Robust Channel Estimation for OFDM Systems with Generative Modeling

Berkay Guler **Neal Sharma** **Soyeon Kwon**

Abstract

Channel estimation for Orthogonal Frequency Division Multiplexing (OFDM) systems is widely addressed with training supervised deep learning models that learn a mapping from the channel at pilot positions to complete channels. While the parameterized mapping approach outperforms traditional statistical channel estimators, its performance is highly sensitive to varying channel conditions. On the other hand, estimating channels by sampling from a learned distribution might provide inherently more robust estimates, capturing the underlying uncertainty of wireless channels. This distribution-based approach allows the estimator to generalize better across different channel conditions by learning the full statistical relationship between pilot measurements and channel responses, rather than being constrained to a deterministic mapping. In this paper, we explore conditional generative adversarial networks (GANs), variational autoencoders (VAEs), and denoising diffusion probabilistic models (DDPMs) for channel estimation on a dataset generated from extensive wireless system simulations.

1 Introduction

Orthogonal Frequency Division Multiplexing (OFDM) is widely used for its resilience to multipath fading and high spectral efficiency. Accurate channel estimation (CE) is vital for optimal OFDM performance, particularly in dynamic wireless environments [1]. High noise degrades OFDM signal quality, while Doppler shifts and frequency offsets in high-mobility scenarios disrupt subcarrier orthogonality. Delay spread from multipath propagation introduces inter-symbol interference, further affecting signal integrity. These challenges limit the effectiveness of conventional channel estimators, complicating equalization and data recovery. While increasing pilot density helps address these issues, it reduces throughput and spectral efficiency, making advanced adaptive methods essential for handling noisy and high-mobility environments [2, 3].

Pilot-assisted channel estimation (PA-CE) is the predominant CE method for OFDM and relies on placing known symbols at predefined frequencies and time indices. Two common PA-CE methods are the LS estimator[4] and Linear Minimum Mean Square Error (LMMSE) estimators [5]. The LS estimator offers simplicity but does not consider channel statistics, while LMMSE provides better adaptability using second-order statistics at the cost of higher complexity.

Deep learning-based channel estimation (DL-CE) has emerged as a promising alternative to traditional methods, gaining popularity for its superior performance [6, 7]. The typical DL-CE framework aims to learn a parameterized mapping from the LS-estimated channel at pilot positions to the complete channel. Although sophisticated deep learning architectures have been developed as mappers, they suffer from poor generalization. In this paper, we explore a generative channel estimation (GEN-CE) framework to achieve higher robustness in OFDM channel estimation. The fundamental advantage of our method is that it learns a conditional probability distribution rather than learning a deterministic mapping, which enables uncertainty modeling.

Preprint. Under review.

We compare our framework with traditional channel estimators and state-of-the-art DL-CE methods on an in-house dataset of OFDM channel samples collected from a realistic wireless system simulation.

2 Task Description

In the PA-CE framework, the receiver utilizes the pilot symbols, whose values and locations in the OFDM grid are predefined, to predict the entire channel matrix. As illustrated in Fig. 1, this is a 2-D interpolation problem where the aim is to recover the entire grid as accurately as possible. Typically, the initial channel estimates for pilot positions are obtained by the LS estimator. In this work, we aim

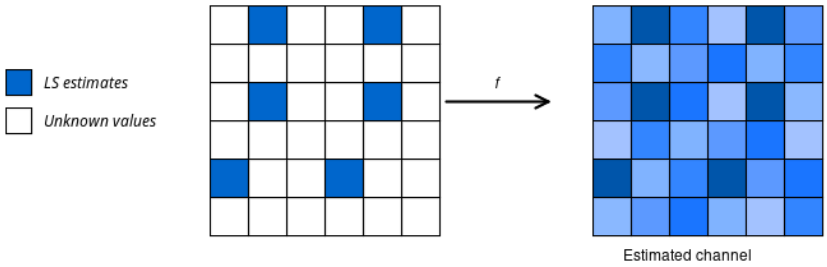


Figure 1: Channel estimation as a 2-D interpolation problem where x and y axes represent OFDM time indices and subcarrier indices, respectively.

to learn the conditional distribution $P(\mathbf{H} \mid \hat{\mathbf{H}}_p^{\text{LS}})$ where $\mathbf{H}$ is the complete channel and $\hat{\mathbf{H}}_p^{\text{LS}}$ is the estimated channel at pilot positions.

2.1 Dataset Generation and Preprocessing

We generated OFDM channels by simulating a single-input single-output (SISO) wireless system in MATLAB. The parameters of the simulated system were adopted from a valid configuration in the latest 5G-NR specification [8]. We used the TDL-A channel model described in [9]. The system employed a 15 kHz subcarrier spacing, 10 resource blocks, and 12 subcarriers per block, resulting in $N_f = 120$ subcarriers and $N_t = 14$ OFDM symbols per frame, with a total bandwidth of 1.8 MHz. QPSK-modulated signals were transmitted over the TDL-A channel and sampled at a rate of 3.84 MHz.

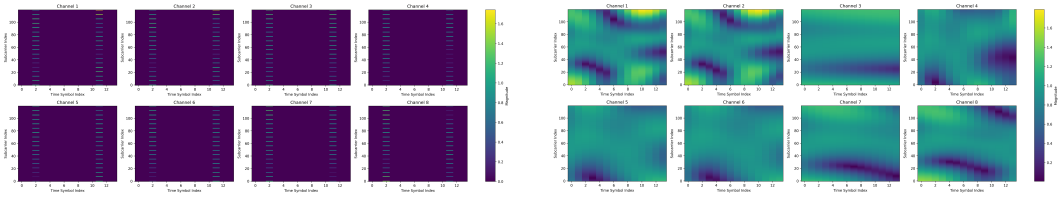
We inserted pilot symbols every seventh subcarrier position along the third and twelfth time indices, resulting in 36 pilot symbols per frame. We employed the LS algorithm to estimate the channel at pilot positions. Our dataset $\mathcal{D} = \{(\hat{\mathbf{H}}_{p_i}^{\text{LS}}, \mathbf{H}_i)\}_{i=1}^{K}$ consists of pairs of LS-estimated pilot channel matrices and ground-truth channel matrices, where $\hat{\mathbf{H}}_{p_i}^{\text{LS}} \in \mathbb{C}^{18\times 2}$ and $\mathbf{H}_i \in \mathbb{C}^{120\times 14}$.

2.1.1 Training and Validation Sets

We generated 100,000 and 10,000 OFDM channel realizations for the training and validation sets, respectively. For each channel realization, we selected an SNR value from $\{0, 5, \ldots, 25\}$ dB, a maximum Doppler shift value from $\{50, 100, \ldots, 1000\}$ Hz, and a delay spread value from $\{25, 50, \ldots, 300\}$ ns. Fig. 2 displays random samples from $\mathcal{D}$.

2.1.2 Test Sets

We compare the performance of our models on dynamic delay spread (DS) and noisy dynamic DS test sets. The dynamic DS test set contains 2000 channel realizations for each DS value $s \in \{50, 100, \ldots, 300\}$ nanoseconds (ns), whereas the noisy dynamic DS test set contains 2000 channels for each nominal DS value $s_i \in \{50, 100, \ldots, 300\}$ ns, where each channel experiences an actual DS of $(s_i + \epsilon)$ ns with $\epsilon \sim \mathcal{N}(0, \sigma^2)$ where we set $\sigma = 50$ ns.

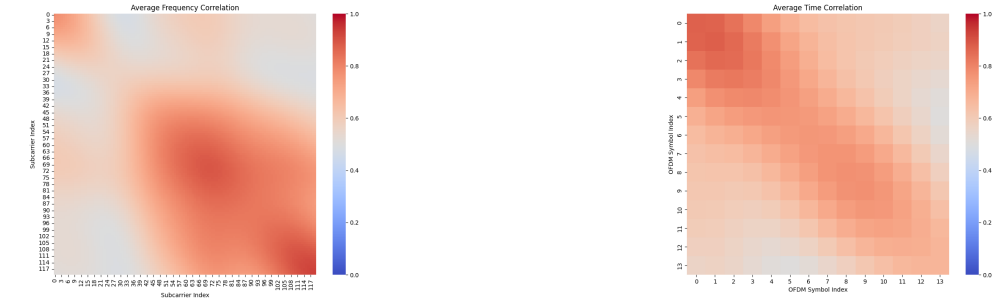


(a) LS estimate at pilot positions. (b) ground-truth channels.

Figure 2: Dataset Illustration.

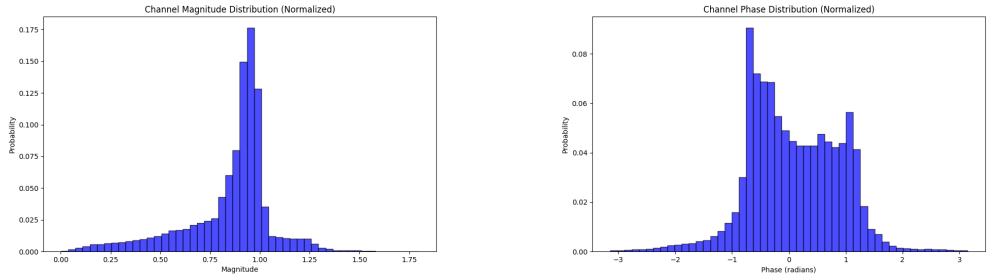
3 Data Exploration

We calculate correlation across time to show the strong dependency between neighbouring time indices. As illustrated in Fig. 3b, the correlation strength decays gradually and symmetrically as time indices get further apart, with the strongest correlation (shown in dark red) along the diagonal and progressively weaker correlations (transitioning to lighter shades) towards the edges, suggesting a temporal coherence of approximately 4–5 symbol periods. Correlation across frequency, on the other hand, has a more interesting structure, as seen in Fig. 3a, where we observe a stronger correlation among subcarrier indices at higher frequencies. This asymmetric correlation pattern suggests frequency-selective fading characteristics in the channel. In Fig. 4, we display normalized histograms of the distribution of values in channel matrices, revealing distinct patterns: the magnitude distribution exhibits a right-skewed bell shape centered around 0.75 with a long tail extending to 1.5, while the phase distribution shows a relatively uniform spread across the range $[-\pi, \pi]$ with slightly higher density near zero, characteristic of complex wireless channel responses in multipath environments.



(a) Correlation matrix of subcarrier indices. (b) Correlation matrix of time indices.

Figure 3: Correlation analysis in time and frequency.



(a) Distribution of magnitudes of channel elements. (b) Distribution of phase of channel elements.

Figure 4: Density of channel elements in magnitude and angle.

4 Model Exploration and Experiments

4.1 Conditional GAN

Model Architecture and Loss Function We reimplemented a popular conditional generative adversarial network (GAN) [10], where the conditional GAN objective is:

$$\mathcal{L}_{\text{cGAN}}(G_\theta, D_\phi) = \mathbb{E}_{(\hat{\mathbf{H}}^{\text{LS}}_{p_i}, \mathbf{H}_i)\sim p_{\text{data}}}[\log D_\phi(\hat{\mathbf{H}}^{\text{LS}}_{p_i}, \mathbf{H}_i)] + \mathbb{E}_{\hat{\mathbf{H}}^{\text{LS}}_{p_i}\sim p_{\text{data}}}[\log(1 - D_\phi(\hat{\mathbf{H}}^{\text{LS}}_{p_i}, G_\theta(\hat{\mathbf{H}}^{\text{LS}}_{p_i}, z)))]$$

where we modeled G_θ with a U-Net architecture containing three encoding-decoding blocks with dual 3×3 convolutions, instance normalization, ReLU, and dropout. The encoder path expands from 2 to 256 channels, while the decoder uses bilinear upsampling with skip connections, concluding with a 1×1 convolution to output 2 channels. To model z, we apply dropout and instance normalization at test time [10]. D_ϕ implements a PatchGAN with four convolutional layers (4×4 kernel, padding 1), expanding from 16 to 64 filters, each followed by LeakyReLU and instance normalization except the final layer, which outputs patch-wise predictions using asymmetric striding for rectangular inputs.

Following the original paper, we further guide the generator by forcing its outputs to be close to the ground-truth channel matrices $\mathbf{H}_i$ in the L1 sense [10].

$$\mathcal{L}_{L1}(G_\theta) = \mathbb{E}_{(\hat{\mathbf{H}}^{\text{LS}}_{p_i}, \mathbf{H}_i)\sim p_{\text{data}}}[\|\mathbf{H}_i - G_\theta(\hat{\mathbf{H}}^{\text{LS}}_{p_i}, z)\|_1]$$

The combined objective for the generator becomes:

$$G^*_\theta = \arg\min_{G_\theta}\max_{D_\phi} \mathcal{L}_{\text{cGAN}}(G_\theta, D_\phi) + \lambda_{\text{L1}}\mathcal{L}_{\text{L1}}(G_\theta)$$

where λ_{L1} is the coefficient of the L1 loss.

Training and Parameter Selection We used a relatively small U-Net architecture to quickly iterate and optimize the parameters of the GAN and the training setup. With the initial rounds of training, we experimented with different learning rates and λ_{L1} values. We were able to obtain competitive loss values for the generator when we chose very large λ_{L1}. However, this turned our model into a regular neural network minimizing L1 loss. Hence, we picked $\lambda_{\text{L1}} = 5$ to keep the adversarial nature alive while also guiding of the generator independent of the discriminator.

We realized that our generator cannot learn to deceive the discriminator, so we scaled the loss for the discriminator with $d_l < 1$. We also started using smaller learning rates for the discriminator compared to the generator. Finally, we set $d_l = 0.1$ and used $lr_{\text{G}} = 0.0001$ and $lr_{\text{D}} = 0.00001$. We also employed checkpointing as a way of early stopping once the discriminator can no longer guide the generator and the generator loss starts to increase. We illustrate the generator and discriminator losses throughout a 100 epoch training in Fig. 5, where it is obvious that the generator fails to deceive the discriminator and the discriminator dominates the generator after a few epochs.

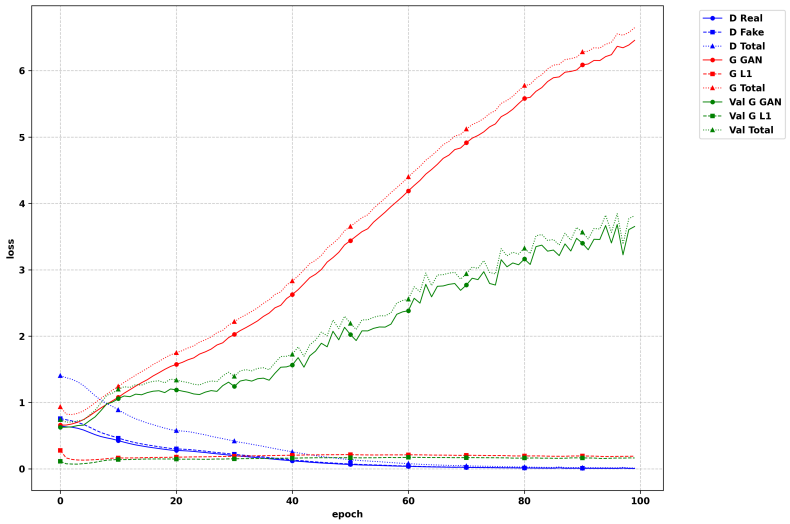


Figure 5: Training and Validation Losses of cGAN Training

Since the randomness in the output of generator came from the instance-normalization and dropout, we experimented with different dropout rates and chose an aggressive dropout rate of 0.5. We set the batch size to 256 and selected the model checkpoint with the lowest validation set error for final evaluation on the test sets.

4.2 VAE

Another model we tested is Variational Autoencoders (VAEs), which are probabilistic generative models that approximate a data distribution by mapping inputs to a latent space representation and then decoding samples back into the data space. In the context of channel estimation for OFDM systems, VAEs offer a framework to model the inherent uncertainty of wireless channels. Specifically, we designed the VAE to learn the conditional distribution $P(\mathbf{H} \mid \hat{\mathbf{H}}_p^{\text{LS}})$, where $\mathbf{H}$ represents the complete channel and $\hat{\mathbf{H}}_p^{\text{LS}}$ is the LS-estimated channel at pilot positions.

Model Architecture

The architecture of the VAE was designed based on the characteristics of the input OFDM channel matrices. The encoder processes the LS-estimated pilot, $\hat{\mathbf{H}}_p^{\text{LS}} \in \mathbb{R}^{18\times 2}$, where 18 pilot positions and 2 channels are flattened into a vector of size 72 ($18 \times 2 \times 2$).

The first layer that maps the input from $\mathbb{R}^{72}$ to $\mathbb{R}^{128}$ to expand the feature space and enable the model to learn relationships in the data, followed by a ReLU activation to introduce non-linearity. The second linear layer reduces the dimensionality from $\mathbb{R}^{128}$ to $\mathbb{R}^{64}$, again applying a ReLU activation, condensing the learned features to a more compact representation. Finally, the encoder generates the parameters of the latent distribution by producing the mean $\mu \in \mathbb{R}^{32}$ and the log-variance $\log \sigma^2 \in \mathbb{R}^{32}$ through two separate linear layers. The latent space size of 32 was chosen to capture variability in the pilot channel data while providing a compact latent representation.

To enable backpropagation through stochastic sampling, the reparameterization trick is employed. Latent samples z are drawn as:

$$z = \mu + \sigma \odot \epsilon, \quad \epsilon \sim \mathcal{N}(0, \mathbf{I}),$$

where $\sigma = \exp(0.5 \cdot \log \sigma^2)$. This approach allows the VAE to model the inherent variability and uncertainty in wireless channels effectively, making it particularly suited for the challenges of channel estimation.

The decoder reconstructs the full flattened channel matrix $\hat{\mathbf{H}} \in \mathbb{R}^{3360}$ ($2 \times 120 \times 14$) from the latent representation z. It starts with a linear layer that maps the latent space from $\mathbb{R}^{32}$ to $\mathbb{R}^{64}$, followed by a ReLU activation. A second linear layer further expands the dimensionality from $\mathbb{R}^{64}$ to $\mathbb{R}^{128}$, also followed by a ReLU activation. The final linear layer outputs $\mathbb{R}^{3360}$, which is reshaped into the original channel dimensions $[2, 120, 14]$, preserving the spatial structure of OFDM channels. This architecture ensures that the reconstructed channel estimates align with the original structure and characteristics of the complete OFDM channel.

Loss Function

The VAE optimizes the evidence lower bound (ELBO) for the channel estimation task. The ELBO combines reconstruction accuracy and regularization:

$$\mathcal{L}_{\text{VAE}} = \mathcal{L}_{\text{recon}} + \beta \mathcal{L}_{\text{KL}}, \tag{1}$$

where:

- $\mathcal{L}_{\text{recon}}$ is the reconstruction loss, measured as the mean squared error (MSE) between the reconstructed channel $\hat{\mathbf{H}}$ and the ground truth $\mathbf{H}$, ensuring that the decoder accurately interpolates the full channel matrix.
- $\mathcal{L}_{\text{KL}} = -0.5 \sum(1 + \log \sigma^2 - \mu^2 - \sigma^2)$ is the Kullback-Leibler divergence, regularizing the latent space to follow a standard normal distribution.

Training Process and Parameter Selection

The VAE was trained for 20 epochs using the Adam optimizer with a learning rate of 3×10^{-4}. Training batches consisted of 32 samples, with each input normalized to ensure consistent scale across the training data. The two LS-estimated pilot-based input channels ($[18, 2]$) were flattened into a vector of size 72 for compatibility with the encoder, while the decoder's output was reshaped back into the complete channel matrix dimensions $[2, 120, 14]$.

The choice of hyperparameters, including the KL divergence coefficient $\beta = 1.0$, was the result of iterative experimentation. Early trials with lower β values resulted in overly deterministic reconstructions that lacked diversity, while higher values overly prioritized regularization, degrading reconstruction accuracy. The final setting of $\beta = 1.0$ struck a balance, ensuring that the VAE captured latent uncertainty without sacrificing reconstruction quality, as shown in Figure 6. Similarly, the learning rate was adjusted after observing convergence stability and the smoothness of loss reduction, with 3×10^{-4} providing the best compromise between learning speed and stability.

These parameters were validated using a dedicated validation dataset, which included scenarios with varying Doppler shifts and SNRs. This ensured that the VAE generalized well across diverse channel conditions without overfitting to specific cases.

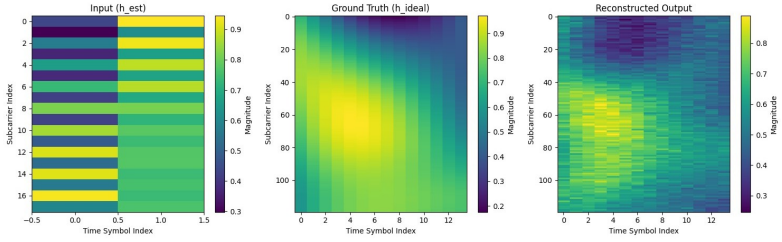


Figure 6: Sample VAE channel reconstructed output vs. ground truth

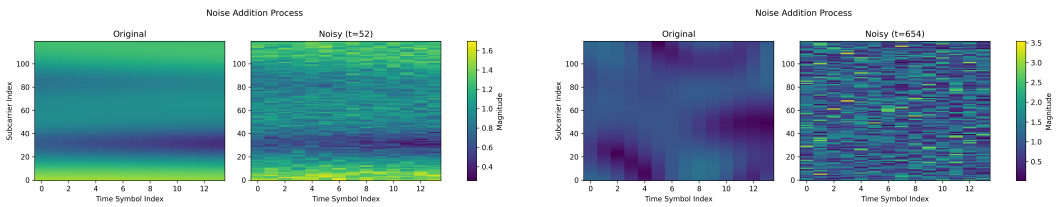
4.3 DDPM

One of the generative models we use is a denoising diffusion probabilistic model (DDPM), combining a conditional U-Net with cross-attention mechanisms to generate accurate channel estimates. DDPM gradually processes the ground-truth channels through a fixed Markov chain of T steps that progressively adds Gaussian noise, followed by learning a reverse process that restores the channel state by iteratively denoising. In this process, the pilot symbols serve as the conditional information to guide the denoising process [11].

The forward diffusion process is defined by a variance schedule $\beta_1, \ldots, \beta_t$ that gradually adds Gaussian noise to the ground-truth channel, $\mathbf{x}_0$, producing increasingly noisy variants $\mathbf{x}_1, \ldots, \mathbf{x}_t$. At each timestep t, the noisy data $\mathbf{x}_t$ is computed through a reparameterization trick that enables sampling at arbitrary timesteps [11]:

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t}\boldsymbol{\varepsilon},\ \boldsymbol{\varepsilon} \sim N(0, \mathbf{I})$$

where $\bar{\alpha}_t = \prod_{i=1}^{t}(1 - \beta_i)$ represents the cumulative product of noise scheduling terms $\alpha_i = 1 - \beta_i$. The noise addition process is in Fig. 7.



(a) Addition of noise at timestep 52.

(b) Addition of noise at timestep 654.

Figure 7: Addition of randomized timestep noise in the original channel estimation.

Our implementation utilizes a cosine noise schedule that provides smoother noise addition compared to linear or quadratic schedules:

$$\bar{\alpha}_t = \cos^2\left(\frac{t/T + s}{1+s} \cdot \frac{\pi}{2}\right)$$

$$\beta_t = \left[1 - \frac{\bar{\alpha}_t}{\bar{\alpha}_{t-1}}\right]_{0.0001}^{0.02}$$

where $s = 0.008$ is an offset parameter that prevents singularities at $t = 0$ [12].
The reverse process then learns to gradually denoise the data by estimating the noise component at each timestep [11]. Given a noised sample $\mathbf{x}_t$, the model predicts the noise $\boldsymbol{\varepsilon}_t$, allowing us to compute the denoised estimate $\hat{\mathbf{x}}_{t-1}$ shown in Fig. 8.

$$\hat{\mathbf{x}}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\varepsilon}_\theta(\mathbf{x}_t, t)\right) + \sigma_t \mathbf{z}$$

where σ_t represents the scheduled noise level and $\mathbf{z} \sim N(0, \mathbf{I})$ is an additional noise term added only when $t > 0$ to maintain the stochastic nature of the process [13]. θ represents the learnable parameters that predict the noise such as the conditional U-Net in this model.
Our DDPM model utilizes a conditional U-Net architecture with integrated cross-attention mecha-

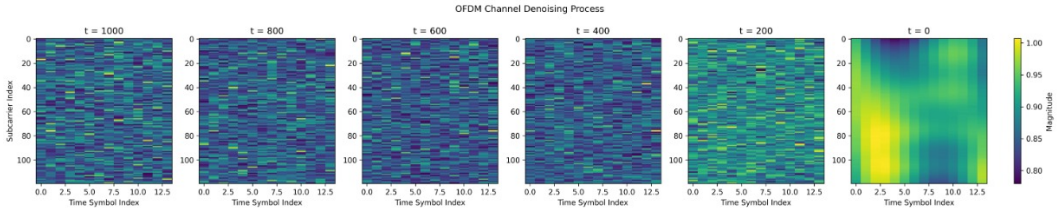


Figure 8: OFDM channel denoising visualization across different timesteps (t) from random noise (t=1000) to a clear channel estimate sample (t=0).

nism. The U-Net processes the noised channel data through a series of downsampling and upsampling operations, while maintaining skip connections to preserve information. At each scale, cross-attention modules enable the model to leverage pilot estimates through the attention mechanism [14, 15].

$$Attention(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = softmax(\mathbf{Q}\mathbf{K}^T/\sqrt{d})\mathbf{V}$$

where the queries $\mathbf{Q}$ are derived from the main channel features, while the keys $\mathbf{K}$ and values $\mathbf{V}$ are computed from the pilot signals through a dedicated pilot encoder, and d is the query dimension per attention head, where the input is split into multiple heads to enable parallel attention computation. This encoder consists of convolutional layers so that we can keep spatial structure, followed by self-attention mechanisms that capture pilot interactions [14].
During training, the model is optimized using MSE loss between the predicted and actual noise. The NMSE (Normalized Mean Square Error) in dB scale is also calculated for monitoring the training progress and evaluating the model's performance, but not for optimization. At inference time, the model performs T denoising steps, starting from pure Gaussian noise $\mathbf{x}_T \sim N(0, \mathbf{I})$ and iteratively applying the learned denoising process conditioned on the pilot symbols to arrive at the final channel estimate.

Experiments

For experiments, we implemented three iterations of DDPM, each with increasing complexity or change in approach. The first model was a basic implementation to understand DDPM and resolve dimension matching issues in the U-Net. It includes a simple conditioning where pilot estimates were projected and interpolated to match the output dimensions. The core architecture consisted of a lightweight U-Net with basic convolutional blocks in both encoder and decoder paths, and time embedding through a shallow neural network with activation. For the diffusion process, we used a linear noise scheduler with beta values from 0.0001 to 0.02. This baseline model achieved

suboptimal performance with a training MSE of 0.03 with its best hyperparameters.

After the intermediate iteration with deeper architecture and residual connections, we fundamentally changed the approach to preserve spatial relationships in channel estimation. The key change was implementing a dedicated pilot processing module and multi-head attention mechanisms throughout the network. The conditioning was redesigned to use multi-head attention at each level with assymmetric downsampling, allowing the model to better handle the rectangular input shape and utilize pilot information more effectively. We also modified the diffusion process by implementing cosine scheduler for noise, which provided smoother transitions between timesteps compared to the linear scheduler. Along with comprehensive and efficient validation strategies, this architecture achieved significantly better performance while maintaining the core DDPM framework and cosine beta scheduler.

Training Process and Parameter Selection

The training process runs for 20 epochs with 1000 noise timesteps, using a batch size of 64 and a U-Net architecture with 128 hidden dimensions. The model utilizes an AdamW optimizer with an initial learning rate of 3e-4, complemented by a cosine annealing scheduler that gradually reduces the learning rate to 1e-6 throughout training. To manage computational load, validation is performed every 10 epochs, with model checkpoints stored for subsequent testing. Performance metrics are calculated in dB to ensure consistent evaluation, with the training pipeline maintaining the five most recent checkpoints and separately preserving the best model based on validation.

5 Results

Our experimental results compare three generative models - DDPM, VAE, and cGAN - against traditional approaches LS and LMMSE, and another deep learning method DL, which directly minimizes the MSE. We illustrate the generated outputs in Fig. 10 across different delay spread values. The NMSE performance plot shows the distinct characteristics for each model.

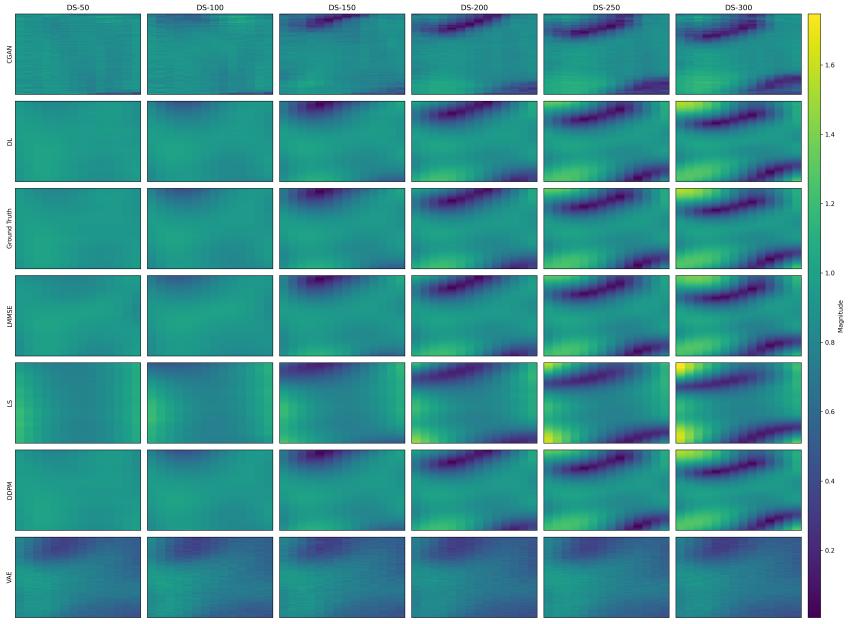


Figure 9: Samples generated by six channel estimation methods against ground truth across varying delay spreads.

The DDPM demonstrates superior performance among the generative approaches, achieving NMSE values between -37 to -27 dB. It shows strong resilience at lower delay spreads but experiences degradation beyond 200 ns, particularly in noisy conditions. Despite this, it maintains a clear

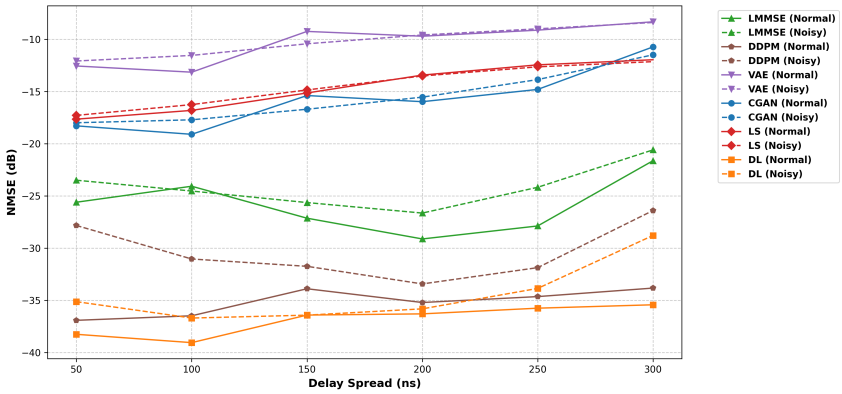


Figure 10: Performance comparison of five channel estimation methods across varying delay spreads.

performance advantage over other generative methods.

The cGAN achieves NMSE values between -18 to -10 dB, performing slightly better than VAE in normal conditions but showing greater sensitivity to noise. Its performance remains relatively consistent across delay spreads, though with a slight degradation at higher values.

The VAE shows moderate performance with NMSE ranging from -12 to -8 dB, exhibits relatively stable behavior across different delay spreads. Notably, the VAE demonstrates similar performance in both normal and noisy conditions, suggesting inherent robustness in its probabilistic modeling approach. However, compared to the baseline LS and LMMSE estimator, the VAE exhibits higher NMSE values, indicating poorer performance in estimating channels in both normal and noisy conditions. The VAE, as a probabilistic model, may have struggled to adapt to channel conditions in the dataset compared to the deterministic models. Although the VAE provides a generative approach, the channel reconstruction accuracy might not be as competitive in this task compared to traditional and other deep learning methods, overall performing the worst among the listed methods.

While all three generative approaches maintain relatively stable performance across varying delay spreads, only DDPM demonstrates competitive performance with the DL-based approach. The VAE and cGAN perform worse than the traditional LS estimator, indicating that current generative modeling methods, except for DDPM, still face challenges in matching conventional channel estimation techniques. However, we also note that it is more challenging to train GANs and VAEs, and our implementations might not have explored the full potential of the respective generative approaches.

6 Conclusion

We investigated the potential of generative modeling approaches for channel estimation in OFDM systems, comparing three different architectures: DDPM, VAE and cGAN. While traditional deterministic approaches like LS and LMMSE estimators, as well as deep learning methods, have shown success in channel estimation, they have limitations such as higher complexity and poor generalization. We explored the channel characteristics, including temporal coherence patterns and frequency-selective fading, which motivated our investigation of distribution-based approaches. Through extensive experiments with varying delay spreads and noisy conditions, DDPM showed the potential to be the most effective generative method. The underperformance of VAE and cGAN highlights that successfully modeling wireless channel distributions remains challenging, suggesting the need for further research in developing generative architectures that can better capture the complex statistical relationships in OFDM channel estimation while maintaining computational efficiency.

Future work for improving channel estimation across all methods can focus on several promising directions. Across all methods, integrating probabilistic models like VAEs, DDPMS, and GANs with

deterministic approaches like DL and LMMSE into hybrid architectures could lead to more robust and adaptable solutions for OFDM channel estimation.

7 Contributions and Acknowledgments

Neal was primarily responsible for the design and implementation of the VAE for OFDM channel estimation. This involved defining the encoder to process the pilot channels, designing the latent space, and creating a decoder to reconstruct the full channel matrix. Neal also fine-tuned the selection and optimization of its hyperparameters, like the KL divergence coefficient, learning rate, and batch size, based on iterative experiments and validation.

Soyeon implemented the DDPM for the channel estimation problem including designing the conditional U-Net that consists of pilot estimate encoder, multi-scale cross-attention modules, and an asymmetric structure optimized for the problem. This task involves designing the baseline DDPM and investigates effective approach of handling pilot condition inputs to preserve the spatial information.

Berkay worked on the problem formulation, the dataset design and generation, the dataset exploration, and the literature review for OFDM channel estimation. He implemented the architecture in the Pix2Pix cGAN paper, closely following every detail, and obtained relevant results on the OFDM dataset after several training iterations. He implemented the baselines consisting of the LMMSE and LS estimators and the deep learning model with convolutional layers and a transformer encoder. He also helped with the training and with designing efficient validation and testing pipelines for the DDPM.

References

[1] J.-J. van de Beek, O. Edfors, M. Sandell, S.K. Wilson, and P.O. Borjesson. On channel estimation in OFDM systems. In *1995 IEEE 45th Vehicular Technology Conference. Countdown to the Wireless Twenty-First Century*, volume 2, pages 815–819 vol.2, 1995.

[2] Y. Li, L.J. Cimini, and N.R. Sollenberger. Robust channel estimation for OFDM systems with rapid dispersive fading channels. *IEEE Transactions on Communications*, 46(7):902–915, 1998.

[3] Yinsheng Liu, Zhenhui Tan, Hongjie Hu, Leonard J. Cimini, and Geoffrey Ye Li. Channel estimation for OFDM. *IEEE Communications Surveys Tutorials*, 16(4):1891–1908, 2014.

[4] Jia-Chin Lin. Least-squares channel estimation for mobile OFDM communication on time-varying frequency-selective fading channels. *IEEE Transactions on Vehicular Technology*, 57(6):3538–3550, 2008.

[5] Vincent Savaux and Yves Louët. LMMSE channel estimation in OFDM context: a review. *IET Signal Processing*, 11(2):123–134, 2017.

[6] Hao Ye, Geoffrey Ye Li, and Biing-Hwang Juang. Power of deep learning for channel estimation and signal detection in OFDM systems. *IEEE Wireless Communications Letters*, 7(1):114–117, 2018.

[7] Qiang Hu, Feifei Gao, Hao Zhang, Shi Jin, and Geoffrey Ye Li. Deep learning for channel estimation: Interpretation, performance, and comparison. *IEEE Transactions on Wireless Communications*, 20(4):2398–2412, 2021.

[8] Physical channels and modulation. Technical Specification 138.211, 3rd Generation Partnership Project (3GPP), 07 2020. Version 16.2.0.

[9] Study on channel model for frequency spectrum above 6 GHz. Technical Report 138.900, 3rd Generation Partnership Project (3GPP), 06 2017. Version 14.2.0.

[10] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5967–5976, 2017.

[11] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[12] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 8162–8171. PMLR, 18–24 Jul 2021.

[13] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

[14] A Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.

[15] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, pages 234–241. Springer, 2015.