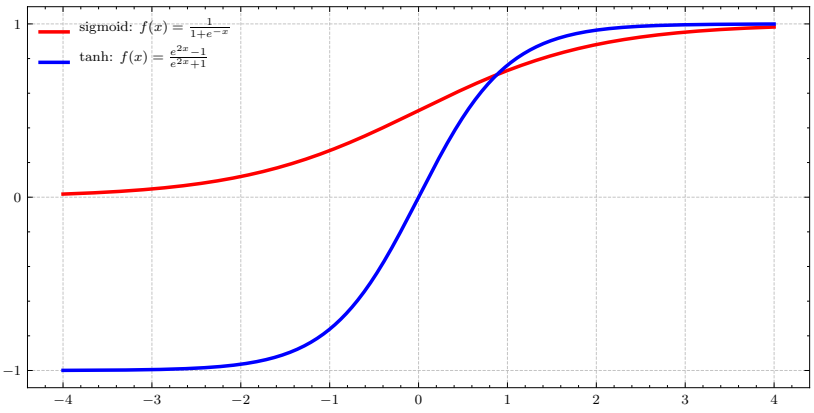


Figure 2.3: Typical activation functions used at the output of convolutional autoencoders.

compression and CSI estimation typically apply *minmax normalization*, where the extrema (i.e., the minimum and the maximum) of the real and imaginary channels are used to scale the entire dataset. For the scalar $H_n(i,j)$, the minmax-scaled version of this element is

$$H_{n,\text{minmax}}(i,j) = \frac{H_n(i,j) - H_{\text{min}}}{H_{\text{max}} - H_{\text{min}}} \in [0,1],$$

for $n \in [1,\ldots,N]$ given a dataset of N samples and i/j indexing the rows/columns of the CSI matrices. The resulting samples are cast to the range $[0,1]$.

For image data, minmax normalization results in each image's color channels scaled to the range $[0,1]$. The resulting distribution for each color channel is typically satisfactory for image tasks, as the variance is not much smaller than the range of the normalized data (see Fig. 2.4).

However, for CSI matrices, minmax normalization is applied to the real and imaginary channels of each element. For typical channel models and parameters, the distribution of

channel elements tends to have much lower variance than that of image data (see Fig. 2.5). This smaller variance can be explained by the difference in the datasets' ranges – while the channels in image data (e.g., ImageNet) assume integer values between $[0, 255]$, the channels in CSI data (e.g., COST2100) assume floating point values smaller than 10^{-3}.

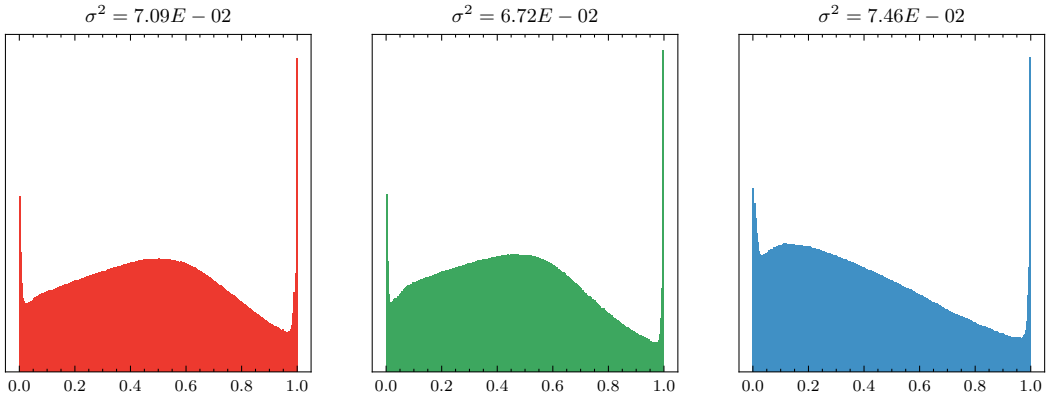


Figure 2.4: Distribution and variance of minmax-normalized ImageNet color channels ($N = 50000$) images.

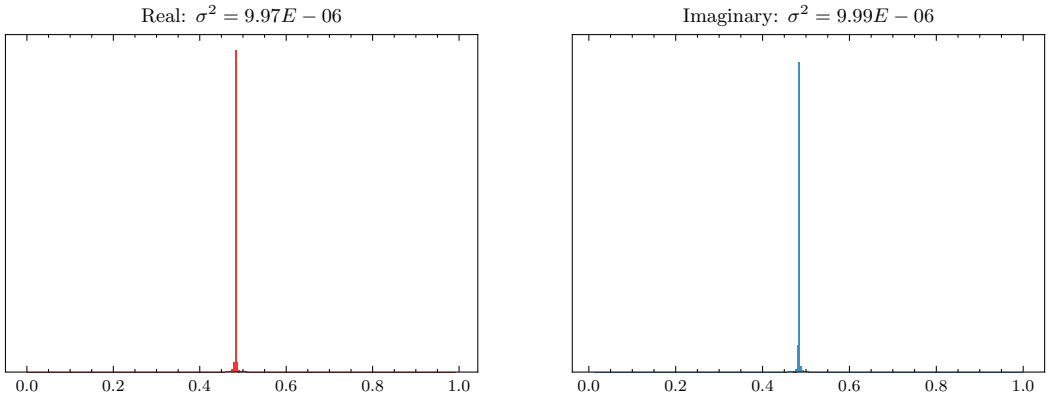


Figure 2.5: Distribution and variance of minmax-normalized COST2100 real/imaginary channels ($N = 99000$) images.

2.2 Related Work

In image processing, several works have investigated normalization techniques such as batch normalization [30], instance normalization [31], layer normalization [32], and group

normalization [33]. These normalization techniques scale the outputs of latent layers in neural networks, which helps to solve the problem of covariate shift [30] where the mean and variance of changes between subsequent layers of the network.

Other works have studied normalization of the network's inputs. A number of works have investigated adaptive normalization techniques for time series estimation tasks [34–36]. In [37], the authors proposed a trainable input network which learns to shift, scale, and filter the unnormalized data while training the target network for a time series prediction task.

2.3 Spherical Normalization

Here, we discuss our work in spherical normalization (Section 2.3) and our optimized network architecture, CsiNet-Pro (Section 2.3.1) [38].

Rather than apply minmax normalization, which is adversely impacted by outliers, we propose spherical normalization. Before describing spherical normalization in detail, consider z-score normalization. Given a random variable, x, with mean μ and standard deviation σ. The z-score normalized version of this random variable is given as

$$z = \frac{x - \mu}{\sigma^2}. \tag{2.1}$$

Assuming x is normally distributed, the resulting random variable, z, is a standard normal distribution such that $z \sim \mathcal{N}(0, 1)$. Inspired by z-score normalization, we seek a normalization scheme which adjusts the range of each channel sample. Under spherical normalization, each sample in the dataset is scaled by its power. Denote the n-th downlink CSI matrix of the dataset as $\mathbf{H}_d^n$. The spherically normalized version of the downlink CSI is given as

$$\check{\mathbf{H}}_d^n = \frac{\mathbf{H}_d^n}{\|\mathbf{H}_d^n\|}. \tag{2.2}$$

Observe that (2.2) is similar to (2.1) without the mean shift in the numerator[1] and with the power term of each CSI sample rather than the variance of the entire distribution. After applying (2.2) to each sample, minmax scaling is applied to the entire dataset. The resulting dataset under spherical normalization can exhibit a larger variance than the same dataset under minmax scaling (compare Fig. 2.6 with Fig. 2.5).

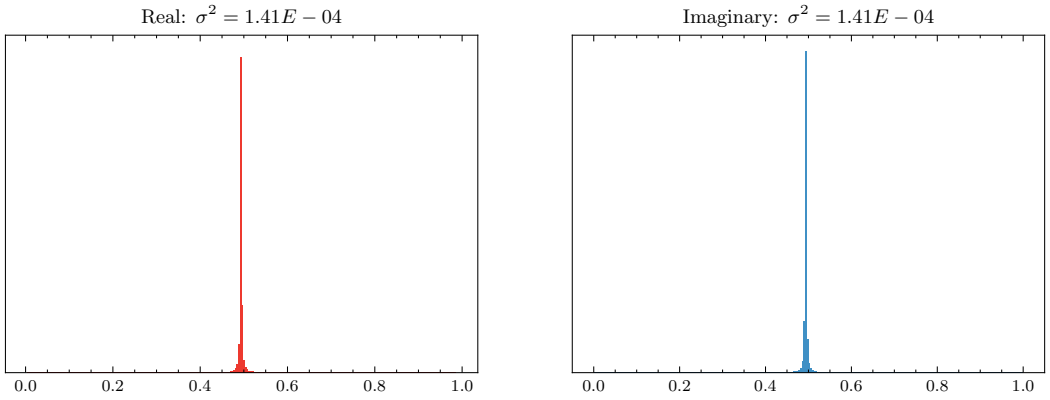


Figure 2.6: Distribution and variance of COST2100 real/imaginary channels under spherical normalization ($N = 99000$) images.

Beyond desirable properties in the input distribution, spherical normalization also results in an objective function which is better matched with the evaluation criterion. Neural networks for CSI estimation are optimized using the mean-squared error loss,

$$\text{MSE} = \frac{1}{N}\sum_{k=1}^{N}\|\mathbf{H}_k - \hat{\mathbf{H}}_k\|^2, \tag{2.3}$$

while channel state reconstruction accuracy is measured in terms of normalized mean-squared error,

$$\text{NMSE} = \frac{1}{N}\sum_{k=1}^{N}\frac{\|\mathbf{H}_k - \hat{\mathbf{H}}_k\|^2}{\|\mathbf{H}_k\|^2}. \tag{2.4}$$

[1]Since the mean of COST2100 data is $\approx 10^{-10}$, we can safely ignore this mean shift in spherical normalization.

Observe that when the $\mathbf{H}_k$ ($\hat{\mathbf{H}}_k$) in (2.3) is replaced with $\check{\mathbf{H}}_k$ ($\hat{\check{\mathbf{H}}}_k$), we have

$$\frac{1}{N}\sum_{k=1}^{N}\|\check{\mathbf{H}}_k-\hat{\check{\mathbf{H}}}_k\|^2=\frac{1}{N}\sum_{k=1}^{N}\left\|\frac{\mathbf{H}_k}{\|\mathbf{H}_k\|^2}-\frac{\hat{\mathbf{H}}_k}{\|\mathbf{H}_k\|^2}\right\|^2$$
$$=\frac{1}{N}\sum_{k=1}^{N}\frac{\|\mathbf{H}_k-\hat{\mathbf{H}}_k\|^2}{\|\mathbf{H}_k\|^2},$$

which is equivalent to (2.4). Thus, a neural network optimized with MSE as the loss function and trained using spherically normalized data is in fact being optimized with respect to NMSE of the original data.

2.3.1 CsiNet-Pro

In [38], we proposed a network with larger convolutional kernels and no residual connections called CsiNet-Pro. Large kernels (e.g., (7×7) in CsiNet-Pro) allow the network to capture features corresponding to larger delay spreads than comparatively small kernels (e.g., (3×3) in CsiNet [4]). In addition to the compressed feedback of the autoencoder, the encoder must feedback the power of the CSI matrix, $\|H\|$, meaning the number of floating point elements to feed back increases from r to $r+1$.

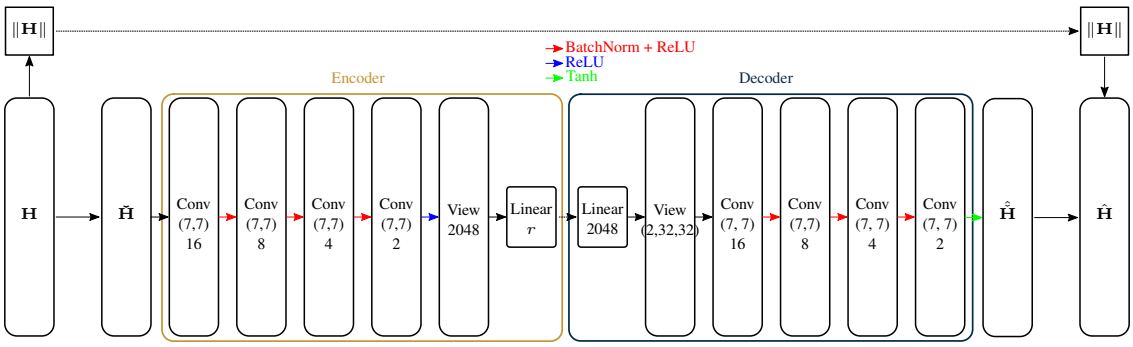
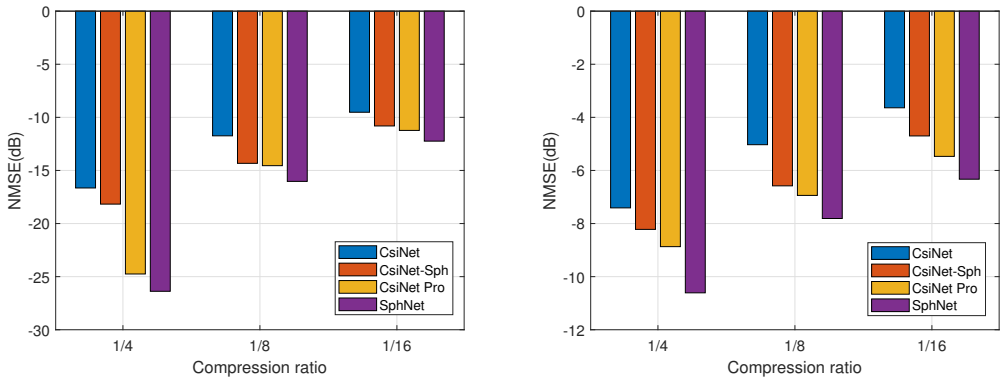


Figure 2.7: SphNet – CsiNet-Pro architecture with Spherical Normalization.

2.3.2 Results

Training on spherically normalized data and optimizing with respect to NMSE can yield better accuracy. Fig. 2.8 demonstrates this improvement for CsiNet and CsiNet-Pro on the COST2100 dataset. CsiNet and CsiNet-Pro are trained with minmax normalization while CsiNet-Sph and SphNet are trained with spherical normalization.



(a) Indoor (b) Outdoor

Figure 2.8: Reconstruction error for CsiNet [4] and CsiNet-Pro with and without spherical normalization. SphNet combines CsiNet-Pro with spherical normalization [38].

Chapter 3

Temporal Coherence

In this chapter, we consider methods for exploiting temporal correlation between CSI of subsequent timeslots. The *coherence time* of a channel is the amount of time that a channel estimate can be used before that estimate's SNR falls beneath a given threshold [39]. Within this window of time ($\Delta t = t_i - t_{i-1}$), the correlation between CSI matrices $\mathbf{H}_i$ and $\mathbf{H}_{i-1}$ is high (see Figure 3.1 for an illustrative example).

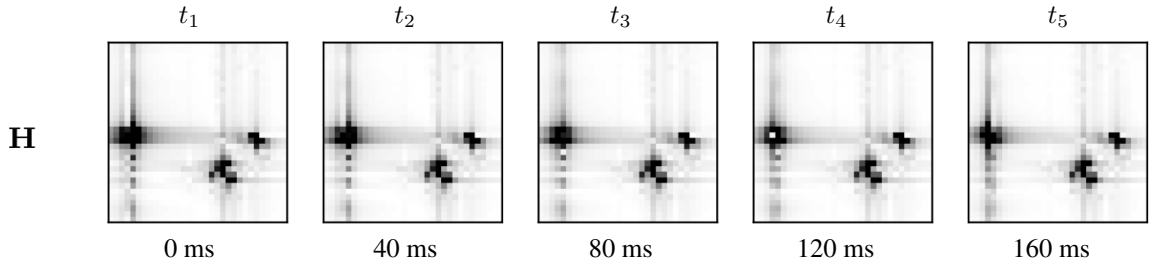


Figure 3.1: Ground truth CSI ($\mathbf{H}$) for five timeslots (t_1 through t_5) on one sample from the validation set of the outdoor dataset.

Assuming the channel exhibits temporal coherence within a certain window of time, a reasonably accurate CSI estimate at time t_{i-1} can be used to estimate the CSI at time t_i. Generically, we can write this estimator as

$$\grave{\mathbf{H}}_i = h(\hat{\mathbf{H}}_{i-1}) \tag{3.1}$$

where $\mathbf{H}_i$ is the CSI matrix at time t_i and $\hat{\mathbf{H}}_i$ is its estimator. The estimation error under $\grave{\mathbf{H}}_i$ is

$$\mathbf{E}_i = \mathbf{H}_i - \grave{\mathbf{H}}_i. \tag{3.2}$$

3.1 Recurrent Neural Networks

Prior work in temporal correlation for CSI estimation utilized state-space methods such as the Kalman filter [40–42]. Since it relies on explicit state space and noise models, the Kalman filter's predictive power in CSI estimation is limited. Furthermore, such work generally does not propose a method for feedback compression, making comparison with the following ML methods difficult.

Recent works have leveraged recurrent neural networks (RNNs) to exploit temporal correlation for CSI estimation [43–47]. RNNs include recurrent layers, such as the long short-term memory (LSTM) cell or the gated recurrent unit (GRU), which are capable of learning long-term dependencies of a given process through backpropagation [48] and can be used to predict future states of the process [49].

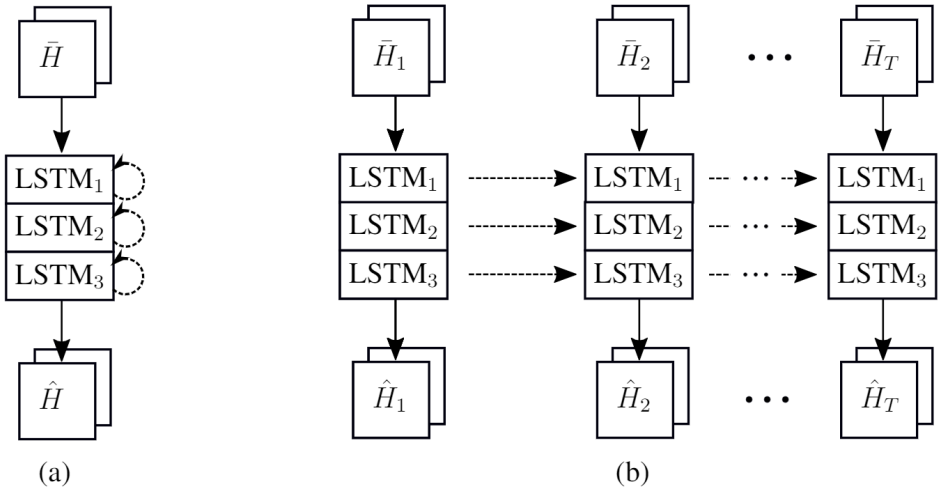


Figure 3.2: An example of LSTMs used for CSI estimation. (a) "Stacked" LSTM network of depth 3 shown with recurrent connections. (b) Same LSTM network "unrolled" into T timeslots

RNNs have been used extensively in natural language processing (NLP) for machine translation [50] and sentiment extraction [51]. For such works in NLP, authors have empirically found "stacked" or "deep" RNNs to be effective (e.g., Fig. 3.2), hypothesizing that having multiple recurrent layers allows the network to extract different semantic timescales [51, 52]. Works in CSI estimation have taken cues from this work in NLP, proposing CSI estimation networks with stacked LSTMs after a sequence of autoencoders [47]. While such work has demonstrated the utility of RNNs, the computational cost of LSTMs can be prohibitively high. For example, the RNN portion of the network proposed in [47] accounts for 10^8 additional parameters. Since channel estimation should not place an undue computational burden on the communications system, LSTMs can be problematic.

3.2 Differential Encoding

Rather than use RNNs to extract temporal dependencies in CSI data, we proposed a lightweight network based on the principle of differential encoding. We trained a network to estimate the error (3.2) under a linear estimator,

$$\grave{\mathbf{H}}_i = \hat{\mathbf{H}}_{i-1}\mathbf{W}$$

where $\mathbf{W} \in \mathbb{C}^{R_b \times R_b}$ is the minimum mean squared error (MMSE) estimator,

$$\begin{aligned}\mathbf{H}_i &= \mathbf{H}_{i-1}\mathbf{W} + \mathbf{E}_i \\ \mathbf{H}_{i-1}^H\mathbf{H}_i &= \mathbf{H}_{i-1}^H\mathbf{H}_{i-1}\mathbf{W} + \cancelto{\mathbf{0}}{\mathbf{H}_{i-1}^H\mathbf{E}_i}\end{aligned}$$

where the cancellation of the product $\mathbf{H}_{i-1}^H\mathbf{E}_i$ is due to the principle of orthogonality (i.e., the error terms are orthogonal to the observed data). Denoting the cross correlation matrix

as $\mathbf{R}_i = \mathbb{E}\left[\mathbf{H}_{t-i}^H \mathbf{H}_t\right]$, we solve for the MMSE estimator,

$$\mathbf{W} = \mathbf{R}_0^{-1}\mathbf{R}_1.$$

In practice, the population correlation matrices are estimated via finite samples of size N,

$$\hat{\mathbf{R}}_k = \frac{1}{N}\sum_{j}^{N} \mathbf{H}_{i-k}^H(j)\mathbf{H}_i(j),$$

where $\mathbf{H}_i(j)$ is the j-th sample in the training set. The MMSE estimator based on the sample correlation matrices is written as

$$\hat{\mathbf{W}} = \hat{\mathbf{R}}_0^{-1}\hat{\mathbf{R}}_1.$$

We can further simplify this estimator to a scalar, $\gamma \in \mathbb{R}$, as

$$\hat{\gamma} = \frac{\text{Trace}(\hat{\mathbf{R}}_1(k,l))}{\sum_k^{R_d}\sum_l^{N_b}\hat{\mathbf{R}}_0(k,l)},$$

where k (l) are the row (column) indices of the correlation matrices. The estimator in this case is

$$\grave{\mathbf{H}}_i = \hat{\gamma}\hat{\mathbf{H}}_{i-1}. \tag{3.3}$$

Under the estimator γ, we proposed to encode the error, $\mathbf{E}_t$, using a convolutional autoencoder, $f(\mathbf{E}_t)$,

$$\hat{\mathbf{E}}_i = g(f(\mathbf{E}_i, \vec{\theta}_e), \vec{\theta}_d),$$

where $\mathbf{E}_i = \mathbf{H}_i - \gamma\hat{\mathbf{H}}_{i-1}$. The base station has access to the estimators γ and $\hat{\mathbf{H}}_{i-1}$, and the resulting CSI estimate at t_i is

$$\hat{\mathbf{H}}_i = \hat{\gamma}\hat{\mathbf{H}}_{i-1} + \hat{\mathbf{E}}_i \tag{3.4}$$

3.2.1 MarkovNet

In [53], we proposed MarkovNet, a deep differential autoencoder. Each timeslot of MarkovNet uses an instance of CsiNet-Pro with unique parameters. The network at the first timeslot (t_1) is trained directly on the CSI ($\mathbf{H}_1$). For all subsequent timeslots, t_i for $i \geq 2$, we use the MMSE estimator (3.3) to produce an error term $\mathbf{E}_t$, and the autoencoder in each timeslot is trained to produce an error estimate, $\hat{\mathbf{E}}_t$. The estimated error is added back per (3.4) to produce a refined estimate.

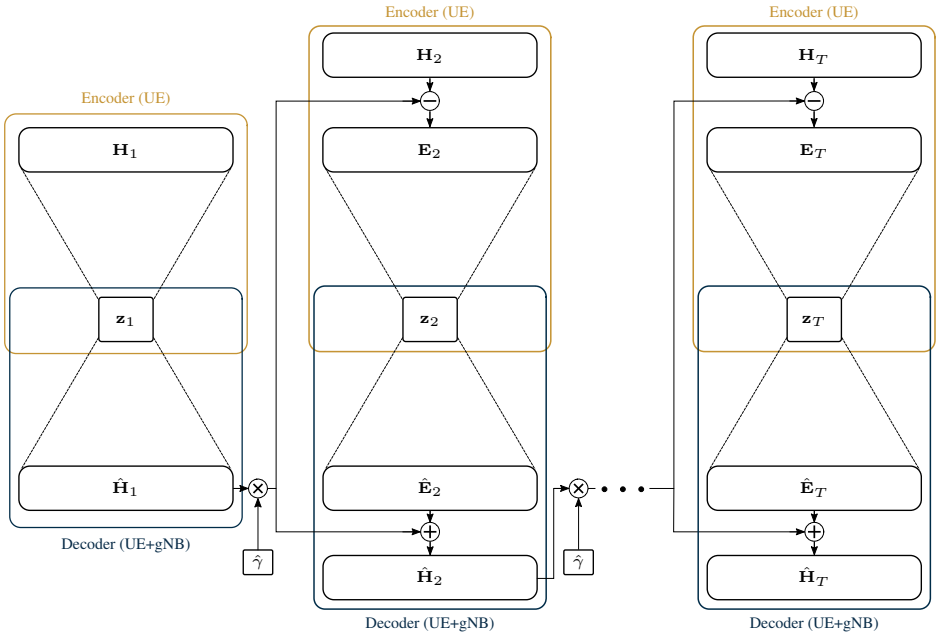


Figure 3.3: Abstract architecture for MarkovNet. Networks at t_i for $i \geq 2$ are trained to predict the estimation error, $\mathbf{E}_i$.

The resulting network requires no recurrent layers, resulting in a substantial reduction in computational complexity. Table 3.1 shows the number of parameters and FLOPs per timeslot for CsiNet-LSTM, MarkovNet, and CsiNet. The parameter count of MarkovNet

is on par with CsiNet, and CsiNet-LSTM requires orders of magnitude more parameters. While the number of FLOPs for MarkovNet is nearly 10 times smaller than CsiNet-LSTM, MarkovNet requires 5 to 10 times more FLOPs than CsiNet due to the increased kernel size of CsiNet-Pro.

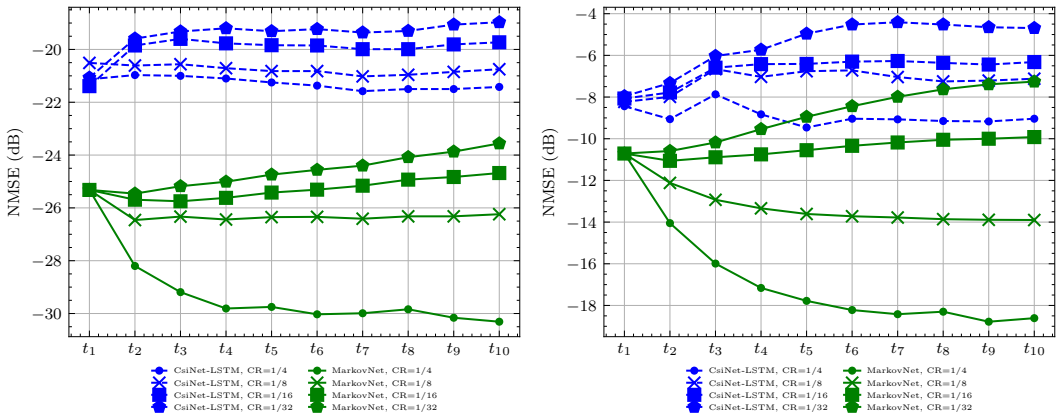
Table 3.1: Model size/computational complexity of tested temporal networks (CsiNet-LSTM, MarkovNet) and comparable non-temporal network (CsiNet). M: million.

	Parameters			**FLOPs**		
	CsiNet-LSTM	**MarkovNet**	**CsiNet**	**CsiNet-LSTM**	**MarkovNet**	**CsiNet**
CR=$1/4$	132.7 M	2.1 M	2.1 M	412.9 M	44.5 M	7.8 M
CR=$1/8$	123.2 M	1.1 M	1.1 M	410.8 M	42. 4 M	5.7 M
CR=$1/16$	118.5 M	0.5 M	0.5 M	409.8 M	41.3 M	4.7 M
CR=$1/32$	116.1 M	0.3 M	0.3 M	409.2 M	40.8 M	4.1 M
CR=$1/64$	115.0 M	0.1 M	0.1 M	409.0 M	40.5 M	3.9 M

3.2.2 Results

We compare MarkovNet with CsiNet-LSTM [47] on the indoor and outdoor COST2100 datasets (for details, see Section 1.3). For MarkovNet, we train the network at the first timeslot for 1000 epochs. In each subsequent timeslot, we initialize the network using the weights from the previous timeslot and train for 200 epochs. We use a batch size of 200. We perform a training/testing split of 75k/25k samples, and we estimate γ using the training set. To compare the estimation accuracy of each network, we report the NMSE.

Figure 3.4 shows the NMSE of MarkovNet and CsiNet-LSTM for four different compression ratios. For the indoor network, all instances of MarkovNet achieve lower NMSE than all instances of CsiNet-LSTM. In the outdoor scenario, each CR for MarkovNet demonstrates lower NMSE than the corresponding CR for CsiNet-LSTM. Between both channel scenarios, MarkovNet shows gradual improvement for subsequent timeslots if the CR is high enough while CsiNet-LSTM only improves gradually in the outdoor environment for CR$= \frac{1}{4}$. Figure 3.5 shows a random sample from the test set, $\mathbf{H}$, and the estimates produced by CsiNet-LSTM and MarkovNet for a CR of $\frac{1}{4}$. This sample contains three "peak" magni-



(a) Indoor (b) Outdoor

Figure 3.4: NMSE comparison of MarkovNet and CsiNet-LSTM at various compression ratios (CR).

tude regions. While both networks manage to capture the two larger samples, MarkovNet is able to recover the small peak magnitude region (green arrow) which CsiNet-LSTM fails to produce (red arrow).

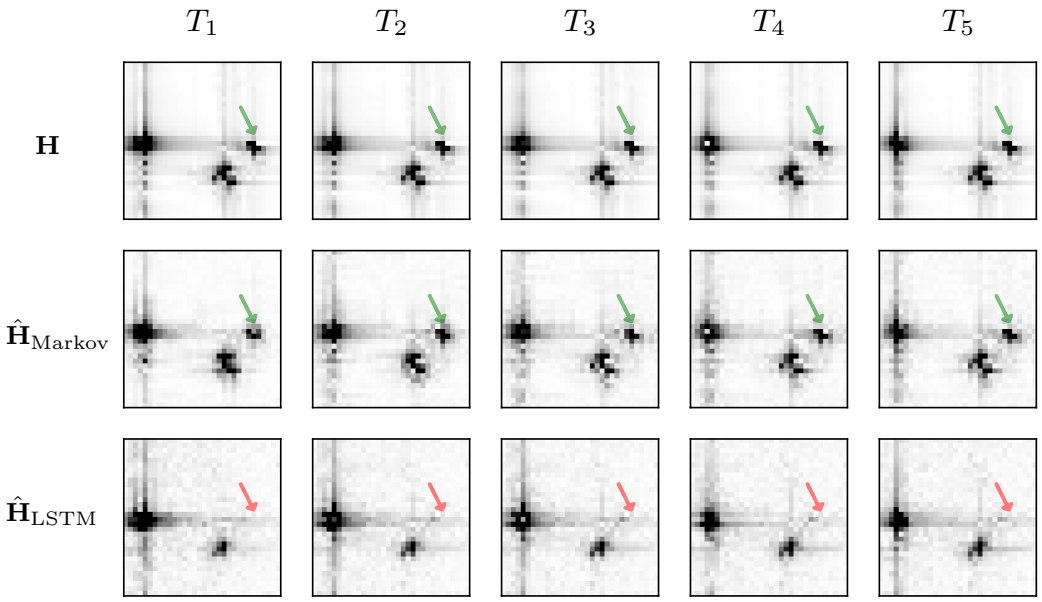


Figure 3.5: CSI ($\mathbf{H}$), MarkovNet estimates ($\hat{\mathbf{H}}_{\text{Markov}}$), and CsiNet-LSTM estimates ($\hat{\mathbf{H}}_{\text{LSTM}}$) across five timeslots (T_1 through T_5) on one outdoor channel sample from the test set, using CR $= \frac{1}{4}$.

Chapter 4

Spectrum-efficient Pilot-based CSI Feedback

This chapter details a scheme for acquiring angular-delay domain CSI at the UE based on a limited number of spatial-frequency domain pilots. This scheme adheres to the 3GPP standard for pilot allocation across time-frequency resources.

4.1 Sparse Pilots in Practical Networks

To estimate the downlink CSI in wireless networks, transmitters allocate pilot reference signals. To reserve spectral resources, pilots are restricted to a limited number of spatial-frequency positions, and the allocation of these pilots is defined in the 3GPP technical standards, TS 36.211 for 4G/LTE networks [54] and TS 38.211 for 5G/NR networks [55]. In these two standards, the pilots are called CSI reference signals (CSI-RS) or demodulation reference signals (DM-RS), respectively. Figure 4.1 shows valid placements of CSI-RS/DM-RS in the time-frequency resource grid as defined by TS 36.211 and TS 38.211.

Considering the sparse placement pilot is important for practical deployment of CSI estimation networks. The works discussed thus far have assumed that (truncated) angular-delay domain CSI is readily available at the UE. However, the pilots as defined in 3GPP standards

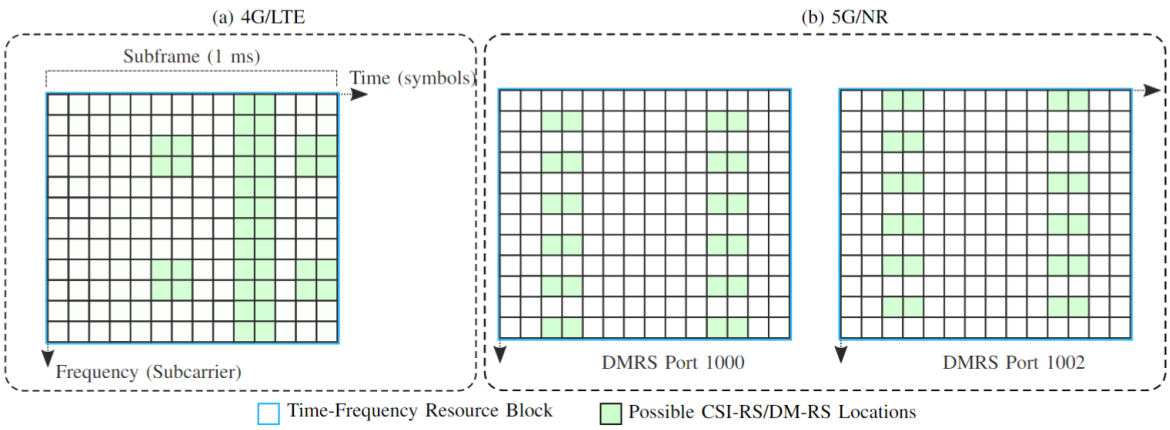


Figure 4.1: (a) LTE resource blocks with CSI-RS locations. (b) 5G NR resource blocks with DM-RS locations.

are defined in the spatial-frequency domain, and to utilize these pilots effectively with CSI estimation networks, one must specify a method for estimating the delay-domain CSI based on the sparse pilots.

The following section details the pilots-to-delay estimator (P2DE), and Figure 4.2 demonstrates the operating principle behind P2DE.

4.2 Pilots-to-delay Estimator (P2DE)

Denote $\boldsymbol{\eta}_i \in \mathbb{C}^{N_f}$ as the i-th row of the spatial-frequency matrix $\mathbf{H}$, and denote the downsampled version of $\boldsymbol{\eta}_i$ as $\boldsymbol{\eta}_{d,i} \in \mathbb{C}^{M_f}$ where $M_f << N_f$. Thus, the spatial-frequency CSI, $\mathbf{H}$, and its downsampled counterpart, $\mathbf{H}_d$, can be written as,

$$\mathbf{H} = \begin{bmatrix} \boldsymbol{\eta}_1 \\ \boldsymbol{\eta}_2 \\ \vdots \\ \boldsymbol{\eta}_{N_b} \end{bmatrix} \in \mathbb{C}^{N_b \times N_f}, \ \mathbf{H}_d = \begin{bmatrix} \boldsymbol{\eta}_{d,1} \\ \boldsymbol{\eta}_{d,2} \\ \vdots \\ \boldsymbol{\eta}_{d,N_b} \end{bmatrix} \in \mathbb{C}^{N_b \times M_f}. \tag{4.1}$$

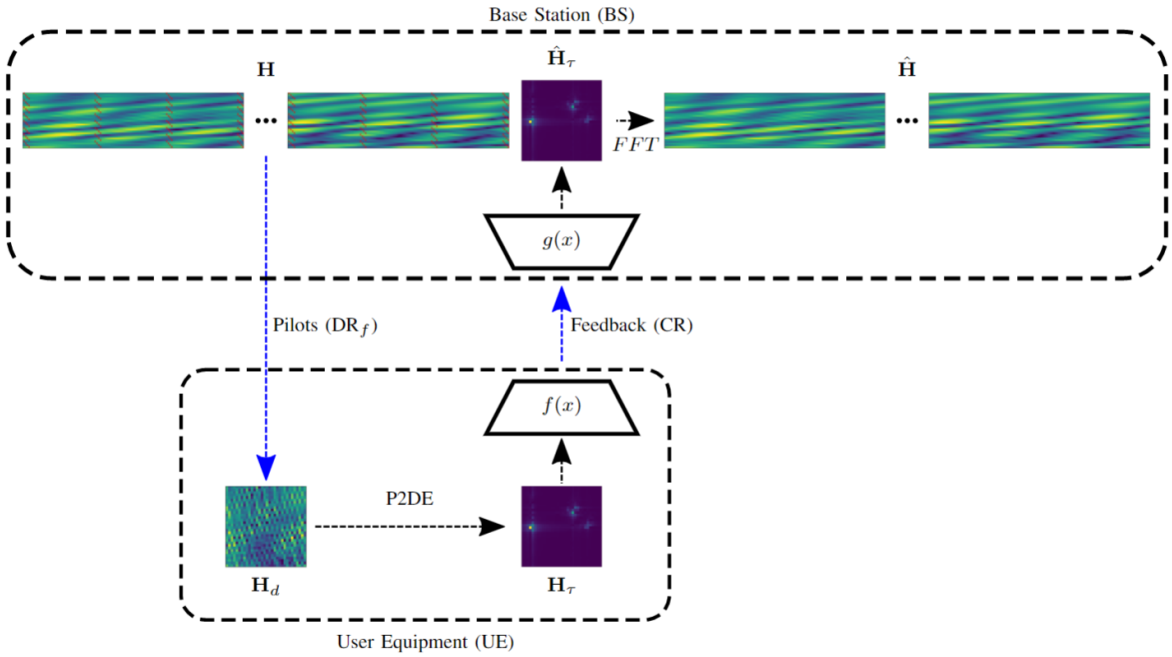


Figure 4.2: Compressive CSI estimation based on linear P2D estimator. First, we use downlink pilots to generate a sparse, frequency domain CSI estimate of size $M_f << N_f$. We then apply the P2D estimator, $\mathbf{Q}_{N_t}^{\dagger}$ of (4.5), to establish the truncated delay domain CSI estimate. We train a learnable encoder, $f(x)$, and decoder, $g(x)$, to compress and decode the feedback, respectively. The gNB recovers the frequency domain CSI from the decoded delay domain CSI estimate.

$\boldsymbol{\eta}_{d,i}$ is related to $\boldsymbol{\eta}_i$ by the downsampling matrix for the i-th antenna port, $\mathbf{P}_i$, as

$$\boldsymbol{\eta}_{d,i} = \boldsymbol{\eta}_i \mathbf{P}_i \; \forall \; i \in [1, \ldots, N_b]. \tag{4.2}$$

Denote the delay-domain CSI vector, $\tilde{\boldsymbol{\eta}}_i$, which is defined as

$$\tilde{\boldsymbol{\eta}}_i \mathbf{F} = \boldsymbol{\eta}_i, \tag{4.3}$$

where $\mathbf{F}$ is the $\mathbf{C}^{N_f \times N_f}$ discrete Fourier transform (DFT) matrix. To relate the frequency domain pilots to the delay domain, we apply the pilot downsampling matrix $\mathbf{P}_i$ to both sides of 4.3,

$$\begin{aligned} \tilde{\boldsymbol{\eta}}_i \mathbf{F} \mathbf{P}_i &= \boldsymbol{\eta}_i \mathbf{P}_i \\ \tilde{\boldsymbol{\eta}}_i \mathbf{Q}_i &= \boldsymbol{\eta}_{d,i} \end{aligned} \tag{4.4}$$

where $\mathbf{Q}_i = \mathbf{F}\mathbf{P}_i \in \mathbb{C}^{N_f \times M_f}$ is the downsampled DFT matrix. Leveraging the sparsity of CSI data in the delay domain (see Section 2.1.1, Figure 2.1), many works choose to feedback and compress the truncated delay domain vectors, $\tilde{\boldsymbol{\eta}}_{c,i} \in \mathbb{C}^{N_t}$. The zero-padded vector $\tilde{\boldsymbol{\eta}}_i$ defined as

$$\tilde{\boldsymbol{\eta}}_i = \left[\tilde{\boldsymbol{\eta}}_{c,i}, \mathbf{0}_{N_f - N_t} \right]. \tag{4.5}$$

Based on 4.4, the delay domain can be related directly to the pilots by taking the pseudoinverse,

$$\begin{aligned} \tilde{\boldsymbol{\eta}}_i \mathbf{Q}_i \mathbf{Q}_i^T &= \boldsymbol{\eta}_{d,i} \mathbf{Q}_i^T \\ \tilde{\boldsymbol{\eta}}_i &= \boldsymbol{\eta}_{d,i} \mathbf{Q}_i^T \left(\mathbf{Q}_i \mathbf{Q}_i^T \right)^{-1} \\ &= \boldsymbol{\eta}_{d,i} \mathbf{Q}_i^{\#} \end{aligned} \tag{4.6}$$

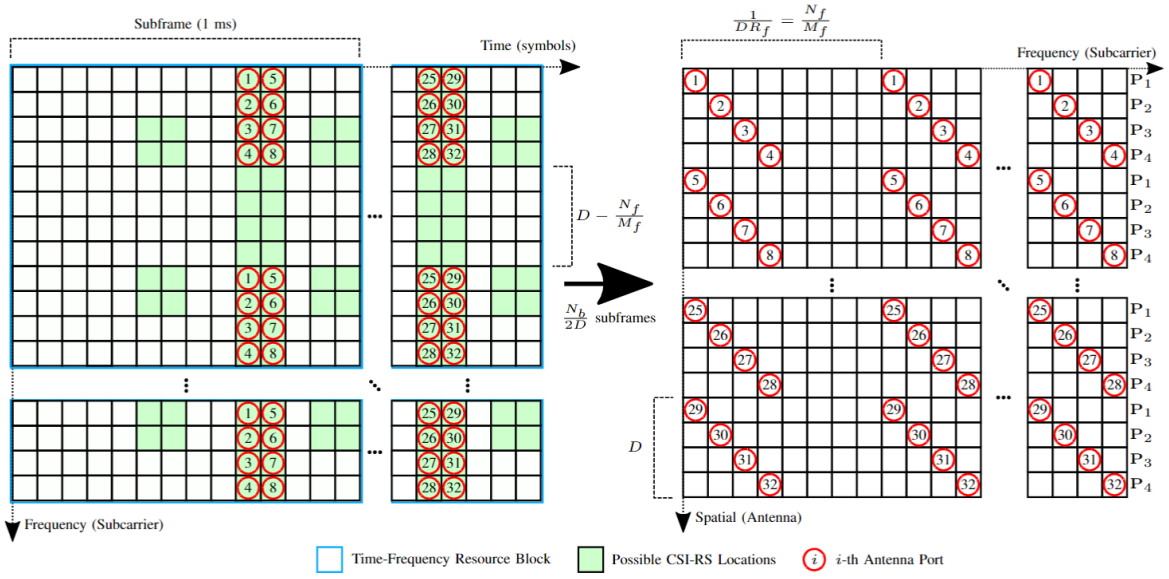


Figure 4.3: (a) LTE Resource Blocks and CSI-RS locations where antenna port pilots are allocated. (b) Schematic for diagonal pilots with relevant parameters, size of diagonal D and frequency downsampling ratio DR_f. In this diagram, $N_b = 32, D = 4, \text{DR}_f = \frac{1}{8}$. The pilot matrix $\mathbf{P}_j$ indicates the downsampling pattern for the j-th element of the diagonal pattern. The number of subframes necessary to populate (b) is inversely proportional to D.

4.2.1 Regularization of P2DE

When the pilot patterns $\mathbf{P}_i$ are equidistant and regularly spaced, the P2DE matrices $\mathbf{Q}_i\mathbf{Q}_i^T$ are typically well-conditioned. However, more irregular patterns can result in ill-conditioned matrices $\mathbf{Q}_i\mathbf{Q}_i^T$, making these matrix inversion unstable.

To compensate for this ill-conditioning, we propose to use off-diagonal regularization (ODIR) to condition the P2DE matrices. Denote $\mathbf{R} = \mathbf{Q}_i\mathbf{Q}_i^T$ and R_{ij} as the element in the i-th row and j-th column of the matrix. We select a non-negative real scaling factor $\delta \in \mathbb{R}^+$ to scale down the off-diagonal elements of $\mathbf{R}$. The element of the resulting ODIR matrix, $\mathbf{R}_{\text{ODIR}}$, are written as

$$R_{ij,\text{ODIR}} = \begin{cases} R_{ij} & \text{if } i = j \\ \frac{R_{ij}}{1+\delta} & \text{if } i \neq j. \end{cases} \tag{4.7}$$

4.3 Results

4.3.1 Accuracy of P2DE

We assess the accuracy of the P2DE under values of M_f and D. Fig. 4.4 demonstrates the accuracy of the P2DE at the UE (i.e., before compression and feedback) for different frequency downsampling ratios. The P2DE achieves impressive accuracy even under aggressive values of DR_f (e.g., better than -30dB at $\text{DR}_f = \frac{1}{8}$). Additionally, the effect of increasing the diagonal size, D, is apparent at larger compression ratios (i.e., CR $\geq \frac{1}{8}$), where the accuracy of the P2DE to a value as low as -25dB. For smaller compression ratios (i.e., CR $\leq \frac{1}{16}$), increasing D has a marginal effect on the accuracy of the P2DE.

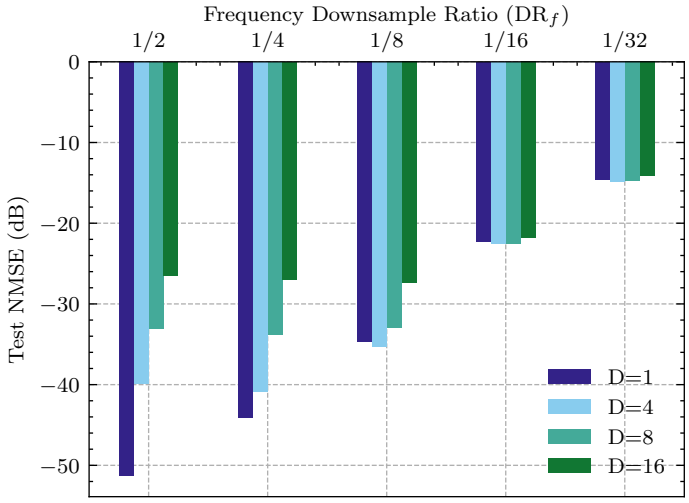


Figure 4.4: P2D estimation performance under different frequency downsampling ratios ($\text{DR}_f = \frac{M_f}{N_f}$) and diagonal dimensions (D) for the Outdoor COST2100 dataset. Downsampling is done along the frequency axis.

Having assessed the initial accuracy of the P2DE at the UE, we now apply a deep learning network to compress the output of the P2DE. Figure 4.5 demonstrates the accuracy of ISTANet+ [56] for multiple compression ratios (CR) using the P2DE as its input. For progressively smaller compression ratios, the accuracy of ISTANet+ remains stable until $\text{DR}_f = \frac{1}{32}$, at which point the network's performance degrades.

Finally, we assess the accuracy of the P2DE assuming noise from pilot estimation. To

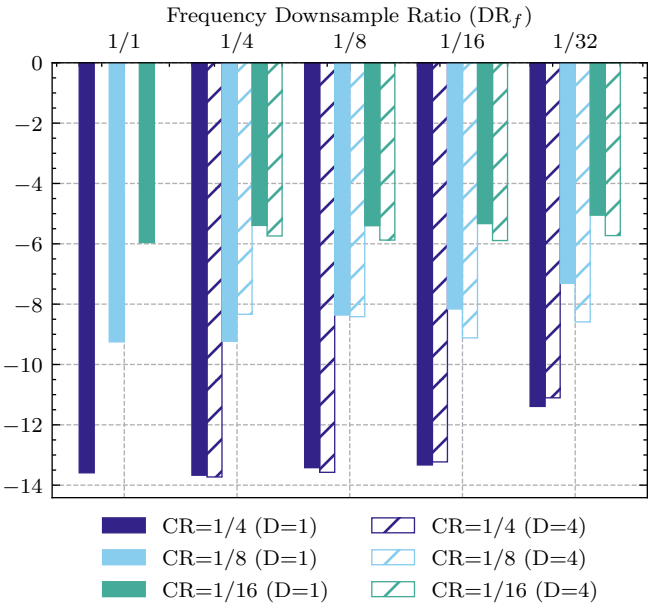


Figure 4.5: Performance of ISTANet+ for multiple compression ratios using P2D estimates with different downsampling ratios ($\text{DR}_f = \frac{M_f}{N_f}$) for the Outdoor COST2100 dataset. Non-diagonal pattern ($D = 1$) is compared with a diagonal pattern of size $D = 4$. Performance for $\text{DR}_f = 1/1$, $D = 4$ is omitted since it is equivalent to the $\text{DR}_f = 1, D = 1$ case.

simulate pilot estimation error, we use additive Gaussian noise,

$$\hat{\mathbf{H}}_d = \mathbf{H}_d + \mathbf{N}_d$$

where the elements of $\mathbf{N}_d$, $\mathbf{N}_d(i,j) \sim \mathcal{N}(0, \sigma^2)$ for $i \in [1, 2, \ldots, N_b], j \in [1, 2, \ldots, M_f]$. To achieve different SNR values for $\hat{\mathbf{H}}_d$, we simply vary the noise variance σ^2, and we use the P2DE at different pilot estimation noise levels. Figure 4.6 shows the accuracy of the P2DE for different values of σ^2.

In addition to varying the pilots estimation SNR, we also showcase the effect of varying δ (i.e., the ODIR parameter as described in Section 4.2.1). We observe that δ helps the P2DE achieve better performance under both low-noise and noisy conditions, i.e.,

- **Low-noise condition (SNR $= -20$ dB)**: The P2DE goes from -8 dB to -22 dB for $\delta = 0$ and $\delta = 0.5$, respectively.

- **Noisy condition (SNR ≥ -10 dB)**: The P2DE goes from -9 dB to -30 dB for $\delta = 0$ and $\delta = 0.5$, respectively.

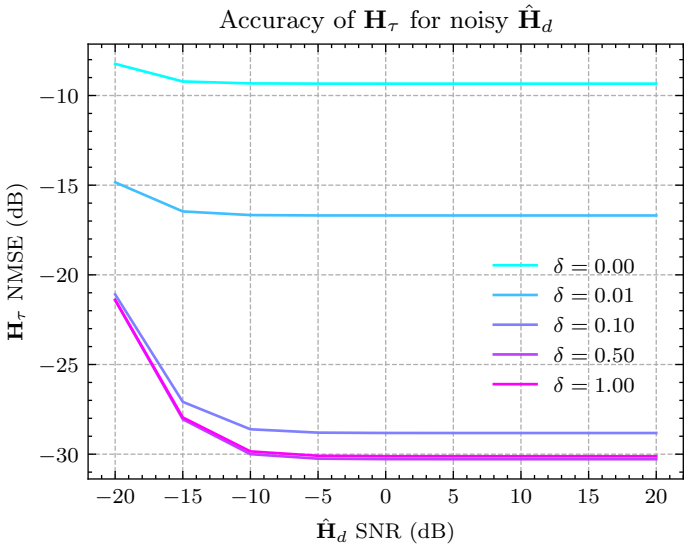


Figure 4.6: Accuracy of P2DE output, $\mathbf{H}_\tau$, assuming noisy pilots, $\hat{\mathbf{H}}_d$. Additive Gaussian noise is used to model the error inherent in pilot estimation. Here, $D = 4, \text{DR}_f = \frac{1}{32}$.

4.3.2 P2DE Compression Network Comparison

References

[1] A. Goldsmith, S. A. Jafar, N. Jindal, and S. Vishwanath, "Capacity limits of mimo channels," *IEEE Journal on selected areas in Communications*, vol. 21, no. 5, pp. 684–702, 2003.

[2] H. Y. H. Q. N. Thomas L. Marzetta, Erik G. Larsson, *Fundamentals of Massive MIMO*. Cambridge University Press.

[3] E. Crespo Marques, N. Maciel, L. Naviner, H. Cai, and J. Yang, "A Review of Sparse Recovery Algorithms," *IEEE Access*, vol. 7, pp. 1300–1322, 2019.

[4] C. Wen, W. Shih, and S. Jin, "Deep Learning for Massive MIMO CSI Feedback," *IEEE Wireless Communications Letters*, vol. 7, no. 5, pp. 748–751, Oct 2018.

[5] J. Guo, L. Wang, F. Li, and J. Xue, "CSI Feedback With Model-Driven Deep Learning of Massive MIMO Systems," *IEEE Communications Letters*, vol. 26, pp. 547–551, 2022, fISTANet.

[6] H. Yang and T. L. Marzetta, "Performance of conjugate and zero-forcing beamforming in large-scale antenna systems," *IEEE Journal on Selected Areas in Communications*, vol. 31, no. 2, pp. 172–179, 2013.

[7] F. Kaltenberger, H. Jiang, M. Guillaud, and R. Knopp, "Relative Channel Reciprocity Calibration in MIMO/TDD Systems," in *2010 Future Network Mobile Summit*, June 2010, pp. 1–10.

[8] D. Mi, M. Dianati, L. Zhang, S. Muhaidat, and R. Tafazolli, "Massive mimo performance with imperfect channel reciprocity and channel estimation error," *IEEE Transactions on Communications*, vol. 65, no. 9, pp. 3734–3749, 2017.

[9] Q. Gao, F. Qin, and S. Sun, "Utilization of channel reciprocity in advanced mimo system," in *2010 5th International ICST Conference on Communications and Networking in China*, Aug 2010, pp. 1–5.

[10] L. Liu, C. Oestges, J. Poutanen, K. Haneda, P. Vainikainen, F. Quitin, F. Tufvesson, and P. D. Doncker, "The COST 2100 MIMO Channel Model," *IEEE Wireless Communications*, vol. 19, no. 6, pp. 92–99, December 2012.

[11] B. Makki and T. Eriksson, "On hybrid arq and quantized csi feedback schemes in quasi-static fading channels," *IEEE transactions on communications*, vol. 60, no. 4, pp. 986–997, 2012.

[12] H. Shirani-Mehr and G. Caire, "Channel state feedback schemes for multiuser mimo-ofdm downlink," *IEEE Transactions on Communications*, vol. 57, no. 9, pp. 2713–2723, 2009.

[13] X. Rao and V. K. Lau, "Distributed compressive csit estimation and feedback for fdd multi-user massive mimo systems," *IEEE Transactions on Signal Processing*, vol. 62, no. 12, pp. 3261–3271, 2014.

[14] M. E. Eltayeb, T. Y. Al-Naffouri, and H. R. Bahrami, "Compressive sensing for feedback reduction in mimo broadcast channels," *IEEE Transactions on communications*, vol. 62, no. 9, pp. 3209–3222, 2014.

[15] T. T. Do, L. Gan, N. Nguyen, and T. D. Tran, "Sparsity adaptive matching pursuit algorithm for practical compressed sensing," in *2008 42nd Asilomar conference on signals, systems and computers*. IEEE, 2008, pp. 581–587.

[16] T. Hastie, R. Tibshiran, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2016. [Online]. Available: https://web.stanford.edu/~hastie/ElemStatLearn/

[17] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[18] R. Hecht-Nielsen, "Theory of the backpropagation neural network," in *Neural networks for perception*. Elsevier, 1992, pp. 65–93.

[19] S. Sabour, N. Frosst, and G. E. Hinton, "Dynamic routing between capsules," *arXiv preprint arXiv:1710.09829*, 2017.

[20] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969.

[21] Z. Lu, J. Wang, and J. Song, "Multi-resolution csi feedback with deep learning in massive mimo system," in *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, 2020, pp. 1–6.

[22] M. Hussien, K. K. Nguyen, and M. Cheriet, "PRVNet: Variational autoencoders for massive MIMO CSI feedback," *arXiv*, 2020.

[23] Y. Sun, W. Xu, L. Fan, G. Y. Li, and G. K. Karagiannidis, "Ancinet: An efficient deep learning approach for feedback compression of estimated csi in massive mimo systems," *IEEE Wireless Communications Letters*, vol. 9, no. 12, pp. 2192–2196, 2020.

[24] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, "Inception-v4, Inception-Resnet and the Impact of Residual Connections on Learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, no. 1, 2017.

[25] J. Ballé, V. Laparra, and E. P. Simoncelli, "End-to-end Optimized Image Compression," in *5th International Conference on Learning Representations, ICLR 2017*, 2017.

[26] J. Xie, L. Xu, and E. Chen, "Image Denoising and Inpainting with Deep Neural Networks," *Advances in Neural Information Processing Systems*, vol. 25, pp. 341–349, 2012.

[27] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, "Language Models are Few-shot Learners," *arXiv preprint arXiv:2005.14165*, 2020.

[28] A. M. Sayeed, "Deconstructing multiantenna fading channels," *IEEE Transactions on Signal processing*, vol. 50, no. 10, pp. 2563–2579, 2002.

[29] Z. Liu, L. Zhang, and Z. Ding, "Exploiting Bi-Directional Channel Reciprocity in Deep Learning for Low Rate Massive MIMO CSI Feedback," *IEEE Wireless Communications Letters*, vol. 8, no. 3, pp. 889–892, 2019.

[30] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*. PMLR, 2015, pp. 448–456.

[31] X. Huang and S. Belongie, "Arbitrary style transfer in real-time with adaptive instance normalization," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 1501–1510.

[32] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.

[33] Y. Wu and K. He, "Group normalization," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 3–19.

[34] E. Ogasawara, L. C. Martinez, D. De Oliveira, G. Zimbrão, G. L. Pappa, and M. Mattoso, "Adaptive normalization: A novel data normalization approach for non-stationary

time series," in *The 2010 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2010, pp. 1–8.

[35] S. Nayak, B. B. Misra, and H. S. Behera, "Impact of data normalization on stock index forecasting," *International Journal of Computer Information Systems and Industrial Management Applications*, vol. 6, no. 2014, pp. 257–269, 2014.

[36] X. Shao, "Self-normalization for time series: a review of recent developments," *Journal of the American Statistical Association*, vol. 110, no. 512, pp. 1797–1817, 2015.

[37] N. Passalis, A. Tefas, J. Kanniainen, M. Gabbouj, and A. Iosifidis, "Deep adaptive input normalization for time series forecasting," *IEEE transactions on neural networks and learning systems*, vol. 31, no. 9, pp. 3760–3765, 2019.

[38] Z. Liu, **M. del Rosario**, X. Liang, L. Zhang, and Z. Ding, "Spherical Normalization for Learned Compressive Feedback in Massive MIMO CSI Acquisition," in *2020 IEEE International Conference on Communications Workshops (ICC Workshops)*, 2020, pp. 1–6.

[39] R. Chopra, C. R. Murthy, and H. A. Suraweera, "On the Throughput of Large MIMO Beamforming Systems With Channel Aging," *IEEE Signal Processing Letters*, vol. 23, no. 11, pp. 1523–1527, 2016.

[40] K. Huber and S. Haykin, "Improved bayesian mimo channel tracking for wireless communications: incorporating a dynamical model," *IEEE transactions on wireless communications*, vol. 5, no. 9, pp. 2458–2466, 2006.

[41] K. S. Ali and P. Sampath, "Time domain channel estimation for time and frequency selective millimeter wave mimo hybrid architectures: Sparse bayesian learning-based kalman filter," *Wireless Personal Communications*, pp. 1–21, 2020.

[42] H. Kim, S. Kim, H. Lee, C. Jang, Y. Choi, and J. Choi, "Massive mimo channel prediction: Kalman filtering vs. machine learning," *IEEE Transactions on Communications*, vol. 69, no. 1, pp. 518–528, 2021.

[43] C. Lu, W. Xu, H. Shen, J. Zhu, and K. Wang, "MIMO Channel Information Feedback Using Deep Recurrent Network," *IEEE Commu. Letters*, vol. 23, no. 1, pp. 188–191, Jan 2019.

[44] Y. Liao, H. Yao, Y. Hua, and C. Li, "CSI Feedback Based on Deep Learning for Massive MIMO Systems," *IEEE Access*, vol. 7, pp. 86 810–86 820, 2019.

[45] X. Li and H. Wu, "Spatio-Temporal Representation With Deep Neural Recurrent Network in MIMO CSI Feedback," *IEEE Wireless Comm. Letters*, vol. 9, no. 5, pp. 653–657, 2020.

[46] Y. Jang, G. Kong, M. Jung, S. Choi, and I. Kim, "Deep Autoencoder Based CSI Feedback With Feedback Errors and Feedback Delay in FDD Massive MIMO Systems," *IEEE Wireless Comm. Letters*, vol. 8, no. 3, pp. 833–836, 2019.

[47] T. Wang, C. Wen, S. Jin, and G. Y. Li, "Deep Learning-Based CSI Feedback Approach for Time-Varying Massive MIMO Channels," *IEEE Wireless Comm. Letters*, vol. 8, no. 2, pp. 416–419, April 2019.

[48] M. Hermans and B. Schrauwen, "Training and analysing deep recurrent neural networks," in *Advances in Neural Information Processing Systems 26*, C. J. C. Burges and et al., Eds., 2013, pp. 190–198.

[49] R. Pascanu, C. Gulcehre, K. Cho, and Y. Bengio, "How to construct deep recurrent neural networks," *2nd International Conference on Learning Representations*, no. March 2014, 2014.

[50] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," *Advances in Neural Information Processing Systems*, vol. 4, no. January, pp. 3104–3112, 2014.

[51] O. Irsoy and C. Cardie, "Opinion mining with deep recurrent neural networks," *Proc. 2014 Conference on Empirical Methods in Natural Language Processing*, pp. 720–728, 2014.

[52] Y. Bengio, "Learning deep architectures for AI," *Foundations and Trends in Machine Learning*, vol. 2, no. 1, pp. 1–27, 2009.

[53] Z. Liu †, **M. del Rosario** †, and Z. Ding, "A markovian model-driven deep learning framework for massive mimo csi feedback," *IEEE Transactions on Wireless Communications*, vol. 21, no. 2, pp. 1214–1228, 2022.

[54] 3GPP, "Evolved Universal Terrestrial Radio Access (E-UTRA); Physical channels and modulation," 3rd Generation Partnership Project (3GPP), TS 36.211, Jan. 2022. [Online]. Available: http://www.3gpp.org/dynareport/36211.htm

[55] ——, "NR; Physical Channels and Modulation," 3rd Generation Partnership Project (3GPP), TS 38.211, Jan. 2022. [Online]. Available: http://www.3gpp.org/dynareport/38211.htm

[56] J. Zhang and B. Ghanem, "ISTA-Net: Interpretable Optimization-inspired Deep Network for Image Compressive Sensing," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 1828–1837.