Over-the-Air Design of GAN Training for mmWave MIMO Channel Estimation

Akash Doshi, Manan Gupta and Jeffrey G. Andrews

Abstract

Future wireless systems are trending towards higher carrier frequencies that offer larger communication bandwidth but necessitate the use of large antenna arrays. Existing signal processing techniques for channel estimation do not scale well to this "high-dimensional" regime in terms of performance and pilot overhead. Meanwhile, training deep learning based approaches for channel estimation requires large labeled datasets mapping pilot measurements to clean channel realizations, which can only be generated offline using simulated channels. In this paper, we develop a novel unsupervised over-the-air (OTA) algorithm that utilizes noisy received pilot measurements to train a deep generative model to output beamspace MIMO channel realizations. Our approach leverages Generative Adversarial Networks (GAN), while using a conditional input to distinguish between Line-of-Sight (LOS) and Non-Line-of-Sight (NLOS) channel realizations. We also present a federated implementation of the OTA algorithm that distributes the GAN training over multiple users and greatly reduces the user side computation. We then formulate channel estimation from a limited number of pilot measurements as an inverse problem and reconstruct the channel by optimizing the input vector of the trained generative model. Our proposed approach significantly outperforms Orthogonal Matching Pursuit on both LOS and NLOS channel models, and EM-GM-AMP – an Approximate Message Passing algorithm – on LOS channel models, while achieving comparable performance on NLOS channel models in terms of the normalized channel reconstruction error. More importantly, our proposed framework has the potential to be trained online using real noisy pilot measurements, is not restricted to a specific channel model and can even be utilized for a federated OTA design of a dataset generator from noisy data.

Index Terms

Channel estimation, compressed sensing, deep generative models, Generative Adversarial Networks (GAN), mmWave MIMO, over-the-air (OTA)

I. INTRODUCTION

A. Motivation

Channel estimation (CE) in 6G and beyond will increasingly be performed for larger antenna arrays at both the base station (BS) and user (UE), increasing the dimensionality and complexity of the problem, as future communication systems tend towards progressively higher carrier frequencies in the mmWave and sub-THz range [1], [2]. While channel estimation at sub-6 GHz in practical wireless deployments is usually performed using Least Squares (LS) or minimum mean squared error (MMSE) estimators [3], these do not scale to higher carrier frequencies which require large antenna arrays.

At higher carrier frequencies, the beamspace sparsity of MIMO channels is exploited to perform beam alignment – steering the antenna arrays at the BS and UE to maximize SNR. Due to latency constraints, the number of candidate beams used to perform beam alignment is relatively small, limiting the SNR achieved. Consequently, researchers have been actively investigating learning-based approaches with an eye to improving both CE and beam alignment in this "high-dimensional" regime. A common drawback of such attempts has been the requirement of large datasets of clean channel realizations for training. Such training datasets are either generated using statistical channel models or through ray-tracing for a given physical environment. What is desirable is a learning-based CE or beam alignment algorithm that can be trained using received (noisy) pilot measurements alone. This will eliminate performance losses stemming from inaccurate channel modelling and allow for online updates to the learned models.

Deep generative modelling [4] is a powerful unsupervised learning tool that can be used to implicitly learn complex probability distributions. Recently, it has also been utilized for solving *inverse problems* i.e. reconstructing an unknown signal or image from observations [5]. The forward process mapping the signal to the observation is typically non-invertible, hence it is impossible to uniquely reconstruct the signal in the absence of some prior knowledge [6]. A trained deep generative network can be used to encode such a prior. In this paper, we describe a novel over-the-air approach based on deep generative modelling to learn the high-dimensional channel distribution from noisy pilot measurements and subsequently use the trained generative networks to accurately reconstruct the beamspace representation of the channel from compressive pilot measurements.

B. Related Work

Channel measurements indicate that mmWave MIMO channels are sparse in their beamspace representation [7] due to clustering of the paths into small, relatively narrowbeam clusters. To exploit this sparsity (that LS and MMSE cannot), several papers have provided a sparse formulation of mmWave CE in both the angular [8] and delay-Doppler domain [9] and subsequently employed Orthogonal Matching Pursuit (OMP) [10] and Basis Pursuit Denoising [11] for sparse channel reconstruction. Approximate Message Passing (AMP) [12] and its variants such as EM-GM-AMP [13] and VAMP [14] have also proved to be robust compressed sensing (CS)-based approaches to sparse channel estimation [15], [16].

CS-based approaches suffer from the drawback of assuming exact sparsity in the DFT basis, and result in poor channel reconstruction if the cross-correlation in the columns of the sensing matrix is high [17]. Meanwhile deep learning based approaches to channel estimation have been making rapid advancements. A plethora of papers recently [18]–[22] have designed supervised deep learning techniques that provide pilot measurements as input to a neural network (NN) that outputs the desired channel. These suffer from the drawback of requiring a labeled dataset to train, and hence have to be trained offline. To circumvent the requirement of a labeled dataset, we designed an unsupervised CE algorithm [23], [24] that trained a deep generative network to output channel realizations using Generative Adversarial Networks (GAN) [25]. Our proposed algorithm used the trained generative network for high-dimensional CE from a small number of pilots and provided a compressed representation of the estimated channel which could be used for channel reconstruction at the BS. By employing posterior sampling via Langevin dynamics in combination with a generative prior [26] for channel estimation, [27] has further improved upon the performance of [23], albeit at the cost of a significantly increased execution time which could run contrary to the block fading assumption made in their exposition. The generative CE algorithm of [23] was also applied to wideband channel estimation in [28], at the cost of designing a separate generative model for each transmit and receive antenna pair, thus not exploiting spatial correlation.

However, the aforementioned generative channel estimation techniques and other works performing generative channel modelling [29] require a large dataset of clean channel realizations to train a GAN. In addition, our prior work [23] has the following two limitations: i) Antenna spacing of $\lambda_c/10$ (where $\lambda_c = c/f_c$ and f_c is the carrier frequency) in place of the conventional

$\lambda_c/2$ was assumed to generate high spatial correlation in channel realizations and ii) the GAN was trained and its performance evaluated only on channels with a strong LOS component.

C. Contributions

We propose a design for an over-the-air (OTA) implementation of GAN training and generative channel estimation. We utilize received pilot measurements, instead of clean channel realizations, to train a Conditional Wasserstein GAN with no assumptions on the channel model or line-of-sight conditions. We propose several new ideas that while collectively overcoming the key real-world limitations of the initial work done in [23], also set forth a new distributed design strategy – FedPilotGAN – as elaborated in our contributions below.

Beamspace Generative Channel Estimation: In order to relax the antenna spacing constraint from $\lambda_c/10$ to the conventional $\lambda_c/2$, we propose to solve generative channel estimation in the beamspace domain. This helps to exploit the angular clustering in mmWave MIMO channels, while simultaneously not imposing any constraints on the sparsity of the beamspace representation as is the case in traditional CS algorithms. Moreover, we demonstrate the advantage of using deep generative models over CS based approaches for solving inverse problems by showing that a trained generative prior eliminates the need for careful tuning of the sensing matrices.

Training Wasserstein GANs for LOS and NLOS channels: In [23], we trained a Wasserstein GAN (WGAN) on channels with a dominating Line-of-Sight (LOS) component. By training the generator to output beamspace channel representations and improving the performance of WGAN using Gradient Penalty (GP) [30], we train generative models for a range of LOS and Non-Line-of-Sight (NLOS) MIMO channels with varying degrees of approximate sparsity in their beamspace representation. Subsequently, we train a single Conditional Wasserstein GAN (CWGAN) that takes as conditional input the channel's LOS/NLOS state to output samples from the entire aforementioned range of channels.

Training a GAN from noisy pilot measurements: In order to overcome the requirement of a dataset of clean channel realizations, we present a novel GAN architecture – *Pilot GAN* – that combines LS channel estimation with Ambient GAN [31] to train a generative model to output MIMO channel realizations given a dataset of full-rank noisy pilot measurements in a hybrid mmWave architecture. We also provide insights into the range of measurement models under which the Pilot GAN framework is applicable.

Federated Pilot GAN: We describe a federated implementation of Pilot GAN – *FedPilotGAN* – to facilitate distributed GAN training from noisy pilot measurements over multiple users. This helps to reduce the UE side computation and move the computationally intensive generator training to the BS without impacting the performance of generative channel estimation, as will be demonstrated in Section VI-C. We also highlight the ability of FedPilotGAN to design a dataset generator at a server (BS) using only noisy realizations collected at multiple users – without transmitting data realizations between the server and the users.

Pilot Conditional GAN: We combine the aforementioned Pilot GAN with the CWGAN architecture to train a conditional generative model from noisy pilot measurements. Furthermore, we train a neural network (NN) to determine the LOS/NLOS state of the channel based on pilot measurements, and use its output as input to the conditional generator, thus designing an OTA GAN training algorithm based on pilot measurements. The resulting trained conditional generator is then used to perform CE from compressive pilot measurements across a variety of high-dimensional MIMO channels.

D. Notation & Organization

We use bold uppercase $\mathbf{A}$ to denote a matrix and bold lowercase $\mathbf{a}$ to denote a vector. $||\mathbf{A}||_2$ is the Frobenius norm of $\mathbf{A}$ and $||\mathbf{a}||_0$ denotes the number of non-zero entries in $\mathbf{a}$. $\text{diag}(\mathbf{A}_1, \ldots, \mathbf{A}_N)$ represents a block diagonal matrix whose diagonal entries are given by $\{\mathbf{A}_1, \ldots, \mathbf{A}_N\}$. $\underline{\mathbf{A}}$ is a vector obtained by stacking the columns of $\mathbf{A}$. For matrix $\mathbf{A}$ of size $M \times N$ and matrix $\mathbf{B}$ of size $P \times Q$, $\mathbf{A} \otimes \mathbf{B}$ denotes the $MP \times NQ$ matrix of their Kronecker product. $\mathbb{E}[.]$ is the expectation operator, $\text{sgn}(x) = 1$ if $x > 0$, -1 if $x < 0$ and 0 otherwise and $\mathbb{P}(B)$ denotes the probability that an event B occurs. $[a]$ denotes the set $\{1, \ldots, a\}$ for any integer a, while $[a] \times [b]$ denotes the Cartesian product of the two sets.

The paper is organized as follows. The system model is outlined in Section II, followed by a detailed description of the beamspace generative channel estimator in Section III. Each of the GAN architectures mentioned above is then expounded upon in Section IV, followed by the architectural and simulation details of these GANs in Section V. The results are presented in Section VI and the paper concludes with final thoughts and future directions in Section VII.

II. SYSTEM MODEL

Consider the problem of single user downlink (DL) narrowband mmWave MIMO channel estimation with a transmitter having N_t antennas and the receiver having N_r antennas. Denote the hybrid precoder at the transmitter by $\mathbf{F} \in \mathbb{C}^{N_t \times N_s}$, and the hybrid combiner at the receiver by $\mathbf{W} \in \mathbb{C}^{N_r \times N_s}$, with N_s being the number of data streams that can be transmitted. With the MIMO channel denoted by $\mathbf{H} \in \mathbb{C}^{N_r \times N_t}$, the transmitted pilot symbols $\mathbf{S} \in \mathbb{C}^{N_s \times N_p}$ are received as

$$\mathbf{Y} = \mathbf{W}^H \mathbf{H} \mathbf{F} \mathbf{S} + \mathbf{W}^H \mathbf{N}, \tag{1}$$

where each element of $\mathbf{N} \in \mathbf{C}^{N_r \times N_p}$ is an independent and identically distributed (i.i.d.) complex Gaussian random variable with mean 0 and some variance σ^2. Implicitly assumed in (1) is a block fading model over N_p pilot symbols i.e. a new i.i.d. channel realization $\mathbf{H}$ is chosen every N_p time slots. In this paper, we assume a fully connected phase shifting network [11]. We also constrain the angles realized by the phase shifters to quantized sets [32] given by

$$\mathcal{A} = \left\{ 0, \frac{2\pi}{2^{N_Q}}, \ldots, \frac{(2^{N_Q} - 1)2\pi}{2^{N_Q}} \right\}, \tag{2}$$

where N_Q is the number of quantization bits. We assume $N_{bit,t}$ and $N_{bit,r}$ phase shift quantization bits at the transmitter and receiver respectively, with the quantization sets denoted by $\mathcal{A}_t$ and $\mathcal{A}_r$ respectively. This implies $[\mathbf{F}]_{i,j} = \frac{1}{\sqrt{N_t}} e^{j\psi_{i,j}}$ and $[\mathbf{W}]_{i,j} = \frac{1}{\sqrt{N_r}} e^{j\phi_{i,j}}$ where $\psi_{i,j} \in \mathcal{A}_t$ and $\phi_{i,j} \in \mathcal{A}_r$. Vectorizing (1) and utilizing the Kronecker product identity $\underline{\mathbf{A}\mathbf{B}\mathbf{C}} = (\mathbf{C}^T \otimes \mathbf{A})\underline{\mathbf{B}}$, we obtain

$$\underline{\mathbf{y}} = (\mathbf{S}^T \mathbf{F}^T \otimes \mathbf{W}^H)\underline{\mathbf{H}} + (\mathbf{I}_{N_p} \otimes \mathbf{W}^H)\underline{\mathbf{n}}, \tag{3}$$

where $\underline{\mathbf{y}} \in \mathbb{C}^{N_s N_p \times 1}$, $\underline{\mathbf{H}} \in \mathbb{C}^{N_r N_t \times 1}$ and $\underline{\mathbf{n}} \in \mathbb{C}^{N_r N_p \times 1}$. Denote $\mathbf{A} = (\mathbf{S}^T \mathbf{F}^T \otimes \mathbf{W}^H)$. $\mathbf{A}$ has dimensions $N_s N_p \times N_t N_r$, however its rank is bounded by N_s^2, since $\mathbf{F}$ can have atmost N_s independent columns. Consequently, if $N_s N_p < N_t N_r$, the system of equations given by (3) does not have a unique solution, and we cannot perform traditional least-squares (LS) estimation to recover $\underline{\mathbf{H}}$. In other words, recovering $\underline{\mathbf{H}}$ from $\underline{\mathbf{y}}$ given $\mathbf{A}$ is an inverse problem. In Section III, we will describe an algorithm to solve this inverse problem using a deep generative prior.

III. GENERATIVE CHANNEL ESTIMATION (GCE)

Deep generative models $\mathbf{G}$ are feed-forward neural networks (NN) that take as input a low dimensional vector $\mathbf{z} \in \mathbb{R}^d$ and output high dimensional matrices $\mathbf{G}(z) \in \mathbb{R}^{c \times l \times w}$ where $d \ll clw$.

Such a model can be trained to take a i.i.d. Gaussian vector $\mathbf{z}$ as input and produce samples from complicated distributions, such as human faces [4]. One powerful method for training generative models is using Generative Adversarial Networks (GAN) [25].

In [23], we developed an algorithm that utilized compressed sensing using deep generative models [5] to perform MIMO channel estimation. We trained $\mathbf{G}$ to output channel realizations $\mathbf{H}$ from a given distribution, and then utilized $\mathbf{G}$ to recover $\mathbf{H}$ from compressive pilot measurements $\underline{\mathbf{y}}$. However, we experimentally demonstrated that a reduced antenna spacing of $\lambda_c/10$ in the antenna arrays at the transmitter and receiver was key to training $\mathbf{G}$ successfully. We attributed this to the high spatial correlation generated in channel realizations by such an antenna spacing, making it easier for $\mathbf{G}$ to learn the underlying channel distribution.

In this paper, we will utilize the following key insight to output channel realizations with the conventional $\lambda_c/2$ antenna spacing: *beamspace representation of mmWave MIMO channels have high spatial correlation due to clustering in the angular domain.* To be precise, assuming uniformly spaced linear arrays at the transmitter and receiver, the array response matrices are given by the unitary DFT matrices $\mathbf{A}_\mathrm{T} \in \mathbb{C}^{N_\mathrm{t} \times N_\mathrm{t}}$ and $\mathbf{A}_\mathrm{R} \in \mathbb{C}^{N_\mathrm{r} \times N_\mathrm{r}}$ respectively. Then, we can represent $\mathbf{H}$ as

$$\mathbf{H} = \mathbf{A}_\mathrm{R}\mathbf{H}_\mathrm{v}\mathbf{A}_\mathrm{T}^H$$
$$\underline{\mathbf{H}} = ((\mathbf{A}_\mathrm{T}^H)^T \otimes \mathbf{A}_\mathrm{R})\underline{\mathbf{H}}_\mathrm{v}. \tag{4}$$

We will train $\mathbf{G}$ to output samples of $\mathbf{H}_\mathrm{v}$ i.e. $\mathbb{P}_\mathbf{G}$ converges to $\mathbb{P}_{\mathbf{H}_\mathrm{v}}$ as GAN training progresses. Note that unlike compressed sensing (CS) algorithms that require $\mathbf{H}_\mathrm{v}$ to have a small number of non-zero measurements in order to be recoverable [33], we impose no such constraints on $\mathbf{G}(\mathbf{z})$. However, we will experimentally validate in Section VI that outputting $\mathbf{H}_\mathrm{v}$ instead of $\mathbf{H}$ enables $\mathbf{G}$ to directly exploit the clustering in angular domain to successfully learn a generative prior.

Subsequently, given a trained generator $\mathbf{G}$ and pilot measurements $\underline{\mathbf{y}}$ as defined in (3), we will utilize GCE to solve the following optimization problem

$$\mathbf{z}^* = \arg\min_{\mathbf{z} \in \mathbb{R}^d} \; ||\underline{\mathbf{y}} - \mathbf{A}_\mathrm{sp}\underline{\mathbf{G}}(\mathbf{z})||_2^2 + \lambda_\mathrm{reg}||\mathbf{z}||_2^2, \tag{5}$$

where $\mathbf{A}_\mathrm{sp} = (\mathbf{A}_\mathrm{T}^H\mathbf{F}\mathbf{S})^T \otimes \mathbf{W}^H\mathbf{A}_\mathrm{R}$ and λ_reg is a regularization parameter. An illustration of the framework is shown in Fig. 1. The beamspace channel estimate is then given by $\mathbf{H}_\mathrm{v,est} = \mathbf{G}(\mathbf{z}^*)$. The performance metric used to assess the quality of $\mathbf{H}_\mathrm{v,est}$ is the normalized mean square error
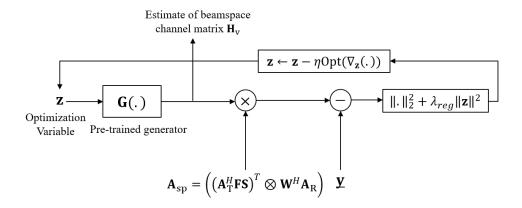
Fig. 1. Generative Channel Estimation Framework. η denotes the step size and Opt denotes any first order optimizer.

(NMSE), defined as

$$\mathrm{NMSE} = \mathbb{E}\left[\frac{||\mathbf{H}_\mathrm{v} - \mathbf{H}_\mathrm{v,est}||_2^2}{||\mathbf{H}_\mathrm{v}||_2^2}\right], \tag{6}$$

where the expected value is over the underlying probability distribution of the channel. It should be noted that since $\mathbf{A}_\mathrm{R}$ and $\mathbf{A}_\mathrm{T}$ are unitary matrices, the NMSE is the same in the beamspace ($\mathbf{H}_\mathrm{v,est}$) and spatial ($\mathbf{H}_\mathrm{est}$) domain. The NMSE, as defined in (6), will serve as a measure of the quality of the generator $\mathbf{G}$ trained by a GAN throughout the paper.

IV. GAN ARCHITECTURES

In this section, we will present three different GAN architectures - (i) Wasserstein GAN with Gradient Penalty (WGAN-GP), (ii) Conditional Wasserstein GAN (CWGAN) and (iii) a new novel GAN architecture called Pilot GAN. We will present all three architectures in the context of narrowband MIMO channel generation (refer Remark 1), in keeping with the GCE framework outlined in Section III. In other words, we will assume that the generator in all GAN architectures will output the beamspace MIMO channel representation given by (4). Then we will describe how the second and third GAN architectures can be combined to form Pilot Conditional GAN, that in combination with the LOS Predictor presented in Section IV-D, can be trained OTA at the receiver across a range of channel models.

Remark 1: Extension to wideband channels could be performed by using GCE for each pilot subcarrier and subsequently interpolating between subcarriers. Comprehensive practical evaluation in [34] has shown minimal performance gain in employing DL-based CE techniques over the OFDM resource grid.

A. Wasserstein GAN with Gradient Penalty

A Wasserstein Generative Adversarial Network (WGAN) [35] consists of two deep neural networks - a generator $\mathbf{G}(.;\theta_g)$ and a critic[1] $\mathbf{D}(.;\theta_d)$ - whose weights are optimized so as to solve the following min-max problem:

$$\min_{\mathbf{G}} \max_{\mathbf{D}\in\mathcal{D}} \mathbb{E}_{\mathbf{x}\sim\mathbb{P}_r(\mathbf{x})}\mathbf{D}(\mathbf{x}) - \mathbb{E}_{z\sim\mathbb{P}_z(z)}\mathbf{D}(\mathbf{G}(\mathbf{z})), \tag{7}$$

where $\mathcal{D}$ is the set of 1-Lipschitz functions[2] and $\mathbb{P}_r$ is the data distribution. The original GAN [25] is famously known to suffer from mode collapse [36] i.e. $\mathbb{P}_{\mathbf{G}}$ collapses to a delta function centered around the mode of the input data distribution. In [35], they attribute this behaviour to the use of the KL (Kullback-Leibler) or JS (Jensen-Shannon) divergence during training, and instead propose using the Wasserstein-1 distance. They show that sequences of probability distributions that converge under Wasserstein-1 are a superset of sequences convergent under KL or JS divergence, making WGAN more robust to mode collapse.

WGAN with Gradient Penalty (WGAN-GP) [30] improves the performance of WGAN by incorporating a penalty on the gradient norm for random samples $\hat{\mathbf{x}} \sim \mathbb{P}_{\hat{\mathbf{x}}}$ as a soft version of the Lipschitz constraint. In this paper, we will be utilizing both WGAN and WGAN-GP, depending on which training procedure yields better performance (further details in Section VI). In case of WGAN, the Lipschitz constraint $\mathbf{D} \in \mathcal{D}$ in (7) is enforced by clipping θ_d to be between $(-\tau, \tau)$, where τ is the clipping constant. In case of WGAN-GP, the Lipschitz constraint is enforced by adding

$$L_{\mathrm{GP}}(\theta_d) = \mathbb{E}_{\hat{\mathbf{x}}\sim\mathbb{P}_{\hat{\mathbf{x}}}}\big[(||\nabla_{\hat{\mathbf{x}}}\mathbf{D}(\hat{\mathbf{x}};\theta_d)||_2 - 1)^2\big] \tag{8}$$

to the objective in (7), where $\hat{\mathbf{x}}$ are points sampled uniformly along straight lines joining pair of points sampled from the data distribution $\mathbb{P}_r$ and the generator distribution $\mathbb{P}_{\mathbf{G}}$, since enforcing the gradient penalty over all possible inputs to $\mathbf{D}$ is intractable [30].

A unified algorithm capturing the training of both WGAN and WGAN-GP in the context of channel generation is outlined in Algorithm 1 by utilizing an indicator $\mathbb{1}_{\mathrm{GP}}$ to indicate if WGAN-GP was chosen or not. We update the critic and generator weights, θ_d and θ_g, using calls to Subroutine 2 and 3 respectively, in Algorithm 1. Both subroutines will be used in all

---

[1]WGAN does not refer to this as a discriminator since it is not trained to classify i.e. there is no Sigmoid activation in the final layer of a critic NN.

[2]A differentiable function f is said to be α-Lipschitz if $||\nabla_x f(x)||_2 \leq \alpha \ \forall \ x$.

subsequent GAN training algorithms. As part of updating the critic in Algorithm 1, we note that the gradient penalty is enforced only along straight lines between pairs of points sampled from the beamspace channel distribution $\mathbb{P}_{\mathbf{H}_{\mathrm{v}}}$ and the generator distribution $\mathbb{P}_{\mathbf{G}}$. In Section VI-A, we will highlight the NMSE improvement obtained by using WGAN-GP instead of WGAN.

Algorithm 1: Wasserstein GAN

for number of training iterations **do**

    for n_{d} iterations **do**

        Sample minibatch of m beamspace channel realizations $\{\mathbf{H}_{\mathrm{v}}^{(i)}\}_{i=1}^{m} \sim \mathbb{P}_{\mathbf{H}_{\mathrm{v}}}$, latent variables $\{\mathbf{z}^{(i)}\}_{i=1}^{m} \sim \mathbb{P}_z$ and random numbers $\{\epsilon^{(i)}\}_{i=1}^{m} \sim U[0,1]$.

        $\tilde{\mathbf{H}}_{\mathrm{v}} = \mathbf{G}(\mathbf{z}; \theta_g)$.

        $\hat{\mathbf{H}}_{\mathrm{v}} = \epsilon \mathbf{H}_{\mathrm{v}} + (1-\epsilon)\tilde{\mathbf{H}}_{\mathrm{v}}$.

        $\theta_d = \mathtt{Update\_D}(\tilde{\mathbf{H}}_{\mathrm{v}}, \mathbf{H}_{\mathrm{v}}, \hat{\mathbf{H}}_{\mathrm{v}}, \mathbb{1}_{\mathrm{GP}}, m, \gamma, \tau, \beta; \theta_d)$

    Sample minibatch of m latent variables $\{\mathbf{z}^{(i)}\}_{i=1}^{m} \sim \mathbb{P}_z$.

    $\theta_g = \mathtt{Update\_G}(\mathbf{G}(\mathbf{z}; \theta_{\mathbf{g}}), m, \gamma; \theta_g)$

Subroutine 2: $\theta_d = \mathtt{Update\_D}(\mathbf{x_G}, \mathbf{x}_r, \mathbf{x}_{r\mathbf{G}}, \mathbb{1}_{\mathrm{GP}}, m, \gamma, \tau, \beta; \theta_d)$

$$L(\theta_d) = \frac{1}{m}\sum_{i=1}^{m}\mathbf{D}(\mathbf{x_G}^{(i)}; \theta_d) - \mathbf{D}(\mathbf{x}_r^{(i)}; \theta_d) + \beta\mathbb{1}_{\mathrm{GP}}(||\nabla_{\mathbf{x}_{r\mathbf{G}}^{(i)}}D(\mathbf{x}_{r\mathbf{G}}^{(i)}; \theta_d)||_2 - 1)^2$$

$$\theta_d = \theta_d - \gamma\mathrm{RMSProp}(\nabla_{\theta_d}L(\theta_d))$$

$$\theta_d = \mathbb{1}_{\mathrm{GP}}\theta_d + (1 - \mathbb{1}_{\mathrm{GP}})\mathrm{clip}(\theta_d, -\tau, \tau)$$

Subroutine 3: $\theta_g = \mathtt{Update\_G}(\mathbf{x_G}, m, \gamma; \theta_g)$

$$L(\theta_g) = \frac{1}{m}\sum_{i=1}^{m}-\mathbf{D}(\mathbf{x_G}^{(i)}) \qquad \texttt{\% x\textbf{G} will be a function of } \theta_g$$

$$\theta_g = \theta_g - \gamma\mathrm{RMSProp}(\nabla_{\theta_g}L(\theta_g))$$

B. Conditional Wasserstein GAN

The WGAN in [23] was trained on channel realizations drawn from a single distribution which was characterized by a very strong Line-of-Sight (LOS) component. Moreover, training on such a channel distribution provides no indication of the generator's ability to learn more
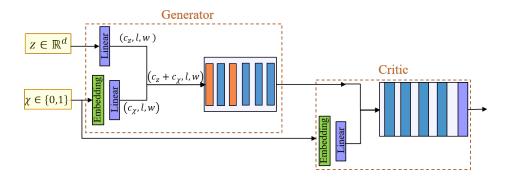
Fig. 2. Architecture of CWGAN to highlight how the condition χ is input to both $\mathbf{G}$ and $\mathbf{D}$. Only the path of the generated channel realizations $\mathbf{G}(z, \chi)$ through the GAN is shown here, a similar path can be traced for $\{\mathbf{H}_v^{(i)}\}_{i=1}^m$.

complex multi-path channels. In this section, we will present a Conditional WGAN (CWGAN) design that will have the ability to be trained on channel realizations drawn from a plurality of distributions, each yielding channel realizations with varying degrees of approximate sparsity in the beamspace domain.

We combine the architecture of Conditional GAN [37] with the training procedure of WGAN outlined in Algorithm 1 to develop CWGAN. While we will describe the NN architectures of the generator and discriminator in Section V-B, we have depicted how they are modified to accept as an input a condition in Fig. 2. For now, we assume that we are provided with a binary label χ indicating whether the channel we are trying to estimate is LOS ($\chi = 1$) or NLOS ($\chi = 0$). We will present different techniques to overcome this assumption during training and testing in Section IV-D. The condition χ is then passed through a learnable Embedding[3] layer, that embeds an integer as a high dimensional vector, followed by a linear and reshaping layer that has output dimensions (c_χ, l, w) as shown in Fig. 2. This embedded output is then appended along the channel dimension to $\mathrm{Linear}(\mathbf{z})$ of shape (c_z, l, w) to yield an input of size $(c_z + c_\chi, l, w)$ that is passed through the remaining deep convolutional generative network. A similar procedure is followed while inputting χ to the critic.

We now need to modify the WGAN training procedure outlined in Algorithm 1 to incorporate the conditional input. To this end, we simply sample $\{\mathbf{H}_v^{(i)}, \chi^{(i)}\}_{i=1}^m \sim \mathbb{P}_{\mathbf{H}_v}$ and utilize $\{\chi^{(i)}\}_{i=1}^m$ as input to $\mathbf{G}$ for computing $\tilde{\mathbf{H}}_v$ and as input to $\mathbf{D}$ for computing $L(\theta_d)$. Despite $\{\chi^{(i)}\}_{i=1}^m$ being input to $\mathbf{D}$, the derivative $\nabla_{\hat{\mathbf{H}}_v^{(i)}}$ continues to remain only w.r.t $\hat{\mathbf{H}}_v^{(i)}$. Subsequently, we

---

[3]https://pytorch.org/docs/stable/generated/torch.nn.Embedding.html

randomly sample m labels in $\{0, 1\}$ from a Bernoulli $\mathrm{Ber}(0.5)$ distribution and utilize these as the conditional input to both $\mathbf{G}$ and $\mathbf{D}$ for computing $L(\theta_g)$.

C. Pilot GAN

A large dataset of clean channel realizations was required to train a WGAN in [23]. Such a dataset can only be generated offline via simulation tools such as the MATLAB 5G Toolbox or Wireless InSite[4] and limits the applicability and adaptability of the trained generative models in practical deployments. Instead, we need to develop a framework to train GANs directly from noisy pilot measurements $\underline{\mathbf{y}}$, as defined in (3). To this end, we will combine LS CE (refer Remark 2) and Ambient GAN [31] in order to train a GAN from full rank received pilot measurements and call this framework Pilot GAN. First we present a brief overview of Ambient GAN and the conditions under which it can provably recover the true underlying data distribution.

Ambient GAN: Ambient GAN [31] was designed with the objective of training a GAN given only lossy measurements of images from the distribution of interest. It was based on the following key idea: rather than distinguish a real image from a generated image as is traditionally done in a GAN, the Ambient GAN discriminator must distinguish a real measurement from a simulated measurement of a generated image, assuming the measurement process is known. They considered a wide variety of measurement processes, for example convolution with a kernel, additive noise, and projection on a random Gaussian vector, and demonstrated that Ambient GAN trained a generator that produced images with good visual quality in spite of the noisy measurement processes.

The space of permissible measurement processes was determined by the following condition: there should be a *unique* data distribution $\mathbb{P}_r$ consistent with the observed measurement distribution $\mathbb{P}_y$. In other words, there must be an invertible mapping between $\mathbb{P}_r$ and $\mathbb{P}_y$ even though the map from an individual image to its measurement may not be invertible.

Combining LS CE and Ambient GAN: In Section II, we described how the rank of the measurement matrix $\mathbf{A}$ is bounded by N_s^2, even if $N_p > N_s$. Hence, let us assume that $N_p = N_s$. In order to obtain a set of full rank measurements, we will consider $K = \lceil N_t/N_p \rceil$ consecutive transmissions each with a different $(\mathbf{F}[i], \mathbf{S}[i], \mathbf{W}[i])$ triplet. Assuming the channel $\mathbf{H}$ remains

---

[4]https://www.remcom.com/wireless-insite-em-propagation-software/

constant over K consecutive transmissions, we can stack the K received signals to write $\underline{\mathbf{y}}[1:K]$ as

$$
\begin{bmatrix}
\mathbf{S}[1]^T\mathbf{F}[1]^T \otimes \mathbf{W}[1]^H \\
\vdots \\
\mathbf{S}[K]^T\mathbf{F}[K]^T \otimes \mathbf{W}[K]^H
\end{bmatrix}
\underline{\mathbf{H}} +
\begin{bmatrix}
\mathbf{I}_{N_p} \otimes \mathbf{W}[1]^H & \mathbf{0} & \cdots & \mathbf{0} \\
\vdots & \vdots & \ddots & \vdots \\
\mathbf{0} & \mathbf{0} & \cdots & \mathbf{I}_{N_p} \otimes \mathbf{W}[K]^H
\end{bmatrix}
\begin{bmatrix}
\underline{\mathbf{n}}[1] \\
\vdots \\
\underline{\mathbf{n}}[K]
\end{bmatrix}
\tag{9}
$$

By appropriately sampling the entries of $\mathbf{F}[i]$, $\mathbf{S}[i]$ and $\mathbf{W}[i]$ for all $i \in [K]$, we can ensure that the stacked measurement matrix $\mathbf{A}[1:K]$ has full rank $N_t N_r$. Hence, the beamspace LS channel estimate is given by

$$
\underline{\mathbf{H}}_{\mathrm{v},LS} = \underline{\mathbf{H}}_{\mathrm{v}} + \boldsymbol{\zeta},
\tag{10}
$$

where $\boldsymbol{\zeta}$ is a zero-mean complex Gaussian random vector of covariance $\boldsymbol{\Sigma}$ such that

$$
\boldsymbol{\Sigma}^{\frac{1}{2}} = (\mathbf{A}_{\mathrm{T}}^T \otimes \mathbf{A}_{\mathrm{R}}^H)(\mathbf{A}[1:K])^\dagger \mathrm{diag}(\{\mathbf{I}_{N_p} \otimes \mathbf{W}[i]^H\}_{i=1}^{K}),
\tag{11}
$$

where $(\mathbf{A}[1:K])^\dagger$ denotes the pseudo-inverse of $\mathbf{A}[1:K]$. The pseudo-inverse provides for $K > N_t/N_p$. If $K = N_t/N_p$, this simply reduces to $(\mathbf{A}[1:K])^{-1}$. In Ambient GAN terms, $\underline{\mathbf{H}}_{\mathrm{v},LS}$ is the lossy measurement, while the measurement process is additive complex Gaussian noise with covariance $\boldsymbol{\Sigma}$. Subsequently, training an Ambient GAN now simply involves adding noise of the same covariance $\boldsymbol{\Sigma}$, as given by (11), to $\underline{\mathbf{G}}(\mathbf{z})$. Given that these are pilot transmissions, we can assume that the transmitter and receiver have agreed upon a set of K $(\mathbf{F}[i], \mathbf{S}[i], \mathbf{W}[i])$ triplets, that are known to both. Since $\boldsymbol{\Sigma} = f\big(\{(\mathbf{F}[i], \mathbf{S}[i], \mathbf{W}[i])\}_{i=1}^{K}\big)$, we can easily generate complex Gaussian noise of the desired covariance $\boldsymbol{\Sigma}$ and add it to the output of the generator to generate fake beamspace LS channel estimates $\underline{\mathbf{G}}_{\mathrm{v},LS}$ as shown in Fig. 3. Having thus performed LS CE to obtain the dataset $\{\underline{\mathbf{H}}_{\mathrm{v},LS}^{(i)}\}$, and added correlated noise to the generator output to obtain $\{\underline{\mathbf{G}}_{\mathrm{v},LS}^{(i)}\}$, we can perform the same WGAN training procedure outlined in Algorithm 1 to train the Pilot GAN.

Throughout the exposition of Pilot GAN, we have assumed full-rank pilot measurements in order to perform LS CE. This can be attributed to the usage of Ambient GAN which requires there to be a unique distribution $\mathbb{P}_{\mathbf{H}_\mathrm{v}}$ consistent with the observed pilot measurements $\mathbb{P}_{\mathbf{Y}}$. It can be easily shown that additive noise preserves distributional invertibility (refer Section 10.3 in [31]). However, if we were to multiply $\mathbf{H}_\mathrm{v}$ by a compressive measurement matrix $\mathbf{A}$, we cannot
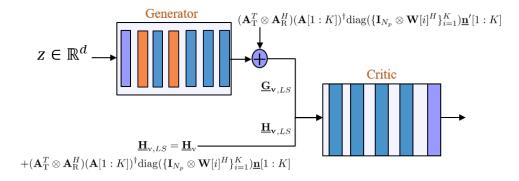
Fig. 3. Architecture of Pilot GAN. Correlated complex noise is added to the output of **G** before passing it to the critic. Complex observations are stacked as real and imaginary along the channel dimension before being input to the critic.

express $\mathbb{P}_{\mathbf{Y}}$ as a unique function of $\mathbb{P}_{\mathbf{H}_v}$, even in the noiseless setting (refer Appendix A). Hence, we have utilized full-rank pilot measurements for *training* Pilot GAN, even though it is important to note that during *inference* the generative model uses compressive measurements for channel estimation.

Federated Pilot GAN: In a setting where $U > 1$ UEs are connected to a single BS, we can easily extend the Pilot GAN training to a federated implementation by utilizing the following insight – only the critic training requires the $\{\underline{\mathbf{H}}_{v,LS}^{(i)}\}$ dataset, the generator training on the other hand does not [38]. Hence we can train the generator **G** at the BS in a centralized fashion, while the critic training is carried out locally at the UEs and its gradient updates are averaged at the BS. The training procedure for Federated Pilot GAN is presented in Algorithm 4. In summary, we assume that each UE u periodically accumulates a small dataset of D beamspace LS channel estimates and uses that to train its critic $\mathbf{D}_u$. After each UE has performed n_d critic training iterations, it transmits the critic weights $\theta_{d,u}$ to the BS where they are averaged to obtain θ_d and used to train **G**, following which both θ_g and θ_d are broadcast to all UEs. Since **G** is typically significantly more complex than **D** (refer Table III), this ensures that the bulk of the training is carried out at the BS.

It should be noted that we assume error free transmission of the federated updates in both the uplink and downlink, which can be achieved by utilizing sub-6 GHz channels for these transmissions as is commonly done in Carrier Aggregation in 5G systems, i.e. utilizing a lower bandwidth channel at a lower f_c along with mmWave.

Remark 2: Unlike channel estimation, generative model training is not a latency sensitive operation. Consequently, waiting K transmissions to obtain a LS channel estimate to train

---

Algorithm 4: Federated Pilot GAN with U UEs.

---

for number of rounds **do**

    UE u accumulates D beamspace LS channel estimates $\{\mathbf{H}_{v,LS,u}^{(i)}\}_{i=1}^{D} \ \forall u \in [U]$

    for l training iterations **do**

        BS broadcasts θ_g and θ_d to all UEs

        for $u = 1 : U$ **do**

            $\theta_{d,u} = \theta_d$

            for n_d iterations **do**

                Sample minibatch $\mathbf{B}_u$ of size m/U from $\{\mathbf{H}_{v,LS,u}^{(i)}\}_{i=1}^{D}$, latent variables $\{\mathbf{z}^{(i)}\}_{i=1}^{m/U} \sim \mathbb{P}_z$ and random numbers $\{\epsilon^{(i)}\}_{i=1}^{m/U} \sim U[0,1]$.

                $\underline{\mathbf{G}}_{v,LS} = \underline{\mathbf{G}}(\mathbf{z}) + \mathcal{CN}(\mathbf{0}, \mathbf{\Sigma})$, with $\mathbf{\Sigma}$ given by (11).

                $\hat{\mathbf{B}}_u = \epsilon \mathbf{B}_u + (1 - \epsilon) \mathbf{G}_{v,LS}$.

                $\theta_{d,u} = \texttt{Update\_D}(\underline{\mathbf{G}}_{v,LS}, \mathbf{B}_u, \hat{\mathbf{B}}_u, \mathbb{1}_{GP}, m/U, \gamma, \tau, \beta; \theta_{d,u})$

        UE u sends $\theta_{d,u}$ in uplink to BS

    $\theta_d = (1/U) \sum_{u=1}^{U} \theta_{d,u}$

    Sample minibatch of m latent variables $\{\mathbf{z}^{(i)}\}_{i=1}^{m} \sim \mathbb{P}_z$

    $\underline{\mathbf{G}}_{v,LS} = \underline{\mathbf{G}}(\mathbf{z}; \theta_g) + \mathcal{CN}(\mathbf{0}, \mathbf{\Sigma})$, with $\mathbf{\Sigma}$ given by (11).

    $\theta_g = \texttt{Update\_G}\big(\mathbf{G}_{v,LS}, m, \gamma; \theta_g\big)$

Pilot GAN should not be misinterpreted as implying that LS channel estimation can be used to benchmark the performance of GCE.

D. Pilot Conditional GAN with LOS Predictor

Having presented a framework for training a generative model from pilot measurements, the sensible next step would be to combine the CWGAN framework presented in Section IV-B with the Pilot GAN framework of Section IV-C to train a generative model from pilot measurements of all LOS and NLOS channels. However, prior to doing that, we will present a simple supervised learning based technique to determine the LOS/NLOS label χ from the received full-rank pilot measurements $\underline{\mathbf{y}}[1 : K]$.

1) LOS Predictor: Intuitively, we expect to easily distinguish between LOS and NLOS channel realizations by looking at the number and magnitude of non-zero entries in the beamspace representation $\mathbf{H}_v$. Since we assume full-rank pilot measurements, the beamspace LS channel estimates $\mathbf{H}_{v,LS}$ are noisy versions of $\mathbf{H}_v$. Hence, we simply train a NN $\mathcal{L}(.; \theta_L)$ in a supervised fashion to take as input $\mathbf{H}_{v,LS}$ and output $\mathbb{P}(\mathbf{H}_{v,LS} \in \text{LOS})$. Given LOS ground truth labels,

such a NN can be trained using the binary cross entropy (BCE) loss[5] and the condition χ is given by

$$\chi = \frac{1}{2}\big(1 + \operatorname{sgn}\big(2\mathcal{L}(\mathbf{H}_{v,LS}; \theta_L) - 1\big)\big). \tag{12}$$

It may appear that the supervised training of an LOS Predictor is at odds with the overarching goal of this paper to design an OTA training procedure. However, the minimal loss in accuracy of the LOS predictor during inference, when trained on only a subset of the CDL channel models as demonstrated in Section VI-D, will reinforce the practicality of the proposed approach.

As explained in Section IV-C, we require full rank pilot measurements to perform LS CE. While such pilot measurements would be available during Pilot GAN training, the measurements received while performing GCE would be compressive. However, the requirement of an LOS predictor during GCE can be easily circumvented by modifying the optimization in (5) to

$$\mathbf{z}^* = \min_{\chi \in \{0,1\}} \arg\min_{\mathbf{z} \in \mathbb{R}^d} \ ||\underline{\mathbf{y}} - \mathbf{A}_{\mathrm{sp}}\underline{\mathbf{G}}(\mathbf{z}, \chi)||_2^2 + \lambda_{\mathrm{reg}}||\mathbf{z}||_2^2. \tag{13}$$

Note that since $\chi \in \{0,1\}$, it only increases the complexity by a constant factor of 2.

2) Pilot Conditional GAN: Having designed a LOS predictor, we now present a unified algorithm to train a generative model from noisy full-rank pilot measurements that is applicable across all MIMO channel models – PCGAN. The technique is summarized in Algorithm 5.

It is worth emphasising the completely unsupervised and OTA nature of this GAN implementation (aside from the usage of the LOS Predictor). Having received pilot measurements $\underline{\mathbf{y}}[1:K]$, we first compute the beamspace LS channel estimate $\mathbf{H}_{v,LS}$. These channel estimates are used to train a conditional Wasserstein GAN. Subsequently, for CE, we will extract the conditional generator $\mathbf{G}$ and compute the beamspace channel estimate $\mathbf{G}(\mathbf{z}^*)$ from compressive pilot measurements $\underline{\mathbf{y}}$ by solving (13). Hence we have utilized pilot measurements to *train a channel estimator* as well as *perform channel estimation*.

V. SIMULATION DETAILS

A. Data Generation & Pre-processing

MIMO channel realizations have been generated using the MATLAB 5G Toolbox. Specifically, in accordance with 3GPP TR 38.901 [39], we have generated training and test datasets consisting

---

[5]https://pytorch.org/docs/stable/generated/torch.nn.BCELoss.html

---

Algorithm 5: Pilot Conditional GAN with LOS Predictor $\mathcal{L}(.; \theta_L)$.

---

for number of training iterations **do**

    for n_{d} iterations **do**

        Sample minibatch of m beamspace LS channel estimates $\{\mathbf{H}_{\mathrm{v},LS}^{(i)}\}_{i=1}^m$, latent variables $\{\mathbf{z}^{(i)}\}_{i=1}^m \sim \mathbb{P}_z$ and random numbers $\{\epsilon^{(i)}\}_{i=1}^m \sim U[0,1]$.

        Compute $\{\chi^{(i)}\}_{i=1}^m$ using (12).

        $\underline{\mathbf{G}}_{\mathrm{v},LS} = \underline{\mathbf{G}}(\mathbf{z}, \chi) + \mathcal{CN}(\mathbf{0}, \boldsymbol{\Sigma})$, with $\boldsymbol{\Sigma}$ given by (11).

        $\hat{\mathbf{H}}_{\mathrm{v},LS} = \epsilon \mathbf{H}_{\mathrm{v},LS} + (1-\epsilon)\mathbf{G}_{\mathrm{v},LS}$.

        $\theta_d = \texttt{Update\_D}\big(\{\mathbf{G}_{\mathrm{v},LS}, \chi\}, \{\mathbf{H}_{\mathrm{v},LS}, \chi\}, \{\hat{\mathbf{H}}_{\mathrm{v},LS}, \chi\}, \mathbb{1}_{\mathrm{GP}}, m, \gamma, \tau, \beta; \theta_d\big)$

    Sample minibatch of m latent variables $\{\mathbf{z}^{(i)}\}_{i=1}^m \sim \mathbb{P}_z$ and conditions $\{\chi^{(i)}\}_{i=1}^m \sim \mathrm{Ber}(0.5)$.

    $\underline{\mathbf{G}}_{\mathrm{v},LS} = \underline{\mathbf{G}}(\mathbf{z}, \chi; \theta_g) + \mathcal{CN}(\mathbf{0}, \boldsymbol{\Sigma})$, with $\boldsymbol{\Sigma}$ given by (11).

    $\theta_g = \texttt{Update\_G}\big(\{\mathbf{G}_{\mathrm{v},LS}, \chi\}, m, \gamma; \theta_g\big)$

Delay Profile	CDL - A,B,C,D,E		Antenna Array Type	ULA
Dataset Size	Train - 6000×5		Antenna Spacing	$\lambda_c/2$
	Test - 50×5		Sampling Rate	15.36 MHz
Subcarrier Spacing	15 kHz		Carrier Frequency	40 GHz
N_{t}	64		Delay Spread	30 ns
N_{r}	16		Doppler Shift	5 Hz

TABLE I: Data Generation Parameters

of an equal number of realizations of all categories of CDL channels i.e. CDL-A, B, C (which are NLOS) and CDL-D, E (which are LOS). The channel simulation parameters are summarized in Table I. Since we assume a narrowband block fading model in this paper, we simply extract the (N_r, N_t) matrix corresponding to the first subcarrier and first OFDM symbol from the $(14, 12, N_r, N_t)$ channel realizations generated by MATLAB. The aforementioned channel realizations will be used to generate LS channel estimates, as defined in (10), which in turn will be used to populate the datasets employed for simulating the OTA training procedures of Algorithm 4 and 5.

The generator output $\mathbf{G}(z)$ and discriminator input are of size $(2, N_t, N_r)$, where the first dimension allows us to stack the real and imaginary parts. Based on empirical evidence that a GAN is unable to learn mean-shifted distributions [36] as well as our own prior experience in training GANs for spatial channel matrix generation [23], it is important to normalize the data used to train a GAN. Given a beamspace channel realization $\mathbf{H}_{\mathrm{v}}$, $\mu[i,j] = \mathbb{E}\big[\mathbf{H}_{\mathrm{v}}[i,j]\big]$,

$\text{Re}(\sigma[i,j]) = \big(\mathbb{E}\big[(\text{Re}(\mathbf{H}_{\text{v}}[i,j]-\mu[i,j]))^2\big]\big)^{0.5}$ and $\text{Im}(\sigma[i,j]) = \big(\mathbb{E}\big[(\text{Im}(\mathbf{H}_{\text{v}}[i,j]-\mu[i,j]))^2\big]\big)^{0.5}$,

we normalize the matrix element-wise as

$$\text{Re}(\mathbf{H}_{\text{v}}[i,j]) \leftarrow \frac{\text{Re}(\mathbf{H}_{\text{v}}[i,j]-\mu[i,j])}{\text{Re}(\sigma[i,j])} \quad \text{Im}(\mathbf{H}_{\text{v}}[i,j]) \leftarrow \frac{\text{Im}(\mathbf{H}_{\text{v}}[i,j]-\mu[i,j])}{\text{Im}(\sigma[i,j])}. \quad (14)$$

We will compute a single $\mu[i,j]$ and $\sigma[i,j]$ for each $(i,j) \in [N_t] \times [N_r]$ over the entire training dataset of each experiment. Such statistics can be computed from the dataset of beamspace LS channel estimates $\{\mathbf{H}_{\text{v},LS}^{(i)}\}$ in an OTA training procedure. In lieu of (14), the operations $\mathbf{G}(.)$ and $\mathbf{D}(.,\chi)$ will implicitly be used to denote $\text{SN}^{-1}(\mathbf{G}(.))$ and $\mathbf{D}(\text{SN}(.),\chi)$ respectively throughout the paper without exception. Here SN denotes the operation of *stacking* the real and imaginary part followed by *normalization* using $\{\mu[i,j],\sigma[i,j]\}_{i,j}$ and SN^{-1} corresponds to *unnormalization* followed by *unstacking* to generate a complex-valued output.

B. Neural Network Architectures & Training Hyperparameters

The basic neural network architectures of the generator $\mathbf{G}$, critic $\mathbf{D}$ and LOS Predictor $\mathcal{L}$ have been detailed in Table II. All layers and their corresponding parameters are described in notation standard to PyTorch [40]. Since the critic and the LOS Predictor have similar architectures, we have merged their representations and used the indicator $\mathbb{1}_{\mathcal{L}}$ to indicate layers that are only in the LOS Predictor and not the critic. In particular, the critic in a WGAN-GP implementation cannot have batch normalization since it invalidates the gradient penalty computation during critic training (refer Section 4 in [30]). All BatchNorm2D layers have $\text{momentum} = 0.8$ [35] and Conv2D layers have $\text{bias} = \text{False}$. We utilize $d = 65$ for $\mathbf{z} \in \mathbb{R}^d$ (refer Appendix B for an empirical justification).

In order to extend the generator and critic architectures to the conditional setting, we employ an $\text{Embedding}(2,10)$ layer in both. This layer learns a 10-dimensional embedding for $\chi = 0$ and $\chi = 1$. Subsequently, $\mathbf{G}$ passes this embedding through $\text{Linear}(10, N_t N_r/16)$ and $\text{Reshape}(1, N_t/4, N_r/4)$ before concatenating it to $\text{Linear}(\mathbf{z})$ of size $(127, N_t/4, N_r/4)$ as shown in Fig. 2. Similarly, $\mathbf{D}$ passes the embedding through $\text{Linear}(10, N_t N_r)$ and $\text{Reshape}(1, N_t, N_r)$ before concatenating it to the input of size $(2, N_t, N_r)$. Table III contains the total number of parameters in $\mathbf{G}$ and $\mathbf{D}$ in the both the conditional and unconditional setting, along with the LOS Predictor $\mathcal{L}$.

In Algorithm 1, 4 and 5, we set $n_{\text{d}} = 5$, $\beta = 10$, $\tau = 0.01$ and $\gamma = 0.00005$ [30] [35]. We utilize a minibatch size of $m = 200$ in all GAN training. For performing GCE, we utilize

19

L	Generator	Critic / LOS Predictor
1	Linear$(65, 8N_tN_r)$	Conv2D$(2, 16, (3,3), \text{stride} = 2)$
	ReLU	LeakyReLU(0.2) + Dropout(0.25)
2	Reshape$(128, N_t/4, N_r/4)$	Conv2D$(16, 32, (3,3), \text{stride} = 2)$
3	Upsample(2)	ZeroPad2D$((0,1,0,1))$
	Conv2D$(128, 128, (4,4))$	LeakyReLU(0.2)+Dropout(0.25)+$\mathbb{1}_\mathcal{L}$BatchNorm2D(32)
	BatchNorm2D(128)	Conv2D$(32, 64, (3,3), \text{stride} = 2)$
	ReLU	LeakyReLU(0.2)+Dropout(0.25)+$\mathbb{1}_\mathcal{L}$BatchNorm2D(64)
4	Upsample(2)	Conv2D$(64, 128, (3,3))$
	Conv2D$(128, 128, (4,4))$	LeakyReLU(0.2)+Dropout(0.25)+$\mathbb{1}_\mathcal{L}$BatchNorm2D(128)
	BatchNorm2D(128)	Flatten
	ReLU	Linear$(3456, 1)$
5	Conv2D$(128, 2, (4,4))$	$\mathbb{1}_\mathcal{L}$ Sigmoid

TABLE II: Architecture Details of Generator $\mathbf{G}$, Critic $\mathbf{D}$ and LOS Predictor $\mathcal{L}$. The layer index is denoted by $\mathbf{L}$ in the top left corner.

	WGAN-GP	CWGAN
Generator $\mathbf{G}$	1,069,568	1,328,468
Critic $\mathbf{D}$	100,753	112,181
LOS Predictor $\mathcal{L}$	101,201	

TABLE III: Parameter Count of $\mathbf{G}$, $\mathbf{D}$ and $\mathcal{L}$

an Adam [41] optimizer with a step size $\eta = 0.1$, $\lambda_{\text{reg}} = 0.001$ and iteration count 100. We also determined empirically that resetting the RMSProp optimizer for the critic at every training iteration improved the performance of Algorithm 1 (refer Appendix C)[6].

C. Compressed Sensing (CS) Baselines

We compare the performance of GCE with a few standard CS baselines. It should be noted that the objective of this paper is to design an OTA GAN training procedure for CE, while using GCE to perform channel estimation. As described in Section I-B, there have been recent works in the deep generative domain [26], [27] that have the potential to further improve the performance and robustness of CE using generative networks. Analyzing the performance and viability of such methods for mmWave MIMO CE is left for future work.

[6]Code and data will be made publicly available at https://github.com/akashsdoshi96/ota-gan-mimo-ce

i) *Orthogonal Matching Pursuit (OMP):* As described in [11], OMP minimizes $||\underline{\mathbf{H}}_v||_0$ subject to $||\underline{\mathbf{y}} - \mathbf{A}_{sp}\underline{\mathbf{H}}_v||_2 \leq \sigma$. To prevent overfitting to the noise, OMP must be stopped[7] when the energy in the residual is less than σ^2.

ii) *EM-GM-AMP:* We utilize the Approximate Message Passing algorithm EM-GM-AMP [13], which accepts as input $\underline{\mathbf{y}}$ and $\mathbf{A}_{sp}$, and recovers the channel estimate $\underline{\mathbf{H}}_v$.

Note that unlike the CS baselines, whose number of iterations depend on the SNR level, GCE is run for a constant number of iterations at all SNR without its performance being impacted.

VI. RESULTS & DISCUSSION

In this section, we will present and analyze the results for each GAN architecture presented in Section IV in sequence, before concluding with results from PCGAN. The performance metric used to assess the quality of the channel estimate will be the NMSE as defined in (6). We will primarily be presenting two figures/tables for each architecture : GCE based NMSE as a function of i) GAN training iterations at a fixed SNR and ii) SNR for a given generative model. All figures that plot NMSE vs training iterations have been evaluated at an SNR of 15 dB and we will utilize $N_s = 16$, $N_p = 25$, $N_{bit,t} = 6$ and $N_{bit,r} = 2$, unless otherwise stated.

A. WGAN-GP

We design a separate generator for each of the five CDL channel models by training a WGAN-GP ($\mathbb{1}_{GP} = 1$) using Algorithm 1 for 60000 training iterations. The smoothed[8] plot of NMSE vs training iterations is shown in Fig. 4a. We have also highlighted the gain of using WGAN-GP over WGAN in Fig. 4b. Using Fig. 4a, we extract the trained generator with the lowest NMSE for each channel model (refer Remark 3) and plot NMSE vs SNR in Fig. 5. Furthermore, in Table IV, we have compared the GCE based NMSE with the baselines outlined in Section V-C at three distinct SNR $= \{-5, 0, 10\}$ dB[9].

We observe that across CDL channel models, the performance of GCE is consistently $B < C < A < E < D$. This is in agreement with the decreasing number of rays/clusters and the increasing magnitude of the LOS component in $\mathbf{H}_v$ as we go from left to right (refer Table 7.7.1 of

---

[7]We set the maximum number of iterations to 200.

[8]Figures have occasionally been smoothed using a Hanning window of size 6 to improve readability, as and where mentioned.

[9]SNR $= -5$ dB is a commonly utilized minimum SNR threshold for initial access [42].

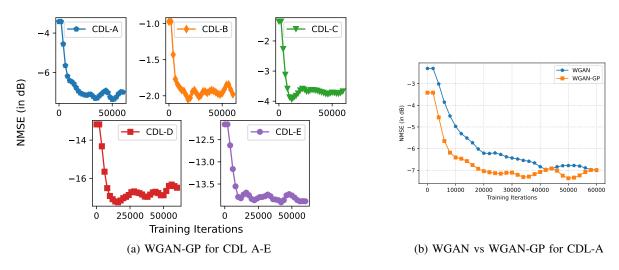(a) WGAN-GP for CDL A-E

(b) WGAN vs WGAN-GP for CDL-A

Fig. 4. NMSE vs Training Iterations. NMSE increases with decrease in approximate beamspace sparsity.
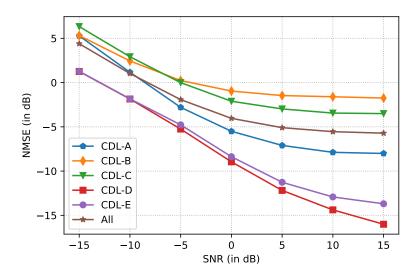

Fig. 5. NMSE vs SNR for WGAN-GP for CDL A-E. "All" denotes an average over all channel models.

[39] for the precise channel profiles). Moreover, GCE consistently outperforms OMP, while also outperforming EM-GM-AMP for all CDL models excluding CDL-B, for which the performance of GCE and EM-GM-AMP are approximately similar. GCE also significantly outperforms both baselines at all SNR for LOS channels – CDL-D and CDL-E.

*Remark 3: All WGAN models designed in Section IV are trained to directly minimize $L(\theta_d)$ and maximize $-L(\theta_g)$, **not** to minimize NMSE. While [35] has noted that the learning curves of a WGAN are empirically seen to correlate well with observed sample quality, this does not imply*

		CDL-A	CDL-B	CDL-C	CDL-D	CDL-E
	GCE	**-2.80**	**0.23**	**-0.01**	**-5.25**	**-4.78**
SNR = -5 dB	OMP	2.26	2.31	2.28	0.48	1.82
	EM-GM-AMP	0.35	0.59	0.52	-1.08	-0.09
	GCE	**-5.50**	**-0.95**	**-2.11**	**-8.95**	**-8.38**
SNR = 0 dB	OMP	1.22	1.39	1.31	-1.76	-0.25
	EM-GM-AMP	-0.67	0.14	0.04	-2.70	-1.71
	GCE	**-7.88**	**-1.60**	**-3.45**	**-14.38**	**-12.92**
SNR = 10 dB	OMP	0.73	0.64	0.55	-3.26	-2.21
	EM-GM-AMP	-1.83	-1.55	-1.63	-2.80	-2.61

TABLE IV: NMSE (in dB) of WGAN-GP based GCE and CS Baselines for CDL A-E

that the NMSE will decrease smoothly with training iterations as will be evident in most NMSE vs training iteration plots in this paper. NMSE convergence smoothness will be further degraded when we use noisy data in Pilot GAN and PCGAN in Section VI-C and VI-D respectively.

Impact of mutual coherence of the sensing matrix $\mu(\mathbf{A}_{\mathrm{sp}})$: Consider the standard CS problem – determine $\mathbf{x}$ given $\mathbf{y} = \mathbf{Ax} + \mathbf{n}$ with $\mathbf{A} \in \mathbb{R}^{m \times n}$ and $m < n$. Suppose $\mathbf{A} = [\mathbf{a}_1, \ldots \mathbf{a}_n]$, then the mutual coherence of $\mathbf{A}$ is given by

$$\mu(\mathbf{A}) = \max_{i,j;i \neq j} \frac{|\mathbf{a}_i^H \mathbf{a}_j|}{||\mathbf{a}_i||_2 ||\mathbf{a}_j||_2}.$$

Compressed sensing using generative models [5] [43] typically employs measurement matrices $\mathbf{A}$ whose entries are i.i.d. and chosen from simple random distributions such as Gaussian or sub-Gaussian. Consequently, it is highly likely that $\mathbf{A}$ is a full-rank matrix with low cross correlation among its columns. In other words, $\mathrm{rank}(\mathbf{A}) = m$ with high probability, and the mutual coherence $\mu(\mathbf{A})$ will be low. For a given $\mu(\mathbf{A})$, if $\mathbf{x}$ satisfies $||\mathbf{x}||_0 < \frac{1}{2}\left(\frac{1}{\mu(\mathbf{A})} + 1\right)$, then there exists a unique $\mathbf{x}$ with the given $\mathcal{L}_0$ norm that satisfies $\mathbf{y} = \mathbf{Ax}$ [17]. Hence low mutual coherence decreases the reconstruction error of $\mathbf{x}$. Owing to the low $\mu(\mathbf{A})$, we believe that [5] fails to truly convey the advantage of using generative models for compressed sensing over other traditional CS baselines for CE.

In Section III, we defined the measurement matrix as $\mathbf{A}_{\mathrm{sp}} = (\mathbf{A}_{\mathrm{T}}^H \mathbf{FS})^T \otimes \mathbf{W}^H \mathbf{A}_{\mathrm{R}}$, with $\mathrm{rank}(\mathbf{A}_{\mathrm{sp}}) \leq N_s^2$. By varying N_s from 1 to 16 for $N_p = 16$ and $N_p = 25$, we observe that $\mu(\mathbf{A}_{\mathrm{sp}}) \geq 0.75$ in Fig. 6a. This implies that a unique $\mathbf{H}_{\mathrm{v}}$ can be recovered by OMP only if $||\mathbf{H}_{\mathrm{v}}||_0 < 2$. This is consistent with our observations in Table IV, where OMP performs quite
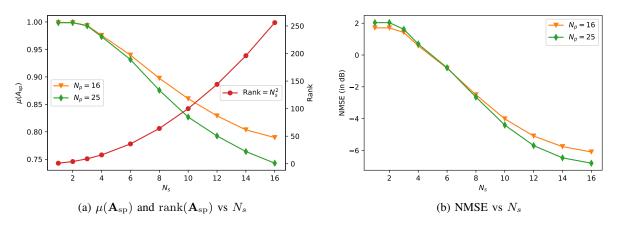
(a) $\mu(\mathbf{A}_{\mathrm{sp}})$ and $\mathrm{rank}(\mathbf{A}_{\mathrm{sp}})$ vs N_s

(b) NMSE vs N_s

Fig. 6. NMSE, $\mu(\mathbf{A}_{\mathrm{sp}})$ and $\mathrm{rank}(\mathbf{A}_{\mathrm{sp}})$ vs N_s for CDL-A at SNR = 15 dB. Low $\mu(\mathbf{A}_{\mathrm{sp}})$ decreases NMSE.

poorly. Meanwhile, GCE significantly outperforms OMP, showing that the rich generative prior compensates for the high mutual coherence of $\mathbf{A}_{\mathrm{sp}}$. Thus, GCE eliminates the need for careful tuning of the entries of $\mathbf{A}_{\mathrm{sp}}$ to minimize $\mu(\mathbf{A}_{\mathrm{sp}})$ as is done in [11].

However, the performance of GCE also improves if we improve the mutual coherence. In Fig. 6b, we can observe that for the same rank at $N_s = 16$, when the value of $\mu(\mathbf{A}_{\mathrm{sp}})$ is higher for $N_p = 16$ than $N_p = 25$, GCE does attain a lower NMSE with $N_p = 25$.

B. CWGAN

We now design a single conditional generative model by training a CWGAN ($\mathbb{1}_{\mathrm{GP}} = 0$) as outlined in Section IV-B for 100,000 training iterations. For CWGAN, we assume that the LOS/NLOS ground truth label χ is known while performing GCE. The smoothed plot of NMSE vs training iterations is shown in Fig. 7a, and the plot of NMSE vs SNR for the best generative model (determined using Fig. 7a) is shown in Fig. 7b. The degradation in average NMSE compared to the individually trained WGAN-GP models increases from 0.5 dB at -5 dB SNR to 1.7 dB at 0 dB SNR to 2.6 dB at 10 dB SNR. However, GCE using CWGAN continues to outperform OMP at all the aforementioned SNR for CDL A-E, and even outperforms EM-GM-AMP for CDL-A, D and E, with the gain being largest for LOS channel models (~ 6 dB) and ~ 0.7 dB for CDL-A.

The reduction in gain on switching to a conditional model can be attributed to the simple binary condition being utilized. For instance, CDL-A and CDL-B are both NLOS channel models, but have significantly different beamspace representations. We are actively investigating the

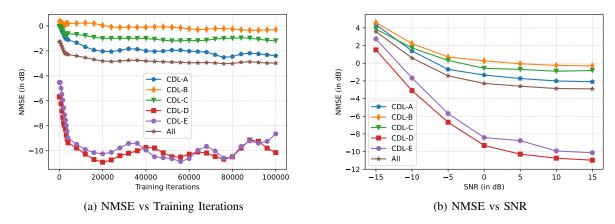(a) NMSE vs Training Iterations (b) NMSE vs SNR

Fig. 7. NMSE for CWGAN trained on CDL A-E. "All" denotes an average over all channel models.

possibility of introducing a learnable condition into the GAN framework such that the condition more accurately reflects the estimated degree of approximate sparsity in $\mathbf{H}_v$.

C. Pilot GAN

We consider two cases, $U = 1$ and $U = 4$, where U is the number of UEs associated with a BS, and utilize full-rank pilot measurements i.e. of rank $N_t N_r$ for training in both cases with $\mathbb{1}_{\mathrm{GP}} = 1$.

1) $U = 1$: We train Pilot GAN as outlined in Section IV-C for 60,000 training iterations. We investigate the noise tolerance of Pilot GAN by training a generative model for CDL-A at SNR $= \{10, 20, 30, 40\}$ dB. The SNR value will determine the variance σ^2 of each term in $\underline{\mathbf{n}}[1 : K]$ in (9) during training. The smoothed plot of NMSE vs training iterations is shown in Fig 8a, and the plot of NMSE vs SNR in Fig. 8b. From Fig. 8a, we can see that the NMSE at a training SNR of 40 dB is roughly similar to the noiseless case, while the NMSE convergence at a training SNR of ≤ 20 dB is significantly degraded. However, the NMSE improves at SNR ≤ -10 dB over WGAN-GP for all training SNR, where WGAN-GP in Fig. 8b denotes training on clean channel realizations. This could be a possible advantage of training with noisy data.

While this might seem a high SNR threshold to meet, two points can be noted: i) Training a single generative model for a larger class of CDL channels could potentially reduce the SNR induced degradation, as we will see in Section IV-D and ii) Practical wireless deployments [44] use precoders and combiners attuned to channel realizations from a few time-slots ago, which yields significantly better link SNR than random precoders and combiners.
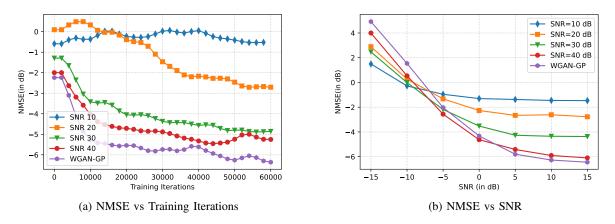
(a) NMSE vs Training Iterations

(b) NMSE vs SNR

Fig. 8. NMSE for Pilot GAN trained on CDL-A. Note that the labels are *training* SNR, while the x-axis in the right figure is the *test* SNR. WGAN-GP denotes no noise during training. NMSE increases at lower *training* SNR.
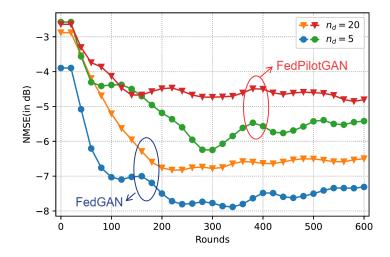

Fig. 9. NMSE vs Rounds for Federated Pilot GAN training on CDL-A. Note the trade-off between BS-UE communication frequency and NMSE, as well as between training noise levels and NMSE.

2) $U = 4$: We train Federated Pilot GAN as outlined in Algorithm 4 for 600 rounds on CDL-A, with $D = 500$ and $m = 200$. Instead of fixing the link SNR, we perform a network level simulation using the parameters outlined in Table V. We fix the large scale path loss for the duration of the simulation, and assume it is known at the UE. We investigate the impact of two factors on the NMSE convergence vs rounds – i) noise in LS channel estimate and ii) BS-UE communication frequency. To analyze i), we consider perfect channel estimates i.e. $\mathbf{H}_{v,LS} = \mathbf{H}_v$ for training the GAN, and refer to this as "FedGAN", while the GAN trained on LS channel estimates generated using the parameters in Table V is referred to as "FedPilotGAN". In order to study ii), we change the number of critic updates (n_d in Algorithm 4) while keeping the total

BS Transmit Power	23 dBm	UE Noise Figure	9 dB
Noise PSD	-174 dBm/Hz	Path Loss model	InH-Office [39]
Bandwidth	20 MHz	Fading model	Block fading

TABLE V: Federated Pilot GAN simulation parameters

number of training iterations equal to the centralized Pilot GAN implementation. We consider $n_d = 5$ and $n_d = 20$ which in turn implies $l = 100$ and $l = 25$ respectively. Note that $n_d = 20$ also involves performing 4 generative updates per training iteration instead of 1, to have a fair comparison.

The smoothed plot of NMSE vs rounds is shown in Fig. 9. From the performance of "FedGAN" with $n_d = 5$, as contrasted with CDL-A in Fig. 4a, we can ascertain that simply moving to a federated implementation did not have any detrimental impact on the NMSE achieved. However, noise in the LS channel estimates worsens the NMSE by ~ 2 dB. Furthermore, lowering the BS-UE communication frequency by a factor of 4 worsens the NMSE by ~ 0.8 dB. Such a trade-off between performance and BS-UE communication frequency as well as between performance and noise in the training data are to be expected in a federated setting [45].

Applications of Federated Pilot GAN: The applicability of FedPilotGAN is not restricted to building a generative channel estimator in an OTA fashion as is the focus in this paper. Here we describe a federated design of a (clean) data generator from noisy data realizations. Consider a BS (server) that requires a large dataset of channels (images) for some task. The BS (server) does not possess the data itself, however it is connected to a set of UEs (clients) that can capture noisy[10] copies of channel (image) realizations [46] [47]. From a practical perspective, a UE cannot store large amounts of data, cannot perform computationally intensive training or tolerate the uplink overhead of communicating "denoised" channels (images). Using the FedPilotGAN framework, we can train a generative model at the BS, without being subject to any of the aforementioned disadvantages, that can produce channel (image) samples from the underlying distribution. In addition, having a generator at the BS eliminates the need for storing large amounts of data at the BS as well i.e. we can simply generate batches of the required size on the fly by sampling from the generator.

---

[10]Obtaining multiple high resolution images can often be expensive and impractical for many sensing and tomography problems such as MRI and astronomical imaging [31] [43].
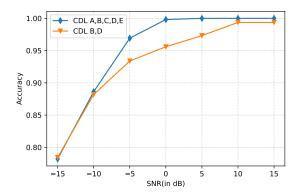
Fig. 10. Accuracy of LOS Predictor on CDL A-E. Using a subset of channel models for training minimally impacts accuracy.

D. PCGAN with LOS Predictor

1) LOS Predictor: We train the LOS Predictor $\mathcal{L}(.; \theta_L)$ for 20,000 iterations using a batch size of 200 and Adam [41] optimizer with $\gamma = 0.0003$. The accuracy of χ as given by (12) as a function of SNR has been plotted in Fig. 10. Two curves have been plotted - the upper curve is when $\mathcal{L}(.; \theta_L)$ is trained and tested on all CDL channel models, and the lower curve is when $\mathcal{L}(.; \theta_L)$ is trained on CDL-B and CDL-D only and tested on CDL A-E. The relatively minimal loss in accuracy supports supervised training of an LOS Predictor. We can simply train it using channel realizations from simulation tools available offline and test it with pilot measurements that need not be from the same channel distribution.

2) PCGAN: We now use the trained $\mathcal{L}(.; \theta_L)$ to compute χ from the LS channel estimate $\mathbf{H}_{v,LS}$ of the received pilot measurements $\underline{\mathbf{y}}[1 : K]$, and subsequently utilize $\{\mathbf{H}_{v,LS}\}$ to design a conditional generator by training a PCGAN ($\mathbb{1}_{GP} = 1$) using Algorithm 5 for 100,000 training iterations. The NMSE vs SNR on the best generative model for training SNR of 20 and 30 dB has been depicted in Fig. 11 for each of CDL A-E. It should be noted that we utilize $\mathcal{L}(.; \theta_L)$ only for *training* PCGAN. NMSE computation during *inference* bypasses the LOS Predictor by solving (13) in place of (5).

The NMSE on the LOS channel models degrades by ~ 2 dB for training SNR 30 dB and ~ 3.5 dB for training SNR 20 dB compared to CWGAN. Meanwhile, the NMSE on the NLOS channel models CDL-B and CDL-C is practically unaffected compared to CWGAN, and more interestingly, a better model has been learnt at the lower 20 dB training SNR for CDL-B. PCGAN based GCE continues to outperform OMP for both training SNR, while outperforming
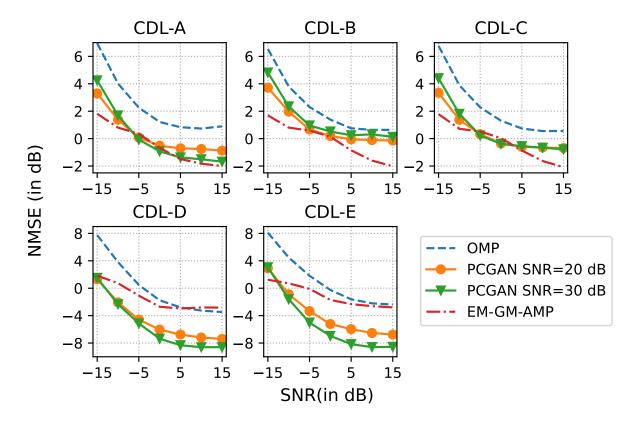
Fig. 11. NMSE vs SNR for PCGAN at *training* SNR of 20 and 30 dB. For reference, OMP and EM-GM-AMP baselines have also been plotted. PCGAN outperforms on LOS models, and achieves competitive results on NLOS models as compared to EM-GM-AMP, while consistently outperforming OMP.

EM-GM-AMP for the LOS channel models and being similar for CDL-A.

E. Complexity Analysis

In our prior work [23], we analyzed the computational complexity of GCE and contrasted it with OMP and EM-GM-AMP. Note that the complexity of both a forward and backward pass through a CNN are $\mathcal{O}(N_t N_r)$ – backward propagation takes roughly twice the time of forward propagation for a CNN [48] and the complexity of a forward pass through **G** or **D** is $\mathcal{O}(N_t N_r)$ [23]. Given that each NN based operation in Algorithm 5 can be expressed as either a forward or backward pass through a CNN, and the remaining are matrix additions, the big-$\mathcal{O}$ complexity of one iteration of PCGAN, given the dataset $\{\mathbf{H}_{v,LS}\}$, is $\mathcal{O}(N_t N_r)$. However, such an analysis has its limitations – i) it does not account for the batch size m, which is practically constrained by the parallel compute capacity of a GPU and ii) the big-$\mathcal{O}$ complexity simply upper bounds a forward pass by $\mathcal{O}(N_t N_r)$, while the actual complexity may be significantly lower.

VII. Conclusions & Future Directions

In this paper, we have presented a novel OTA formulation for training a GAN to learn the MIMO channel distribution for a range of LOS and NLOS channel models in accordance with the 3GPP standard [39]. We designed a LOS Predictor to determine whether the received pilot measurements originated from a LOS or a NLOS channel, and used the conditional output as input to a generative model, while the critic in the GAN was trained to distinguish between real and generated LS channel estimates. Subsequently, we utilized the trained generative model to perform generative channel estimation in the beamspace domain from compressive pilot measurements. While the individually trained WGANs outperformed both CS baselines – OMP and EM-GM-AMP – PCGAN was able to outperform OMP while outperforming EM-GM-AMP on LOS channel models and achieving competitive results on NLOS channels.

As part of future work, channel estimation using generative networks could potentially be performed in the presence of transmit non-linearities as well using the robust median-of-means estimator designed in [43]. There are also several research directions one could pursue to improve the practicality and applicability of the GAN training framework designed in this paper. In order to remove the assumption of block fading, we would need to train recurrent GANs [49] that produce a sequence of temporally correlated channel realizations. As explained in Section VI-B, the LOS predictor does not provide a sufficiently accurate indicator of the estimated degree of sparsity in the channel being reconstructed. Hence, we could instead train 3 GANs – for low, medium and high levels of beamspace sparsity – and then learn a classifier that will indicate which generative model to utilize [50]. Furthermore, we have not considered the degradation owing to quantization of the received signal, and incorporating CS based techniques for channel estimation from quantized pilot measurements [51] in combination with generative models [52] is an exciting future direction. Finally, one could look into combining GCE and GAN training to remove the requirement of full-rank pilot measurements for training Pilot GAN [53].

VIII. Acknowledgements

REFERENCES

[1] T. S. Rappaport, Y. Xing, O. Kanhere, S. Ju, A. Madanayake, S. Mandal, A. Alkhateeb, and G. C. Trichopoulos, "Wireless communications and applications above 100 GHz: Opportunities and challenges for 6G and beyond," *IEEE Access*, vol. 7, pp. 78 729–78 757, Jun. 2019.

[2] H. Elayan, O. Amin, R. M. Shubair, and M.-S. Alouini, "Terahertz communication: The opportunities of wireless technology beyond 5G," in *IEEE Intl. Conf. on Advanced Comm. Technologies and Networking (CommNet)*, Apr. 2018, pp. 1–5.

[3] E. Björnson, J. Hoydis, L. Sanguinetti *et al.*, *"Massive MIMO networks: Spectral, energy, and hardware efficiency"*. Foundations and Trends in Signal Processing, Now Publishers, Inc., Nov. 2017.

[4] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," in *Proc. ICLR*, Nov. 2015.

[5] A. Bora, A. Jalal, E. Price, and A. G. Dimakis, "Compressed sensing using generative models," in *Intl. Conf. on Machine Learning (ICML)*, Aug. 2017, pp. 537–546.

[6] G. Ongie, A. Jalal, C. A. Metzler, R. G. Baraniuk, A. G. Dimakis, and R. Willett, "Deep learning techniques for inverse problems in imaging," *IEEE Journal on Sel. Areas in Information Theory*, vol. 1, no. 1, pp. 39–56, May 2020.

[7] P. A. Eliasi, S. Rangan, and T. S. Rappaport, "Low-rank spatial channel estimation for millimeter wave cellular systems," *IEEE Trans. on Wireless Communications*, vol. 16, no. 5, pp. 2748–2759, Apr. 2017.

[8] W. U. Bajwa, J. Haupt, A. M. Sayeed, and R. Nowak, "Compressed channel sensing: A new approach to estimating sparse multipath channels," *Proc. IEEE*, vol. 98, no. 6, pp. 1058–1076, Jun. 2010.

[9] W. U. Bajwa, A. Sayeed, and R. Nowak, "Sparse multipath channels: Modeling and estimation," in *IEEE Digital Signal Processing and Education Workshop*, Jan. 2009, pp. 320–325.

[10] A. Alkhateeb, O. El Ayach, G. Leus, and R. W. Heath, "Channel estimation and hybrid precoding for millimeter wave cellular systems," *IEEE J. Sel. Topics Sig. Process.*, vol. 8, no. 5, pp. 831–846, Oct. 2014.

[11] R. Méndez-Rial, C. Rusu, N. González-Prelcic, A. Alkhateeb, and R. W. Heath, "Hybrid MIMO architectures for millimeter wave communications: Phase shifters or switches?" *IEEE Access*, vol. 4, pp. 247–267, Jan. 2016.

[12] S. Rangan, "Generalized approximate message passing for estimation with random linear mixing," in *IEEE Intl. Symposium on Information Theory Proceedings*, Jul. 2011, pp. 2168–2172.

[13] J. P. Vila and P. Schniter, "Expectation-maximization Gaussian-mixture approximate message passing," *IEEE Trans. on Signal Processing*, vol. 61, no. 19, pp. 4658–4672, Jul. 2013.

[14] S. Rangan, P. Schniter, and A. K. Fletcher, "Vector approximate message passing," *IEEE Trans. on Info. Theory*, vol. 65, no. 10, pp. 6664–6684, May 2019.

[15] C.-K. Wen, S. Jin, K.-K. Wong, J.-C. Chen, and P. Ting, "Channel estimation for massive MIMO using Gaussian-mixture Bayesian learning," *IEEE Trans. on Wireless Communications*, vol. 14, no. 3, pp. 1356–1368, Oct. 2014.

[16] P. Sun, Z. Wang, and P. Schniter, "Joint channel-estimation and equalization of single-carrier systems via bilinear AMP," *IEEE Trans. on Signal Processing*, vol. 66, no. 10, pp. 2772–2785, Mar. 2018.

[17] M. F. Duarte and Y. C. Eldar, "Structured compressed sensing: From theory to applications," *IEEE Trans. on Signal Processing*, vol. 59, no. 9, pp. 4053–4085, Jul. 2011.

[18] Y. Yang, F. Gao, X. Ma, and S. Zhang, "Deep learning-based channel estimation for doubly selective fading channels," *IEEE Access*, vol. 7, pp. 36 579–36 589, Mar. 2019.

[19] H. He, C.-K. Wen, S. Jin, and G. Y. Li, "Deep learning-based channel estimation for beamspace mmWave massive MIMO systems," *IEEE Wireless Communications Letters*, vol. 7, no. 5, pp. 852–855, Oct. 2018.

[20] X. Ru, L. Wei, and Y. Xu, "Model-driven channel estimation for OFDM systems based on image super-resolution network," in *Intl. Conf. on Signal and Image Processing (ICSIP)*, Oct. 2020, pp. 804–808.

[21] P. Dong, H. Zhang, G. Y. Li, I. S. Gaspar, and N. NaderiAlizadeh, "Deep CNN-based channel estimation for mmWave Massive MIMO systems," *IEEE J. Sel. Topics Sig. Process.*, vol. 13, no. 5, pp. 989–1000, Jul. 2019.

[22] C.-J. Chun, J.-M. Kang, and I.-M. Kim, "Deep learning-based channel estimation for massive MIMO systems," *IEEE Wireless Communications Letters*, vol. 8, no. 4, pp. 1228–1231, Apr. 2019.

[23] E. Balevi, A. Doshi, A. Jalal, A. Dimakis, and J. G. Andrews, "High dimensional channel estimation using deep generative networks," *IEEE Journal on Sel. Areas in Communications*, vol. 39, no. 1, pp. 18–30, Nov. 2020.

[24] A. Doshi, E. Balevi, and J. G. Andrews, "Compressed representation of high dimensional channels using deep generative networks," in *Proc., IEEE Signal Proc. Adv. in Wireless Comm. (SPAWC)*, May 2020, pp. 1–5.

[25] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Adv. in Neural Info. Process. Systems*, Dec. 2014, pp. 2672–2680.

[26] A. Jalal, M. Arvinte, G. Daras, E. Price, A. G. Dimakis, and J. Tamir, "Robust compressed sensing MRI with deep generative priors," *Adv. in Neural Info. Process. Systems*, vol. 34, Dec. 2021.

[27] M. Arvinte and J. I. Tamir, "Deep Diffusion Models for Robust Channel Estimation," *arXiv preprint arXiv:2111.08177*, Nov. 2021.

[28] E. Balevi and J. G. Andrews, "Wideband channel estimation with a generative adversarial network," *IEEE Trans. on Wireless Communications*, vol. 20, no. 5, pp. 3049–3060, May 2021.

[29] T. Orekondy, A. Behboodi, and J. B. Soriaga, "MIMO-GAN: Generative MIMO Channel Modeling," *To appear in IEEE Intl. Conf. on Communications*, Mar. 2022.

[30] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of Wasserstein GANs," *Adv. in Neural Info. Process. Systems*, vol. 30, Dec. 2017.

[31] A. Bora, E. Price, and A. G. Dimakis, "AmbientGAN: Generative models from lossy measurements," in *Proc. ICLR*, Feb. 2018.

[32] K. Venugopal, A. Alkhateeb, N. G. Prelcic, and R. W. Heath, "Channel estimation for hybrid architecture-based wideband millimeter wave systems," *IEEE Journal on Sel. Areas in Communications*, vol. 35, no. 9, pp. 1996–2009, Jun. 2017.

[33] D. L. Donoho, M. Elad, and V. N. Temlyakov, "Stable recovery of sparse overcomplete representations in the presence of noise," *IEEE Trans. on Info. Theory*, vol. 52, no. 1, pp. 6–18, Dec. 2005.

[34] Y. Zhang, A. Doshi, R. Liston, W. Tan, X. Zhu, J. G. Andrews, and R. W. Heath, "DeepWiPHY: Deep learning-based receiver design and dataset for IEEE 802.11 ax systems," *IEEE Trans. on Wireless Communications*, vol. 20, no. 3, pp. 1596–1611, Nov. 2020.

[35] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *Intl. Conf. on Machine Learning (ICML)*, Dec. 2017, pp. 214–223.

[36] A. Srivastava, L. Valkov, C. Russell, M. U. Gutmann, and C. Sutton, "Veegan: Reducing mode collapse in GANs using implicit variational learning," in *Adv. in Neural Info. Process. Systems*, Dec. 2017, pp. 3308–3318.

[37] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *arXiv preprint arXiv:1411.1784*, Nov. 2014.

[38] S. Augenstein, H. B. McMahan, D. Ramage, S. Ramaswamy, P. Kairouz, M. Chen, R. Mathews *et al.*, "Generative models for effective ML on private, decentralized datasets," in *Proc. ICLR*, Nov. 2019.

[39] 3GPP, "Study on channel model for frequencies from 0.5 to 100 GHz," 3rd Generation Partnership Project (3GPP), Technical Report (TR) 38.901, Jan. 2020, version 16.1.0.

[40] A. Paszke *et al.*, "Pytorch: An imperative style, high-performance deep learning library," in *Adv. in Neural Info. Process. Systems*, 2019, vol. 32, pp. 8024–8035.

[41] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. ICLR*, Dec. 2014.

[42] M. Giordani, M. Mezzavilla, and M. Zorzi, "Initial access in 5G mmWave cellular networks," *IEEE Communications Magazine*, vol. 54, no. 11, pp. 40–47, 2016.

[43] A. Jalal, L. Liu, A. G. Dimakis, and C. Caramanis, "Robust compressed sensing using generative models," *Adv. in Neural Info. Process. Systems*, vol. 33, pp. 713–727, Dec. 2020.

[44] 3GPP, "NR; Physical layer procedures for data," 3rd Generation Partnership Project (3GPP), Technical Specification (TS) 38.214, Jan. 2021, version 16.4.0.

[45] K. Bonawitz *et al.*, "Towards federated learning at scale: System design," *Proceedings of Machine Learning and Systems*, vol. 1, pp. 374–388, Apr. 2019.

[46] M. Li, D. G. Andersen, J. W. Park, A. J. Smola, A. Ahmed, V. Josifovski, J. Long, E. J. Shekita, and B.-Y. Su, "Scaling distributed machine learning with the parameter server," in *OSDI*, 2014, pp. 583–598.

[47] C. Hardy, E. Le Merrer, and B. Sericola, "Md-gan: Multi-discriminator generative adversarial networks for distributed datasets," in *IEEE Intl. Parallel and Distributed Processing Symposium (IPDPS)*, May 2019, pp. 866–877.

[48] K. He and J. Sun, "Convolutional neural networks at constrained time cost," *in Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, pp. 5353–5360, Jun. 2015.

[49] C. Esteban, S. L. Hyland, and G. Rätsch, "Real-valued (medical) time series generation with recurrent conditional gans," *arXiv preprint arXiv:1706.02633*, Jun. 2017.

[50] D. K. Park, S. Yoo, H. Bahng, J. Choo, and N. Park, "MEGAN: Mixture of Experts of Generative Adversarial Networks for Multimodal Image Generation," in *Proc. Intl. Joint Conference on Artificial Intelligence*, Jul. 2018, pp. 878–884.

[51] N. J. Myers, K. N. Tran, and R. W. Heath, "Low-rank mmWave MIMO channel estimation in one-bit receivers," in *Proc., IEEE Intl. Conf. on Acoustics, Speech, and Sig. Proc. (ICASSP)*, May 2020, pp. 5005–5009.

[52] S. Qiu, X. Wei, and Z. Qiu, "Robust One-Bit Recovery via ReLU Generative Networks: Improved Statistical Rates and Global Landscape Analysis," *in NeurIPS Deep Inverse Workshop*, Dec. 2019.

[53] M. Kabkab, P. Samangouei, and R. Chellappa, "Task-aware compressed sensing with generative adversarial networks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, Apr. 2018.

[54] K. Li and D.-K. Kang, "Enhanced generative adversarial networks with restart learning rate in discriminator," *Applied Sciences*, vol. 12, no. 3, p. 1191, Jan. 2022.

APPENDIX

A. Proof of Section IV-C

Consider noiseless pilot measurements $\underline{\mathbf{y}} = \mathbf{A}_{\mathrm{sp}}\underline{\mathbf{H}}_{\mathbf{v}}$ such that $N_p N_s < N_t N_r$. As explained in Section II, this implies that we cannot recover a unique $\underline{\mathbf{H}}_{\mathbf{v}}$ given only $\underline{\mathbf{y}}$. Let us utilize the LQ decomposition of $\mathbf{A}_{\mathrm{sp}}$ to represent it as $\mathbf{A}_{\mathrm{sp}} = (\mathbf{L}\ \mathbf{0})\mathbf{Q}$ where $\mathbf{L}$ is a $N_p N_s \times N_p N_s$ lower triangular matrix and $\mathbf{Q}$ is an $N_t N_r \times N_t N_r$ unitary matrix. Hence we have

$$\underline{\mathbf{y}} = (\mathbf{L}\ \mathbf{0})\mathbf{Q}\underline{\mathbf{H}}_{\mathbf{v}} = (\mathbf{L}\ \mathbf{0})\begin{bmatrix} \mathbf{h}_{Q,1} \\ \mathbf{h}_{Q,2} \end{bmatrix}. \tag{15}$$
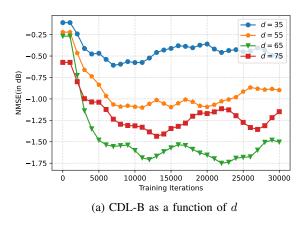
where $\mathbf{h}_{Q,1}$ contains the first $N_p N_s$ entries of $\mathbf{Q}\underline{\mathbf{H}}_{\mathbf{v}}$ and $\mathbf{h}_{Q,2}$ contains the remaining. Denote $\underline{\mathbf{H}}_{\mathbf{v},Q} = \mathbf{Q}\underline{\mathbf{H}}_{\mathbf{v}}$. Since $\mathbf{Q}$ is invertible, there is a one-to-one mapping between the $p_{\underline{\mathbf{H}}_{\mathbf{v},Q}}(.)$ and $p_{\underline{\mathbf{H}}_{\mathbf{v}}}(.)$. However, $\underline{\mathbf{y}}$ is only a function of $\mathbf{h}_{Q,1}$. In fact, since $\mathbf{L}$ is lower triangular, we can compute $\mathbf{h}_{Q,1}$ from $\underline{\mathbf{y}}$ by elementary row operations. Hence all pdfs of the form
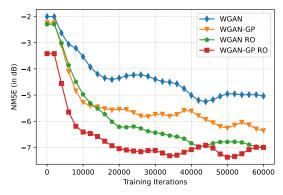
$$p_{\underline{\mathbf{H}}_{\mathbf{v},Q}}(.) = \frac{p_{\underline{\mathbf{y}}}(.)}{|\det(J_{\underline{\mathbf{y}}}(\mathbf{h}_{Q,1}))|} p_{\mathbf{h}_{Q,2}}(.), \tag{16}$$

where $J_{\underline{\mathbf{y}}}(\mathbf{h}_{Q,1})$ is the Jacobian of $\underline{\mathbf{y}}$ evaluated at $\mathbf{h}_{Q,1}$, map to the same $p_{\underline{\mathbf{y}}}(.)$. Note that (16) may not even include the ground truth $p_{\underline{\mathbf{H}}_{\mathbf{v},Q}}$ if $p_{\underline{\mathbf{H}}_{\mathbf{v},Q}} \neq p_{\mathbf{h}_{Q,1}}(.)p_{\mathbf{h}_{Q,2}}(.)$. Hence, we cannot define a one-to-one mapping between $p_{\underline{\mathbf{y}}}$ and $p_{\underline{\mathbf{H}}_{\mathbf{v},Q}}$, which implies the absence of the same between $p_{\underline{\mathbf{y}}}$ and $p_{\underline{\mathbf{H}}_{\mathbf{v}}}$, proving Ambient GAN [31] and by extension – Pilot GAN – cannot be trained with compressive measurements.

B. Optimal value of d for $\mathbf{z} \in \mathbb{R}^d$

Given that 3GPP TR 38.901 [39] defines CDL-B to have the largest number of rays/clusters among all CDL models with only a weak LOS component, and consequently has the highest NMSE, we determine the optimal value of d by training a WGAN-GP on CDL-B. The smoothed plot of NMSE vs training iterations is shown in Fig. 12a for varying values of d. The NMSE decreases going from $d = 55$ to $d = 65$, but increases on going to $d = 75$. Hence we have chosen $d = 65$ as the optimal latent dimension.

(a) CDL-B as a function of d

(b) Impact of resetting critic optimizer (RO) on CDL-A

Fig. 12. NMSE vs Training Iterations for WGAN

C. Effect of resetting critic optimizer

Training instability in GANs can also be attributed to early critic convergence, which reduces the incentive for the generator to improve its performance i.e. "defeat the adversary" [54]. Hence we investigated the impact of resetting the critic optimizer after every training iteration i.e. after n_d critic and 1 generator iterations in Algorithm 1. The results shown in Fig. 12b indicate that optimizer resetting provides a ~ 1 dB reduction in NMSE for CDL-A for WGAN-GP ($\mathbb{1}_{\mathrm{GP}} = 1$) and a ~ 2 dB reduction for WGAN ($\mathbb{1}_{\mathrm{GP}} = 0$).