IEEE Open Journal of the Communications Society

Received XX Month, XXXX; revised XX Month, XXXX; accepted XX Month, XXXX; Date of publication XX Month, XXXX; date of current version 11 January, 2024.

*Digital Object Identifier 10.1109/OJCOMS.2024.011100*

# Toward 6G: Adaptive Pilot Design and Semi-Blind Channel Estimation in MIMO-OFDMA Systems

**Fayad Haddad[1] *(Graduate Student Member, IEEE)*, Carsten Bockelmann[1] *(Member, IEEE)*, and Armin Dekorsy[1] *(Senior Member, IEEE)*.**

[1]Department of Communications Engineering, University of Bremen, 28359 Bremen, Germany

CORRESPONDING AUTHOR: Fayad Haddad (haddad@ant.uni-bremen.de)

This work was supported in part by the German Federal Ministry of Research, Technology and Space (BMFTR) under Grant 16KISK016 (Open6GHub), under Grant 16KIS2408 (Open6GHub+), and in part by the European Space Agency (ESA) under Contract 4000139559/22/UK/AL (AIComS)

**ABSTRACT** This paper addresses the challenge of pilot overhead reduction in Multi-Input Multi-Output (MIMO) systems employing Orthogonal Frequency Division Multiplexing (OFDM), a promising candidate waveform for future 6G networks. As networks scale with increasing antennas and subcarriers, the pilot overhead also rises, reducing spectral efficiency. Moreover, the stringent reliability requirements of 6G demand a robust channel estimation scheme with minimal pilot overhead. To meet this need, we introduce a Semi-Blind Channel Estimation framework with Adaptive Pilot Design (SBCE-APD) that aims to achieve a target estimation accuracy with minimal pilot overhead. Traditional techniques primarily aim to improve estimation accuracy for a fixed pilot budget. In contrast, the proposed SBCE-APD operates as a closed-loop framework that minimizes pilot overhead subject to a target accuracy constraint. Its main novelty lies in the system-level integration of adaptive pilot-count selection, pilot placement, and semi-blind channel estimation. The method combines two complementary estimation branches: a pilot-based estimator and a non-pilot-based estimator. Their integration forms a semi-blind estimation structure, reducing dependence on pilot symbols. Furthermore, the pilot pattern is adaptively adjusted over the air, optimizing the number of pilots under varying network conditions to maintain the desired accuracy efficiently. Simulation results demonstrate substantial gains in pilot efficiency while preserving reliable channel estimation, establishing SBCE-APD as an effective and scalable solution for 6G MIMO-OFDM networks.

**INDEX TERMS** Semi-blind channel estimation, pilot pattern design, 5G/6G networks, MIMO channel correlation, concrete autoencoder, generative artificial intelligence.

## I. INTRODUCTION

WIRELESS communication systems are rapidly evolving to meet the growing demand for higher data rates. Central to these advancements are technologies such as Multiple-Input Multiple-Output (MIMO), which significantly boost the transmission performance [1]. Alongside MIMO, Orthogonal Frequency Division Multiplexing (OFDM) plays a vital role in 5G systems due to its resilience to multipath fading, and it is expected to remain a key waveform in upcoming 6G networks [2]. In future networks, there is a pressing need for ultra-reliable communication with elevated transmission rates, driven by mission-critical applications such as autonomous vehicles and automation [3].

Achieving the required reliability depends directly on the accuracy of channel estimation [4]. Estimation is typically performed using pilot symbols, known as reference signals [5]. However, pilots occupy transmission resources, leading to a trade-off: increasing the number of pilots improves estimation accuracy but reduces spectral efficiency [6]. Most existing pilot design and adaptation techniques focus on improving estimation accuracy or overall system performance for a given number of pilots. In contrast, this work proposes a pilot reduction strategy where the system continuously evaluates instantaneous channel conditions and compares expected estimation quality against a system-defined target accuracy. Based on this evaluation, the number of pilot

symbols is dynamically adjusted to the minimum required to satisfy the target accuracy, avoiding unnecessary pilot overhead. This objective differs from conventional adaptive pilot schemes, as the goal is not to maximize accuracy, but to meet a system-defined target accuracy with minimal pilots. The basic concept was introduced in our previous work [7]. To the best of the authors' knowledge, this represented an early instance of explicitly targeting pilot overhead minimization under an accuracy constraint, and the present paper significantly extends and generalizes this approach.

### A. RELATED WORKS

Traditional channel estimation approaches, such as Least Squares (LS) and Minimum Mean Square Error (MMSE), typically rely on fixed, uniformly spaced pilot symbols across the resource grid [8]. While simple to implement, these methods often require a large number of pilots to achieve satisfactory accuracy. More advanced techniques, such as Compressed Sensing (CS) [9], reconstruct sparse signals from a limited set of measurements. CS leverages the sparsity of the Channel Impulse Response (CIR) [10] to estimate the channel with fewer pilots. To ensure accurate recovery, CS-based methods typically use randomly placed pilots that maintain incoherence with the sparse domain. Recent studies, such as [11], further improve sparse channel estimation by exploiting both the sparsity and the temporal correlation properties of the wireless channel.

Machine Learning (ML) techniques have shown great potential in optimizing pilot utilization and enhancing channel estimation accuracy. A comprehensive survey of ML-based channel estimation methods is provided in [12], while [13] investigates various Deep Learning (DL) architectures for OFDM systems, comparing their estimation performance and computational demands. Typically, ML-based approaches integrate pilot pattern design with the estimator, known as joint channel estimation and pilot pattern design, as demonstrated in [14]. Building on this trend, ML-driven Generative Artificial Intelligence (GenAI) methods have recently emerged as a promising avenue for improving channel estimation while achieving noticeable pilot overhead reduction [15]. However, these studies focus mainly on optimizing pilot patterns to improve estimation accuracy, rather than minimizing the number of pilots needed to meet a target accuracy. To overcome pilot use limitation, blind channel estimation techniques, which eliminate the need for pilot symbols, have been investigated. These methods exploit the statistical properties of the received signal to estimate the channel, thereby improving spectral efficiency [16], [17]. However, under high mobility or poor channel conditions, non-pilot techniques typically yield less accurate estimates than pilot-based methods [18].

As a balanced solution, semi-blind channel estimation combines a reduced set of pilot symbols with channel statistics. This improves accuracy while reducing pilot overhead, particularly in environments with predictable channel. For example, [19] introduces a semi-blind channel estimation and pilot allocation scheme. However, their method is rigid regarding the number of pilots used. As a result, the quality of estimation can change solely based on the channel statistics. This may lead to overestimation, where excessive pilots are allocated when the channel statistics are very accurate, or underestimation, where insufficient pilots are used when statistical information is poor. Thus, the effectiveness of non-adaptive pilot design is limited in their ability to adjust to variations in underlying channel statistics. Overall, to fully benefit from semi-blind channel estimation, both the number and pattern of the pilots should adapt dynamically to available channel statistics.

### B. CONTRIBUTIONS

We propose a unified Semi-Blind Channel Estimation and Adaptive Pilot Design (SBCE-APD) framework for MIMO-OFDM systems. The core novelty lies in its system-level, closed-loop, constraint-driven pilot optimization. The adaptive pilot count and placement are jointly controlled under a target accuracy constraint. Unlike existing methods that optimize accuracy or overhead in isolation, SBCE-APD ensures pilot resources are used only when necessary, shifting the paradigm from accuracy-first to efficiency-first design. To highlight the main conceptual distinctions between conventional channel estimation and the proposed SBCE-APD framework, we summarize them in Table 1.

**TABLE 1. Conceptual comparison: Channel estimation and SBCE-APD**

| Aspect | Channel Estimation | SBCE-APD |
|---|---|---|
| Objective | Minimize channel estimation error | Minimize pilot overhead under accuracy constraint |
| Pilot design | Fixed | Adaptive |
| System type | Open-loop | Closed-loop |

The framework uses three key pilot reduction strategies:

1) Semi-blind channel estimation: leverages both pilot and channel information to reduce pilot dependency.
2) Strategic pilot placement: optimizes pilot placement to improve estimation performance.
3) Adaptive pilot count: dynamically adjusts the number of pilots to meet the target accuracy.

Building on our prior work [7], this study extends its concept into a comprehensive framework with key contributions:

- **Intelligent Pilot Design:** Unlike our previous work that empirically determines the number of pilots, this study introduces a neural network model. This improves both accuracy and scalability in channel estimation.
- **Joint Pilot Number and Placement Optimization:** The system goes beyond merely determining the number of pilot symbols. It dynamically optimizes both the quantity and placement of pilots, for improved estimation accuracy and resource efficiency.
- **Extension to MIMO Systems:** The method is extended to MIMO systems, leveraging the spatial correlation properties to boost performance and efficiency.

- **Support for Adaptive Bandwidth Allocation:** Unlike prior work with fixed bandwidth, this work dynamically adjusts allocated bandwidth, making it flexible.
- **Realistic Channel Statistics Modeling:** Our approach incorporates realistic channel statistics, accounting for varying conditions.
- **Analysis of Complexity and Practicality:** The work analyzes the computational complexity and practical feasibility for real-world implementation.

Fig. 1 provides a simplified overview of the SBCE-APD framework. At the Base Station (BS), the Adaptive Pilot Design (APD) module generates pilot patterns based on channel information, resource allocation, and the target estimation accuracy, then sends them to the User Equipment (UE). At the UE, channel estimation follows a dual-process approach: pilot-based (Branch_1) and non-pilot-based (Branch_2). Importantly, APD and Branch_2 share same inputs, allowing the APD to assess the expected accuracy of Branch_2 and adjust the number of pilots accordingly.

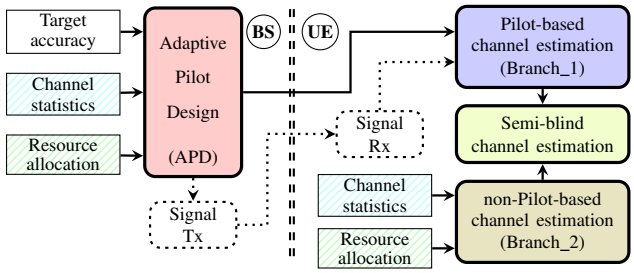


**FIGURE 1. Simplified overview of SBCE-APD. The BS uses APD to design pilots. The UE performs pilot-based (Branch_1) and non-pilot-based (Branch_2) channel estimation, with semi-blind refinement.**

Each component employs a specialized Neural Network (NN). The APD module uses a custom model to determine the pilot count and a Concrete Autoencoder (ConcAE) [20] to generate the pilot pattern. ConcAE, known for feature selection [21], can identify informative pilot positions. The pilot-based estimator uses the ConcAE decoder for channel reconstruction, while the non-pilot-based estimator predicts future channels by capturing spatial, spectral, and temporal correlations from historical data using a Generative Adversarial Network (GAN) [22], inspired by the Spatial-Temporal Nested GAN (STN-GAN) for video inpainting [23]. Although specific ML models are proposed for each module, SBCE-APD is flexible and can adopt alternatives without altering its core concept. Its main contribution is the framework itself, which combines multiple techniques to reduce pilot overhead, including the novel tuning of pilot count to meet desired estimation accuracy. Overall, SBCE-APD provides a scalable solution for dynamic network conditions, making it a strong candidate for future 6G networks.

### C. NOTATIONS

Matrices are denoted by uppercase bold letters, vectors by lowercase bold letters, scalars by lowercase italics. $\odot$ is the Hadamard product. $\mathbb{E}[\cdot]$ is the expectation, $(\cdot)^H$ is the Hermitian, and $(\cdot)^{\frac{1}{2}}$ is the element-wise square root. Main notations and abbreviations used in this work are summarized in Tables 2 and 3, respectively.

**TABLE 2. List of Notations**

| | |
|---|---|
| $\mathbf{G}$ | MIMO channel matrix for all OFDMA users |
| $\mathbf{H}$ | MIMO channel matrix for a single user $u$ |
| $M$ | No. OFDM time symbols |
| $N_\mathrm{t}, N_\mathrm{r}, N$ | No. Tx antenna, Rx antenna, total ($N = N_\mathrm{t}N_\mathrm{r}$) |
| $F, K$ | Total and user-allocated OFDMA subcarriers |
| $\hat{\mathbf{H}}_1, \hat{\mathbf{H}}_2, \hat{\mathbf{H}}$ | estimated channel: Branch_1, Branch_2, and final |
| $\bar{\mathbf{G}}$ | Historical channel observations |
| $\bar{\mathbf{Q}}$ | Historical CQI (Channel Quality Indicator) matrix |
| $\mathbf{X}$, $\mathbf{Y}$ | Transmitted, received signal (data and pilots) |
| $\mathbf{X}_\mathrm{p}$, $\mathbf{Y}_\mathrm{p}$ | Transmitted, received pilots |
| $P$ | No. pilot symbols per antenna |
| $\mathbf{L}$ | Binary matrix indicates pilot positions |
| $\mathbf{C}$ | Binary matrix indicates resource allocation |
| $S, S'$ | Total and user-allocated OFDMA RBGs |
| $\omega$ | Weight representing relevance of historical information |
| $\gamma$ | Target NMSE (Normalized Mean Squared Error) |
| $\mathcal{W}(\cdot)$ | Function computing weight $\omega$ from historical channels |
| $\mathcal{Q}(\cdot)$ | Function of decoder of the ConcAE |
| $\mathcal{R}(\cdot)$ | Function of channel estimation in Branch_1 |
| $\mathcal{X}(\cdot)$ | Function to insert pilots into transmitted signal |
| $\mathcal{X}^{-1}(\cdot)$ | Function to extract pilots from received signal |
| $\mathcal{L}(\cdot)$ | Function of retrieving elements from Look-Up Table |
| $\mathcal{P}(\cdot)$ | Function to compute required pilot count |
| $\mathcal{G}(\cdot)$ | Function generator of STN-GAN (Branch_2) |
| $\mathcal{D}(\cdot)$ | Function discriminator of STN-GAN |

**TABLE 3. List of main Abbreviations**

| | |
|---|---|
| SBCE-APD | Semi-Blind Channel Estimation - Adaptive Pilot Design |
| GAN | Generative Adversarial Network |
| STN-GAN | Spatial Temporal Nested - GAN |
| ConcAE | Concrete Autoencoder |
| PCC | Pilot Count Calculator |
| RBG | Resource Block Group |

## II. SYSTEM AND CHANNEL MODELS

### A. SYSTEM MODEL

We consider a MIMO-OFDM system with $F$ subcarriers. The system comprises one BS equipped with $N_\mathrm{t}$ transmit antennas, and serves $U$ users, each equipped with $N_\mathrm{r}$ receive antennas. Consequently, the total number of spatial MIMO channels is given by $N = N_\mathrm{t}N_\mathrm{r}$ per user. To capture the dynamics of a time-varying channel, the system considers $M$ OFDM symbols over time. For simultaneous data transmission, the system OFDM resources are shared by all $U$ users with Orthogonal Frequency Division Multiple Access (OFDMA). The channel matrix $\mathbf{G} \in \mathbb{C}^{F\times N\times M}$, represents the MIMO channel responses across both time and frequency that are available for all users. During each OFDM symbol $m \in [1, M]$, the subcarriers are grouped into $S$ Resource Block Groups (RBGs), with each group containing $F_\mathrm{rbg} = F/S$ subcarriers. For each user $u \in [1, U]$, a subset

$S'$ of RBGs are allocated, with $0 \leq S' \leq S$, thus, the entire number of subcarriers allocated is $K = S' \cdot F_{\text{rbg}}$. These allocated RBGs can be either contiguous or non-contiguous. Fig. 2 illustrates an example of an OFDMA system with the bandwidth divided into $S = 4$ RBGs, and the time span covers $M = 14$ OFDM symbols. For simplicity, the figure assumes $N_{\text{t}} = 2$ and $N_{\text{r}} = 1$. It shows the resource allocation on the OFDM grid for $U = 4$ users, with gray-shaded regions on the 15th OFDM symbol indicating future resources that are not yet allocated. The figure also shows that the system utilizes a single data stream per user, as indicated by the identical resource allocation across MIMO channels. However, when multiple data streams are used, the resource allocation across MIMO channels may vary, with the maximum number of streams being $\min(N_{\text{t}}, N_{\text{r}})$ [24].

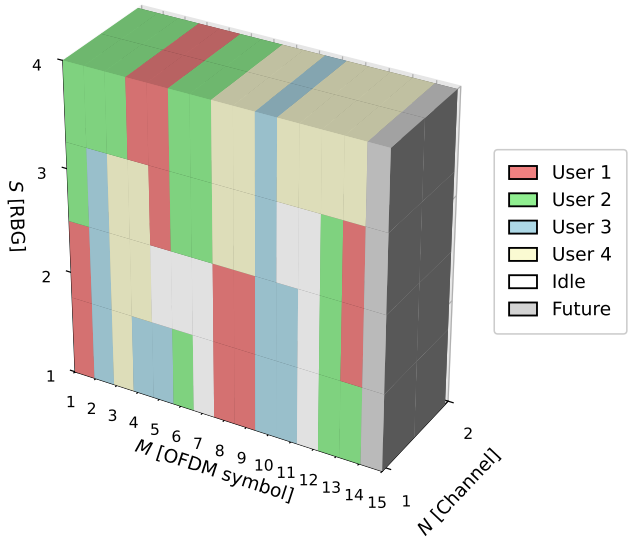


**FIGURE 2. OFDMA system: Allocation of subcarrier blocks (RBGs) in MIMO system with $S = 4$ RBGs, $M = 14$ OFDM symbol, and $N = 2$ MIMO channels. One data stream per user is considered.**

For the $u$th user, the transmitted signal during the $m$th OFDM symbol and over $K$ allocated subcarriers is denoted as $\mathbf{X}_{u,m} \in \mathbb{C}^{K \times N_{\text{t}}}$. Focusing on a single user and OFDM symbol, the indices $u$ and $m$ are omitted. The received signal is denoted by $\mathbf{Y} \in \mathbb{C}^{K \times N_{\text{r}}}$. The transmitted signal includes both pilot and data symbols, with $P$ pilot symbols assigned for each MIMO channel. Accordingly, the transmitted and received pilot symbols are $\mathbf{X}_{\text{p}} \in \mathbb{C}^{P \times N_{\text{t}}}$ and $\mathbf{Y}_{\text{p}} \in \mathbb{C}^{P \times N_{\text{r}}}$. At each subcarrier $k \in [1, \ldots, K]$, the received signal is:

$$\mathbf{y}_k = \mathbf{H}_k \mathbf{x}_k + \mathbf{z}_k, \tag{1}$$

where $\mathbf{y}_k \in \mathbb{C}^{N_{\text{r}}}$, $\mathbf{x}_k \in \mathbb{C}^{N_{\text{t}}}$. $\mathbf{H}_k \in \mathbb{C}^{N_{\text{r}} \times N_{\text{t}}}$ is the channel matrix and $\mathbf{z}_k \in \mathbb{C}^{N_{\text{r}}}$ is the noise with zero mean and variance $\sigma_{\text{z}}^2$ per element, respectively. The channel over all subcarriers can be represented after reshaping as $\mathbf{H} \in \mathbb{C}^{K \times N}$, with $N = N_{\text{t}} N_{\text{r}}$.

To quantify noise, we define the Signal to Noise Ratio (SNR) as $\frac{E\{\|\mathbf{X}\|_F^2\}}{\sigma_{\text{z}}^2}$, assuming equal power for pilots and data, and independent and identically distributed (i.i.d.) noise. The channel $\hat{\mathbf{H}}$ is estimated using the pilots in $\mathbf{X}$. Since channel estimation is imperfect, $\hat{\mathbf{H}}$ may differ from the true channel $\mathbf{H}$. Thus, performance if measured using Normalized Mean Squared Error (NMSE), as: $\text{NMSE} = \frac{E\{\|\mathbf{H}-\hat{\mathbf{H}}\|_F^2\}}{E\{\|\mathbf{H}\|_F^2\}}$.

The distinction between $\mathbf{G}$ and $\mathbf{H}$ lies in their scope within the OFDMA. $\mathbf{G}$ represents the full MIMO channel matrix, capturing the channel responses across all MIMO spatial channels for all users and over $M$ OFDM symbols. Whereas, $\mathbf{H}$ contains the channel coefficients on the allocated resources for the $u$th user on the $m$th OFDM symbol.

### *B. CHANNEL MODEL*

In wireless communication, multipath propagation causes the transmitted signal to reach the receiver through multiple paths with different delays and attenuation levels [25]. These effects are modeled using the Tapped Delay Line (TDL) model, where the channel is represented by discrete taps with distinct delays and gains. For the $n$th MIMO channel and $k$th subcarrier, the complex channel gain of OFDM symbol $m$ is given by [26]:

$$h_{k,n}(m) = \sum_{l=1}^{L} \beta_{k,n,l}\, e^{j2\pi f^{\text{D}}_{k,n,l}(\tau_{k,n,l} - m)}, \tag{2}$$

where $L$ is the number of propagation paths, and $\tau_{k,n,l}$, $\beta_{k,n,l}$, and $f^{\text{D}}_{k,n,l}$ denote the delay, gain, and Doppler frequency of the $l$th path. The term $(\tau_{k,n,l} - m)$ captures the delay between the path and the OFDM symbol index, aligning multipath components with symbol timing in time-varying channels. This model reflects temporal and spectral correlations [27]: temporal correlation stems from delay variations over time, while spectral correlation arises from Doppler-induced frequency variations across subcarriers.

The model discussed so far does not include spatial correlation in MIMO channels. To account for it, the Weichselberger model [28] is employed. The uncorrelated channel responses for each antenna pair $n_{\text{t}} \in [1, N_{\text{t}}]$ and $n_{\text{r}} \in [1, N_{\text{r}}]$, and time $m$ are arranged into $K$ matrices, with each matrix $\mathbf{H}_k^{\text{un}} \in \mathbb{C}^{N_{\text{t}} \times N_{\text{r}}}$ representing the uncorrelated channel response:

$$\mathbf{H}_k^{\text{un}}(m) = \begin{bmatrix} h_{k,1,1}(m) & \cdots & h_{k,1,N_{\text{r}}}(m) \\ \vdots & \ddots & \vdots \\ h_{k,N_{\text{t}},1}(m) & \cdots & h_{k,N_{\text{t}},N_{\text{r}}}(m) \end{bmatrix}. \tag{3}$$

Using the Weichselberger model, the spatially correlated channel is expressed as:

$$\mathbf{H}_k(m) = \mathbf{U}_{\text{r}} \left( \mathbf{\Omega}^{\frac{1}{2}} \odot \mathbf{H}_k^{\text{un}}(m) \right) \mathbf{U}_{\text{t}}^H, \tag{4}$$

where $\mathbf{U}_{\text{r}}$ and $\mathbf{U}_{\text{t}}$ are unitary matrices containing the eigenvectors of the transmit and receive correlation matrices, respectively. These correlation matrices are obtained as:

$$\begin{aligned} \mathbf{R}_{\text{t}} &= \mathbb{E}\big[\mathbf{H}_k^{\text{un}}(m) \big(\mathbf{H}_k^{\text{un}}(m)\big)^H\big], \\ \mathbf{R}_{\text{r}} &= \mathbb{E}\big[\big(\mathbf{H}_k^{\text{un}}(m)\big)^H \mathbf{H}_k^{\text{un}}(m)\big], \end{aligned} \tag{5}$$

with $\mathbf{R}_{\text{t}} \in \mathbb{C}^{N_{\text{t}} \times N_{\text{t}}}$ and $\mathbf{R}_{\text{r}} \in \mathbb{C}^{N_{\text{r}} \times N_{\text{r}}}$. The matrix $\mathbf{\Omega}$ is a power coupling matrix that quantifies how much average energy is exchanged between the eigenvectors at the transmit and receive sides. It can be estimated as:

$$\mathbf{\Omega} = \mathbb{E}\big[|\mathbf{U}_{\text{t}}^H \mathbf{H}_k^{\text{un}}(m) \mathbf{U}_{\text{r}}|^2\big]. \tag{6}$$

Accordingly the spatially correlated Channel Frequency Response (CFR) of the MIMO system $\mathbf{H}$ is represented as:

$$\mathbf{H} = \begin{bmatrix} \mathbf{h}_1(1) & \cdots & \mathbf{h}_1(M) \\ \vdots & \ddots & \vdots \\ \mathbf{h}_K(1) & \cdots & \mathbf{h}_K(M) \end{bmatrix}, \tag{7}$$

with $\mathbf{h}_k(m) \in \mathbb{C}^N$ is the flattened representation of $\mathbf{H}_k(m)$ from the Weichselberger model in (4).

### C. CHANNEL CORRELATION

Channel correlation describes dependencies in the channel across time, frequency, and space. Temporal correlation reflects variations of the channel on the same subcarrier over time due to UE mobility [29], quantified by the coherence time $d_{\text{tc}} = \frac{c}{2vf_c}$, where $v$ is user velocity, $f_c$ the carrier frequency, and $c$ the speed of light. Spectral correlation measures similarity of channel responses across frequencies due to multipath and Doppler effects [29], with coherence bandwidth $d_{\text{fc}} = \frac{1}{\Delta_{\text{ds}}}$ , where $\Delta_{\text{ds}}$ is the channel delay spread. Spatial correlation captures the relationship between channels at different antennas, influenced by array geometry, inter-element spacing, and angular spread of multipath components [30], with high correlation reducing MIMO performance and low correlation improving spatial multiplexing.

### D. HISTORICAL INFORMATION AND ITS WEIGHT

For each user, past channel estimates exist at the allocated resource positions. Each estimate is associated with a Channel Quality Indicator (CQI), computed per RBG based on the measured Signal-to-Interference-and-Noise Ratio (SINR) according to 3GPP specifications [31]. Together, these estimates and CQIs form the user's historical information, which can be leveraged to perform future channel prediction.

Fig. 2 shows the history of allocated resources spanning the past $M$ OFDM symbols, while the future allocation is considered over a single OFDM symbol. Accordingly, for the $u$th user, we define the binary matrices $\mathbf{C}_u^{\text{past}} \in \mathbb{B}^{F\times N\times M}$ and $\mathbf{C}_u \in \mathbb{B}^{F\times N}$ to represent the past and future allocations, corresponding to the colored and gray regions in the figure, respectively. In both matrices, a value of one indicates an allocated resource, while a value of zero denotes an unallocated one.

Let $\mathbf{G}$ denote the full matrix of estimated channel coefficients for the entire OFDM system, as introduced in Section II A. The past estimated channel for user $u$ is then given by $\bar{\mathbf{G}}_u = \mathbf{G} \odot \mathbf{C}_u^{\text{past}}$. The resulting matrix $\bar{\mathbf{G}}_u$ has the same dimensions as $\mathbf{G}$, but contains zeros in positions not allocated to user $u$, and retains the estimated channel coefficients at the allocated positions. Similarly, the CQI values are organized in a quality matrix $\mathbf{Q} \in \mathbb{R}^{F\times N\times M}$,having the same dimensional alignment with $\mathbf{G}$. For user $u$, the historical CQI can be represented as $\bar{\mathbf{Q}}_u = \mathbf{Q} \odot \mathbf{C}_u^{\text{past}}$. Together, $\bar{\mathbf{G}}_u$ and $\bar{\mathbf{Q}}_u$ define the user's historical information. For the remainder of this paper, we omit the user index for clarity, as our analysis focuses on the performance of a single user within the multiple access system.

To quantify the relevance of this information, we introduce a weight parameter $\omega \in [0, 1]$. A higher $\omega$ indicates stronger correlation between past and future channel coefficients, implying that fewer pilots are needed to achieve a target estimation quality. This parameter plays a central role in our adaptive pilot pattern design strategy, which leverages the quality of historical information to optimize the required number of pilots $P$. The computation of $\omega$ is based on the temporal, spectral, and spatial correlation properties of the MIMO channels, as mentioned in Sections II-C, considering the SNR value corresponding to each estimate. A higher value of $\omega$ implies stronger correlation between past and future channel coefficients, indicating that fewer pilots are needed while maintaining a target estimation quality.

The concept of the historical channel weight $\omega$ was first introduced in our previous work [7], where it was computed using an Euclidean distance-based approach reflecting only the correlation properties. In contrast, the current work proposes a NN-based method, accounting for both the correlation properties and the quality of past estimated channels. Further details on the computation of $\omega$ and its role in channel estimation are provided in subsequent sections.

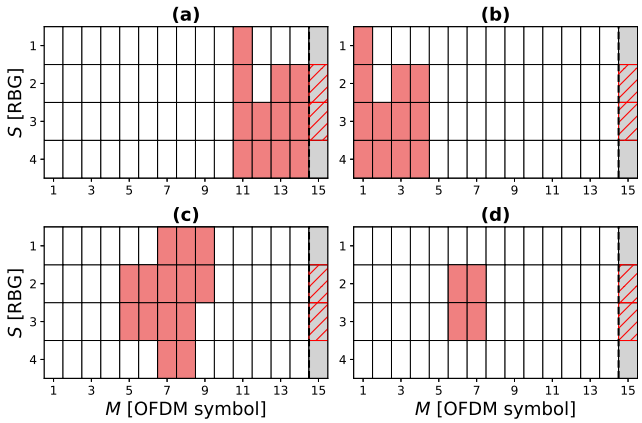


**FIGURE 3. Examples of OFDMA RBG allocation for a single user. Red blocks show past allocation and hashed blocks show future allocation. (a,b) have equal number of past blocks but (a) is temporally closer (higher weight). (c,d) have equal average temporal separation but (c) has more past blocks (higher weight).**

Fig. 3 illustrates how weights are assigned from previously allocated resources. Hashed blocks indicate future RBG allocations, while red blocks represent past allocations. In comparing subfigures (a) and (b), scenario (a) is expected to have a higher weight because its past allocations are temporally closer to the future ones, implying stronger correlation. The same principle applies in the frequency or spatial domain, where smaller separation indicates higher correlation. Between subfigures (c) and (d), scenario (c) is likely to attain a greater weight owing to its larger number of past allocations. Overall, the weight reflects both temporal and spectral proximity as well as the quantity of past allocations, capturing their relevance for predicting future RBG assignments.

## III. CONVENTIONAL METHODS FOR CHANNEL ESTIMATION

Conventional OFDM channel estimation methods, such as LS and MMSE [32], are widely used in wireless systems and serve as standard benchmarks for evaluating.

### A. LEAST SQUARES

The LS estimator is a fundamental, low-complexity method that minimizes the squared error between the received and transmitted pilot symbols. Referring back to the system model in Section II-B, $\mathbf{X}_\mathrm{p}$ and $\mathbf{Y}_\mathrm{p}$ are defined to carry the pilot symbols. The general LS estimate at the pilot positions is given by:

$$\hat{\mathbf{H}}_\mathrm{p} = (\mathbf{X}_\mathrm{p}^\mathrm{H}\mathbf{X}_\mathrm{p})^{-1}\mathbf{X}_\mathrm{p}^\mathrm{H}\mathbf{Y}_\mathrm{p}, \tag{8}$$

where $\hat{\mathbf{H}}_\mathrm{p} \in \mathbb{C}^{P\times N}$ is the estimated channel coefficients at pilot positions, respectively. The full channel estimate is obtained via linear interpolation as $\hat{\mathbf{H}}^\mathrm{LS} = \mathbf{A}_\mathrm{LI} \cdot \hat{\mathbf{H}}_\mathrm{p}$, with $\mathbf{A}_\mathrm{LI} \in \mathbb{C}^{K\times P}$ being the linear interpolation matrix [33].
While computationally efficient, LS is sensitive to noise because it does not exploit statistical knowledge of the channel. Assuming zero-mean additive noise Gaussian with variance $\sigma_z^2$, the estimation error at pilot positions is

$$\hat{\mathbf{H}}_\mathrm{p} - \mathbf{H}_\mathrm{p} = (\mathbf{X}_\mathrm{p}^\mathrm{H}\mathbf{X}_\mathrm{p})^{-1}\mathbf{X}_\mathrm{p}^\mathrm{H}\mathbf{Z}_\mathrm{p}, \tag{9}$$

where $\mathbf{H}_\mathrm{p} \in \mathbb{C}^{P\times N}$ denotes the true channel coefficients at pilot positions, $\mathbf{Z}_\mathrm{p}$ is the noise at pilot positions. The corresponding mean squared error (MSE) is

$$\mathrm{MSE}^\mathrm{LS} = \sigma_z^2 \,\mathrm{tr}\Big((\mathbf{X}_\mathrm{p}^\mathrm{H}\mathbf{X}_\mathrm{p})^{-1}\Big). \tag{10}$$

### B. MINIMUM MEAN SQUARE ERROR

The MMSE estimator uses the channel and noise statistics to minimize the mean squared error under Gaussian noise. Formally, the MMSE estimate of the channel matrix $\hat{\mathbf{H}}$ is:

$$\hat{\mathbf{H}}^\mathrm{MMSE} = \arg\min_{\hat{\mathbf{H}}} E\left\{\|\mathbf{H} - \hat{\mathbf{H}}\|_2^2\right\}. \tag{11}$$

In practical systems, a linear MMSE estimator can be typically used. It computes the estimate by applying a linear transformation to the LS estimate as $\hat{\mathbf{H}}^\mathrm{MMSE} = \mathbf{A}^\mathrm{MMSE}\,\hat{\mathbf{H}}_\mathrm{p}^\mathrm{LS}$. The optimal transformation matrix $\mathbf{A}^\mathrm{MMSE}$ minimizes:

$$\mathbf{A}^\mathrm{MMSE} = \arg\min_{\mathbf{A}_\mathrm{LI}} E\left\{\|\mathbf{H} - \mathbf{A}_\mathrm{LI}\,\hat{\mathbf{H}}_\mathrm{p}\|_2^2\right\}. \tag{12}$$

Assuming the LS estimate is unbiased and uncorrelated with the noise, the optimal matrix $\mathbf{A}^\mathrm{MMSE}$ is given by:

$$\mathbf{A}^\mathrm{MMSE} = \mathbf{R}_\mathrm{hp}\left(\mathbf{R}_\mathrm{pp} + \sigma_z^2\mathbf{I}\right)^{-1}, \tag{13}$$

where $\mathbf{R}_\mathrm{hp} \in \mathbb{C}^{F\times P}$ is the cross correlation matrix between actual channel matrix to be estimated and pilot matrix $\mathbf{H}_\mathrm{p}$. $\mathbf{R}_\mathrm{pp} \in \mathbb{C}^{P\times P}$ is the autocorrelation matrix of $\mathbf{H}_\mathrm{p}$. Thus, the final Linear MMSE estimate of the full channel is:

$$\hat{\mathbf{H}}^\mathrm{MMSE} = \mathbf{R}_\mathrm{hp}\left(\mathbf{R}_\mathrm{pp} + \sigma_z^2\mathbf{I}\right)^{-1}\,\hat{\mathbf{H}}_\mathrm{p}. \tag{14}$$

MMSE is effective but challenging in practice due to the need for accurate channel and noise statistics, though it performs well when these are known.

## IV. OVERVIEW OF THE UTILIZED NEURAL NETWORKS

This section introduce the ML-based neural networks that are utilized through this work.

### A. SPATIAL-TEMPORAL NESTED GAN

The neural network used for Branch_2 is based on the STN-GAN architecture [23], a conditional GAN [34] originally developed for video inpainting. Its nested design with 3D residual blocks captures spatiotemporal correlations, enabling coherent reconstruction and joint exploitation of temporal, spectral, and spatial channel dependencies. A reduced-depth variant is adopted to lower complexity while maintaining temporal consistency across consecutive OFDM symbols. The resulting architecture is illustrated in Fig. 4.

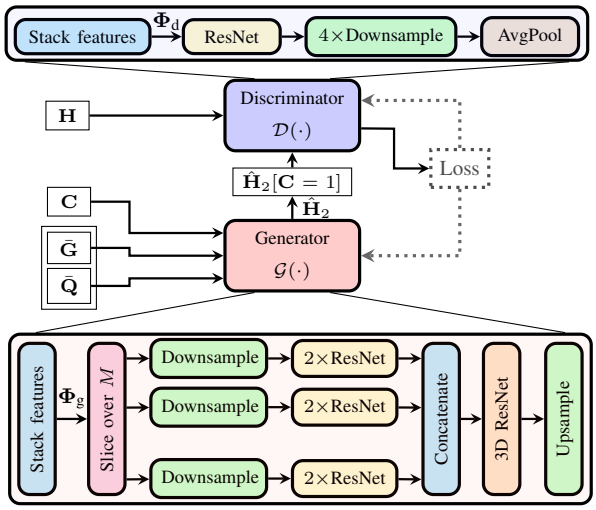


**FIGURE 4. STN-GAN architecture used in this work, showing the generator and discriminator networks and their main processing blocks (feature stacking, down/up-sampling, ResNet blocks, and masking).**

For user $u$, the generator $\mathcal{G}(\cdot)$ exploits historical channel channel information $\bar{\mathbf{G}}$ and $\bar{\mathbf{Q}}$ from the past $M$ OFDM symbols, together with the resource allocation matrix $\mathbf{C}$, to predict the channel $\hat{\mathbf{H}}_2 \in \mathbb{C}^{F\times N}$. As the user is assigned only $K$ out of $F$ subcarriers, $\hat{\mathbf{H}}_2$ is element-wise masked by $\mathbf{C}$, enforcing zero-valued coefficients on unallocated. The discriminator $\mathcal{D}(\cdot)$ subsequently assesses the predicted channel restricted to the allocated subcarriers, $\hat{\mathbf{H}}_2[\mathbf{C}=1] \in \mathbb{C}^{K\times N}$, by comparing it with the corresponding ground-truth channel $\mathbf{H} \in \mathbb{C}^{K\times N}$.
*Generator architecture:* The generator forms its input by stacking the real-valued $\bar{\mathbf{Q}}$ with the real and imaginary components of $\bar{\mathbf{G}}$, yielding $\mathbf{\Phi}_\mathrm{g} \in \mathbb{R}^{3\times F\times N\times M}$. This representation is partitioned along the temporal dimension $M$, and each slice is processed in parallel through a stride 2 convolutional layer for downsampling, followed by a ResNet block [35] for spatial-frequency features extraction. Temporal correlations across the $M$ previous OFDM symbols are captured by a 3D ResNet block with kernel size $(1,1,M)$ and stride $(1,1,M)$, which aggregates historical information to infer the next symbol. The resulting features are upsampled via a transposed convolutional layer to produce $\hat{\mathbf{H}}_2$.

*Discriminator architecture:* The discriminator receives both $\hat{\mathbf{H}}_2[\mathbf{C}=1]$ and $\mathbf{H}$ as input. The real and imaginary parts are stacked along a separate feature dimension, forming $\mathbf{\Phi}_{\mathrm{d}} \in \mathbb{R}^{4\times K\times N}$. This tensor passes through a residual block for spatial–frequency feature extraction, followed by four convolutional layers with stride 2 for progressive downsampling. Global average pooling produces a scalar output serving as the critic score in the training loss.

### B. CONCRETE AUTOENCODER

The Concrete AutoEncoder (ConcAE) [20] is a DL model for differentiable feature selection, making it well suited for identifying optimal pilot positions for channel estimation. As shown in Fig. 5, it comprises a concrete selector layer (encoder) followed by a three-layer Multi-Layer Perceptron (MLP) [36] (decoder). The encoder receives the channel realization across the allocated subcarriers, i.e. the vector $\mathbf{h} \in \mathbb{C}^K$. Through the concrete selector layer, a low-dimensional representation $\mathbf{\Psi} = [\psi_1, \ldots, \psi_P]$ is generated via a sparsity-inducing selection function, expressed as $\mathbf{\Psi} = \mathcal{S}(\mathbf{h})$. This selection operation can be interpreted as multiplication by a sparse binary matrix $\mathbf{W} \in \mathbb{R}^{P\times K}$ that retains only the $P$ most informative subcarriers, such that $\mathbf{\Psi} = \mathbf{W}\mathbf{h}$. The decoder then reconstructs the full channel vector using $\hat{\mathbf{h}} = \mathcal{Q}(\mathbf{\Psi})$. After training, the learned matrix $\mathbf{W}$ explicitly reveals the subcarrier indices that contribute most significantly to the representation $\mathbf{\Psi}$, thereby identifying the pilot positions that yield the highest channel-estimation accuracy.

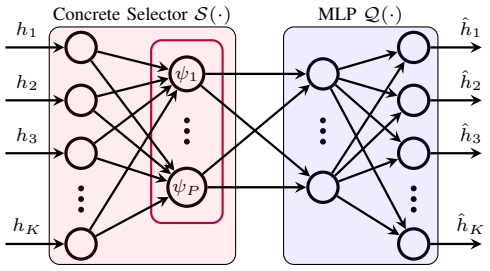


**FIGURE 5. Concrete autoencoder for pilot position selection. A concrete selector chooses $P$ informative subcarriers from the input channel vector, and an MLP decoder reconstructs the full channel.**

*Concrete Selector* uses the Gumbel-Softmax (concrete) distribution for differentiable sampling via continuous relaxation [37]. For each output unit $p$, it defines learnable parameters $\boldsymbol{\alpha}_p = [\alpha_{1,p}, \alpha_{2,p}, ..., \alpha_{K,p}] \in \mathbb{R}^K_{>0}$ as unnormalized scores reflecting preference for each input. A temperature parameter $T > 0$ regulates the relaxation smoothness. The soft selection weight for input $k$ and output $p$ is given by:

$$w_{k,p} = \frac{\exp((\log \alpha_{k,p} + g_{k,p})/T)}{\sum_{j=1}^{K} \exp((\log \alpha_{j,p} + g_{j,p})/T)}, \qquad (15)$$

where each $g_{k,p}$ is sampled from a Gumbel distribution, defining a soft approximation to discrete feature selection. The ConcAE is trained on channel realizations $\mathbf{h}$ to identify the most informative pilot positions. At the beginning of training, the selection parameters $\boldsymbol{\alpha}$ are initialized with small positive values to promote diverse feature combinations, while a high temperature $T$ yields smooth soft selections. As training proceeds, $T$ is gradually annealed toward zero, driving each unit to focus on a single dominant feature. During inference, the soft selections become hard, and the selected features directly correspond to pilot positions, enabling data-driven pilot optimization.

### C. ATTENTION-BASED CHANNEL ESTIMATOR

For comparison with state-of-the-art neural channel estimators, we include the Channelformer network [38], although it is not part of the proposed SBCE-APD framework. Channelformer employs an encoder-decoder architecture in which the encoder uses multi-head self-attention to reweight the LS estimate and emphasize the most informative components, while the decoder applies residual convolutional blocks to produce a refined channel estimate, as illustrated in Fig. 6.

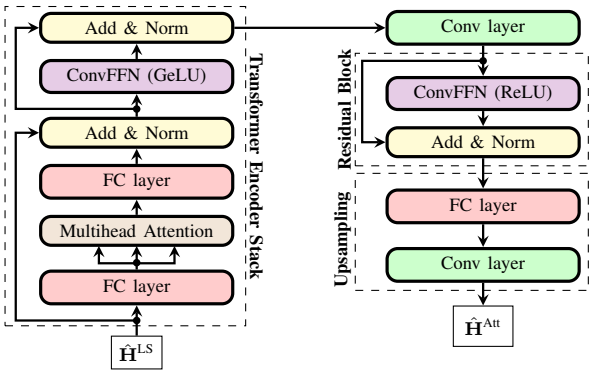


**FIGURE 6. Channelformer baseline architecture. A transformer encoder with multi-head self-attention refines the LS estimate, followed by a residual convolutional decoder to produce the final channel estimate.**

*Channelformer Encoder:* It begins by linearly projecting the LS estimate $\hat{\mathbf{H}}^{\mathrm{LS}} \in \mathbb{C}^{K\times N}$ into query, key, and value tensors through a Fully Connected (FC) layer. These serve as the inputs to a multi-head self-attention module, which identifies LS components that show a strong correlation with the underlying channel. In this way, the encoder emphasizes the most informative subcarriers instead of treating all inputs as equally important. The attention outputs from all heads are concatenated and processed through an Add & Norm layer with a residual connection. This stabilizes training while preserving information. A Convolutional Feed-Forward Network (ConvFFN) with GELU activation then refines the representation further and increases its expressive power.

*Channelformer Decoder:* It applies a set of residual convolutional blocks to denoise and reconstruct the channel. It starts with a convolutional layer, followed by a ConvFFN with ReLU activation. The block is wrapped with an Add & Norm layer to maintain feature consistency and reduce degradation when stacking layers. The final stage is an upsampling module composed of a Fully Connected layer and a final convolutional layer. This module restores the spatial structure across subcarriers and OFDM symbols and outputs the refined channel estimate $\hat{\mathbf{H}}^{\mathrm{Att}} \in \mathbb{C}^{K\times N}$. The decoder enhances robustness to different SNR conditions and enables accurate reconstruction of the full channel matrix.

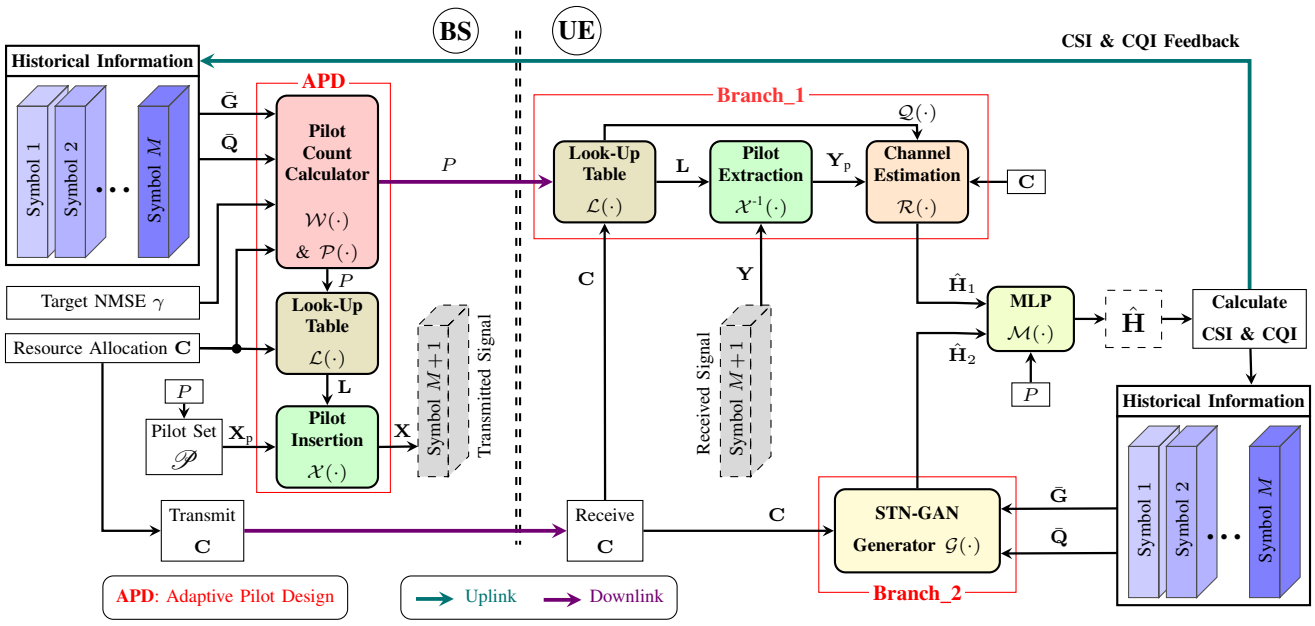


**FIGURE 7. Block diagram of the SBCE-APD framework, highlighting its two main branches for channel estimation. Branch_1 (pilot-based) uses the APD module at the BS to generate adaptive pilot patterns, which are then transmitted to the UE for channel estimation $\hat{\mathbf{H}}_1$. Branch_2 (non-pilot-based) employs the STN-GAN generator to estimate the channel $\hat{\mathbf{H}}_2$ solely from historical channel information $(\bar{\mathbf{G}}, \bar{\mathbf{C}})$. The combined estimates are fused via an MLP to produce the final $\hat{\mathbf{H}}$, which is then used to compute CSI/CQI feedback.**

## V. SEMI-BLIND CHANNEL ESTIMATION AND ADAPTIVE PILOT DESIGN FRAMEWORK

Building on the conceptual model shown in Fig. 1, the detailed structure of the SBCE-APD framework is illustrated in Fig. 7. The central objective of SBCE-APD is to minimize pilot overhead while guaranteeing a predefined channel estimation accuracy, expressed as a target NMSE $\gamma$.

In the adaptive scheme, the pilot pattern can be generated at either the BS or UE and communicated to the other end. It is represented by a binary matrix $\mathbf{L} \in \mathbb{B}^{K \times N}$, with ones indicating pilot positions. Due to the high computational cost of pilot design, this task is performed at the BS, which has greater processing power and energy resources than the UE. Updating the full pattern $\mathbf{L}$ at every estimation instance would incur significant signaling overhead. To address this, the pilot design process is split into two stages: first, determining the required number of pilots $P$, and second, selecting their positions within $\mathbf{L}$. In SBCE-APD, computing $P$ is the more demanding step and is therefore centralized at the BS, while position selection is carried out simultaneously at both the BS and the UE. This approach requires transmitting only the integer $P$ to the UE, resulting in a small downlink signaling overhead.

At the BS, within the APD block, the Pilot Count Calculator (PCC) module determines the pilot count $P$ and send it to the UE. The pilot pattern $\mathbf{L}$ is then selected at both BS and UE using identical predefined Look-Up Tables (LUTs), as shown in Fig. 7. in addition to $P$, the LUTs take the resource allocation matrix $\mathbf{C}$ as input, which is determined at the BS and transmitted to the UE as part of standard 3GPP. Importantly, the signaling associated with $\mathbf{C}$ is inherent to system operation and does not originate from the adaptive pilot design mechanism [39]. Consequently, the only additional signaling required by SBCE-APD is a few bits to convey $P$. As an example, in a system with $S = 4$ RBGs, each consisting of $F_{\text{rbg}} = 32$ subcarriers and supporting up to two pilots, the maximum pilot count is $P = 8$, which requires only 3 bits of signaling. While updating $P$ introduces a small overhead, adaptive pilot design significantly reduces the total number of pilot symbols by allocating only the minimum required to meet the target accuracy. Finally, within the APD at the BS, a subset of $P$ pilot symbols $\mathbf{X}_{\text{p}}$ is selected from the candidate pilot set $\mathscr{P}$ and inserted at the designed positions, yielding the transmit signal $\mathbf{X} = \mathcal{X}(\mathbf{X}_{\text{p}}, \mathbf{L})$.

At the UE, semi-blind channel estimation is performed through two parallel branches. Branch_1 performs pilot-based channel estimation by first receiving $P$ and then determining the corresponding pilot pattern $\mathbf{L}$ using the LUT at the UE. Using $\mathbf{L}$, the UE then extracts the pilot symbols from the received signal $\mathbf{Y}$ by applying the inverse of the BS insertion function $\mathcal{X}(\cdot)$, yielding $\mathbf{Y}_{\text{p}} = \mathcal{X}^{-1}(\mathbf{Y}, \mathbf{L})$. The LUT used at the UE corresponds to the same entry as that used at the BS. A subtle difference, however, is that the UE-side LUT additionally stores a tailored trained ConcAE decoder $\mathcal{Q}(\cdot)$ associated with each pilot pattern $\mathbf{L}$. This decoder produces an initial channel estimate over the $K$ allocated resources, which is reshaped to the full frequency band $F$ using the resource allocation $\mathbf{C}$, yielding the pilot-based estimate $\hat{\mathbf{H}}_1 \in \mathbb{C}^{F \times N}$. Branch_2 performs non-pilot-based channel estimation using a pre-trained STN-GAN generator. This branch exploits the historical information $\bar{\mathbf{G}}$ and $\bar{\mathbf{Q}}$, and the resource allocation $\mathbf{C}$ to generate

the channel estimate $\hat{\mathbf{H}}_2 \in \mathbb{C}^{F\times N}$. The outputs of the two branches, together with the number of pilots $P$, are concatenated and processed by a lightweight three-layer MLP defined as function $\mathcal{M}(\cdot)$, to produce the refined channel estimate $\hat{\mathbf{H}} = \mathcal{M}(\hat{\mathbf{H}}_1, \hat{\mathbf{H}}_2, P) \in \mathbb{C}^{K\times N}$. The MLP adaptively balances the reliability of the pilot- and non-pilot-based branches: higher $P$ increases reliance on $\hat{\mathbf{H}}_1$, while lower $P$ favors $\hat{\mathbf{H}}_2$. The final estimate is then used to derive the CQI and explicit Channel State Information (CSI) in accordance with 3GPP specifications [31]. The explicit CSI feedback involves compressing the estimated channel matrix and transmitting it to the BS [40]. As a result, the BS gains access to the same channel information available at the UE. This information is accumulated over time to form historical channel knowledge, which is subsequently exploited in Branch_2 at the UE and serves as a key input to the PCC module at the BS. By leveraging this historical information, the PCC anticipates the expected performance of the non-pilot-based estimator, i.e., the STN-GAN generator, and determines the minimum pilot count $P$ required to satisfy the target estimation accuracy.

### A. SBCE-APD COMPONENTS FUNCTIONALITY

In the following, we provide detailed descriptions of the three main components of the proposed framework: the adaptive pilot design APD, Branch_1, and Branch_2. We then describe the training process in another clause.

#### 1) ADAPTIVE PILOT DESIGN

To perform adaptive pilot design, the BS must calculate the required number of pilot symbols $P$ for each estimation instance. This task is performed by the module PCC, see Fig. 7, that computes $P$ dynamically based on three key categories of input information:

- **Historical information:** The BS receives CSI feedback and is assumed to maintain a buffer containing the most recent $M$ channel estimates $\bar{\mathbf{G}}$, together with their corresponding quality indicators $\bar{\mathbf{Q}}$.
- **Resource allocation:** In OFDMA systems, the BS assigns system resources to each user through the matrix $\mathbf{C}$, which is then conveyed to the UE via the Downlink Control Information (DCI) [41].
- **Target accuracy $\gamma$:** The desired estimation quality, expressed as an NMSE threshold, is defined at the BS according to the system's performance requirements.

Fig. 8 illustrates the neural network architecture used in the PCC module. As shown, the module operates in two functions. First function $\mathcal{W}(\cdot)$ analyzes $\bar{\mathbf{G}}$, $\bar{\mathbf{Q}}$, and $\mathbf{C}$ to produce a weight $\omega$, detailed in Section II-D. Second function $\mathcal{P}(\cdot)$ takes $\omega$ together with the target accuracy $\gamma$ and the number of allocated subcarriers $K$ to determine the required number of pilots $P$. The value of $K$ is obtained by counting the active entries across the subcarriers in $\mathbf{C}$, as $K = \|\mathbf{C}[:,0]\|_0$. The complete process of PCC module is expressed as:

$$P = \mathcal{P}(\mathcal{W}(\bar{\mathbf{G}}, \bar{\mathbf{Q}}, \mathbf{C}), \gamma, K). \tag{16}$$

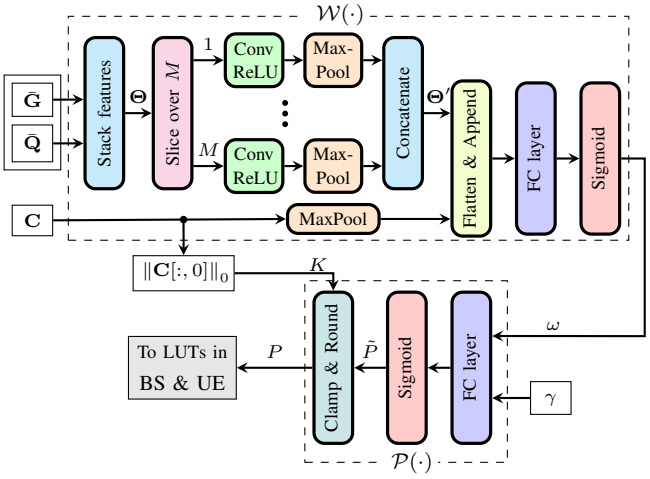


**FIGURE 8. Architecture of pilot count calculator. It first computes the relevance weight from historical information ($\bar{\mathbf{G}}$, $\bar{\mathbf{Q}}$) and the future resource allocation $\mathbf{C}$, then combines this weight with the target accuracy $\gamma$ and the number of allocated subcarriers $K$ to output the pilot count.**

The computation of the relevance weight $\omega$ begins by concatenating the matrices of historical information $\bar{\mathbf{G}}$ and $\bar{\mathbf{Q}}$, ensuring that the reliability, represented by the CQI values, of each past estimate is explicitly accounted for. The resulting tensor $\mathbf{\Theta} \in \mathbb{C}^{3\times F\times N\times M}$ stacks the real and imaginary parts of the channel together with the CQI information along the feature dimension. This tensor is then sliced along the temporal dimension into $M$ sub-matrices. Each temporal slice is independently processed by a convolutional layer with ReLU activation, followed by a MaxPooling operation. The convolutional layers extract local spatial–frequency patterns across subcarriers and antennas, enabling the network to capture correlations in the channel structure. The convolution parameters are chosen such that the output feature maps preserve the input spatial dimensions. To reduce computational complexity while retaining resource-level information, the subsequent MaxPooling operation reduces only the frequency dimension from $F$ to $S$, where $S$ is the number of RBGs, each contains $F_{\text{rbg}}$ subcarriers. This is achieved by using a pooling kernel and stride of $(F_{\text{rbg}}, 1)$, effectively aggregating information within each RBG. The MaxPooling outputs from all $M$ time instances are then concatenated, yielding a reduced-dimension tensor $\mathbf{\Theta}' \in \mathbb{C}^{3\times S\times N\times M}$. The same MaxPool operation with the same parameters is applied to the resource allocation matrix $\mathbf{C} \in \mathbb{C}^{F\times N}$ to reduce its dimensionality to $\mathbf{C}' \in \mathbb{C}^{S\times N}$. This application does not change the main information carried in binary $\mathbf{C}$ since the allocated RBGs are indicated with ones along all subcarriers of these groups so in $\mathbf{C}'$ each one indicate one allocated RBG. The tensors $\mathbf{\Theta}'$ and $\mathbf{C}'$ are then flattened, concatenated, and processed by a FC layer. This layer acts as a lightweight temporal aggregator that learns how trends in past channel quality interact with the upcoming resource allocation. Its output is passed through a sigmoid activation to produce the normalized relevance weight $\omega \in [0, 1]$.

To proceed with determining the required number of pilot symbols, the second function $\mathcal{P}(\cdot)$ receives $\omega$ together with the target accuracy $\gamma$. Both inputs are first passed through a FC layer, which jointly interprets their influence on the pilot requirement. This is followed by a sigmoid activation that produces a value $\tilde{P} \in [0, 1]$, where $\tilde{P}$, representing the fraction of the number of pilots required for the current estimation. Finally, using the number of allocated subcarriers $K$, a Clamp & Round layer converts $\tilde{P}$ into the actual number of pilots $P$. This two-steps design $\mathcal{W}(\cdot) \rightarrow \mathcal{P}(\cdot)$ improves modularity: the first function summarizes the past channel relevance, while the second maps this summary to required pilots under given target accuracy.

Once the BS determines $P$, it is transmitted to the UE, and the corresponding pilot pattern $\mathbf{L}$ is obtained from a predefined LUT at both ends, as shown in Fig. 7. The LUT is constructed offline and identically available at the BS and UE, enabling synchronized pilot pattern selection without explicitly signaling $\mathbf{L}$. While both LUTs store the same pilot patterns, the UE-side LUT additionally associates each $\mathbf{L}$ with a dedicated channel estimator $\mathcal{Q}(\cdot)$. During operation, the BS and UE access the LUT using the same inputs, namely the pilot count $P$ and the resource allocation $\mathbf{C}$, ensuring consistent pilot placement and estimator selection. The offline LUT construction incurs a one-time computational cost, after which all entries are reused without further updates, enabling low-complexity operation. Formally, the LUT implements the mapping $(\mathbf{L}, \mathcal{Q}(\cdot)) = \mathcal{L}(P, \mathbf{C})$.

#### 2) BRANCH_1: PILOT-BASED ESTIMATION

At the UE, the pilot-based estimation branch operates by first identifying the pilot configuration using the received pilot count $P$ and the resource allocation matrix $\mathbf{C}$, which jointly index the corresponding pilot pattern $\mathbf{L}$ and the decoder $\mathcal{Q}(\cdot)$ from the UE-side LUT. Accordingly, the UE extracts the received pilot symbols $\mathbf{Y}_{\mathrm{p}}$ from the signal $\mathbf{Y}$ using the pilot extraction function $\mathbf{Y}_{\mathrm{p}} = \mathcal{X}^{-1}(\mathbf{Y}, \mathbf{L})$. The channel estimator $\mathcal{R}(\cdot)$ subsequently processes the initial estimate $\mathcal{Q}(\mathbf{Y}_{\mathrm{p}})$ along with the resource allocation $\mathbf{C}$ to generate the final pilot-based channel estimation estimation of Branch_1, $\hat{\mathbf{H}}_1 = \mathcal{R}(\mathcal{Q}(\mathbf{Y}_{\mathrm{p}}), \mathbf{C})$. This procedure constitutes the pilot-based channel estimation component of the SBCE-APD framework, as depicted in Fig. 7.

#### 3) BRANCH_2: NON-PILOT-BASED ESTIMATION

Branch_2 operates in parallel with Branch_1 and performs channel estimation without relying on pilot symbols. As shown in Fig. 7, this branch employs a pre-trained GAN generator $\mathcal{G}(\cdot)$ based on the STN-GAN architecture illustrated in Fig. 4. The generator takes as input user-specific historical channel information $\bar{\mathbf{G}}$ and $\bar{\mathbf{Q}}$, together with the resource allocation matrix $\mathbf{C}$. The resulting channel prediction is given by $\hat{\mathbf{H}}_2 = \mathcal{G}(\bar{\mathbf{G}}, \bar{\mathbf{Q}}, \mathbf{C})$. This branch is fully data-driven without relying on pilot symbols. Its estimation performance depends on the relevance of the historical channel information, which is quantified at the BS as the weight $\omega$ used for calculating $P$. Although $\omega$ is not explicitly modeled within the GAN itself, its effect is implicitly captured through the pilot count $P$ determined by the PCC. In particular, higher prediction reliability of STN-GAN generator corresponds to a reduced pilot requirement, whereas lower reliability necessitates additional pilots.

### *B. TRAINING PROCESSES*

As the SBCE-APD framework comprises multiple ML-based components, all of which are trained in a supervised manner, the training is performed in a staged approach. Initially, the STN-GAN network is trained independently to obtain the generator $\mathcal{G}(\cdot)$. Simultaneously, the LUT $\mathcal{L}(\cdot)$ is populated offline for all possible pilot patterns $\mathbf{L}$, with each entry associated with its corresponding decoder $\mathcal{Q}(\cdot)$. Once the generator and the LUT are finalized, both are fixed and used to train the functions $\mathcal{W}(\cdot)$ and $\mathcal{P}(\cdot)$ of the PCC module as well as the fusion function $\mathcal{M}(\cdot)$ of MLP layer.

#### 1) STN-GAN STRUCTURE AND TRAINING

The architectures of both the generator and discriminator are derived from the framework proposed in [23], and an overview of the overall GAN design is presented in Fig. 4.

*Training process:* The STN-GAN is trained using a hybrid objective combining the WGAN-GP adversarial loss and a perceptual feature loss [23]. The WGAN-GP formulation enforces a soft Lipschitz constraint on the critic, providing smooth gradients and stabilizing training, while the perceptual loss encourages structural fidelity in the reconstructed channels. The overall objective is

$$\min_{\mathcal{G}} \max_{\mathcal{D}} \ \big(\mathcal{L}_{\text{adv}}(\mathcal{G}, \mathcal{D}) + \eta\, \mathcal{L}_{\text{pct}}(\mathcal{G})\big), \tag{17}$$

where $\eta = 10$ balances the two terms. Training uses the Adam optimizer with $(\beta_1, \beta_2) = (0.5, 0.99)$, learning rate $10^{-4}$ and batch size 16. During training, we use the dataset $\mathbf{S}_1 = \{(\bar{\mathbf{G}}_r^{\mathrm{S1}}, \bar{\mathbf{Q}}_r^{\mathrm{S1}}, \mathbf{C}_r^{\mathrm{S1}}, \mathbf{H}_r^{\mathrm{S1}})\}_{r=1}^{R_1}$, where $R_1$ denotes the number of realizations. Each tuple contains the historical information $\bar{\mathbf{G}}$ and $\bar{\mathbf{Q}}$, the resource allocation $\mathbf{C}$, and the true channel realization $\mathbf{H}$. The generator produces an estimate $\hat{\mathbf{H}}_2$, which is evaluated against the real channel $\mathbf{H}$ through the adversarial and perceptual losses described earlier. The training workflow is summarized in Algorithm 1.

**Algorithm 1:** Pseudocode for Training STN-GAN

1: **Input:** $\mathbf{S}_1 = \{(\bar{\mathbf{G}}_r^{\mathrm{S1}}, \bar{\mathbf{Q}}_r^{\mathrm{S1}}, \mathbf{C}_r^{\mathrm{S1}}, \mathbf{H}_r^{\mathrm{S1}})\}_{r=1}^{R_1}$.
2: **for** each training batch from $\mathbf{S}_1$ **do**
3: Estimate channel: $\hat{\mathbf{H}}_2 = \mathcal{G}(\bar{\mathbf{G}}, \bar{\mathbf{Q}}, \mathbf{C})$.
4: Take only the channel on allocated resources $\hat{\mathbf{H}}_2[\mathbf{C} = 1]$
5: Classify using discriminator: $\mathcal{D}(\hat{\mathbf{H}}_2)$ as True or Fake based on real channels $\mathbf{H}$.
6: Update $\mathcal{G}(\cdot)$ and $\mathcal{D}(\cdot)$ parameters.
7: **end for**
8: **Output:** Trained generator $\mathcal{G}(\cdot)$.

### 2) PILOT PATTERN DESIGN AND LUT CONSTRUCTION

The objective of pilot pattern design is to place a limited number of pilot symbols on the most informative subcarriers for channel estimation. Since subcarriers contribute unequally to channel observability, selecting pilot locations based on underlying channel properties can significantly improve estimation performance. To this end, we employ feature-selection technique [42] to identify pilot positions that capture the most relevant channel information. This approach is particularly effective in sparse wireless channels [43], where the impulse response is dominated by a small number of significant multipath components [44]. By focusing pilots on these informative positions, pilot overhead is reduced while maintaining high estimation accuracy.

Building on this principle, the ConcAE, introduced in Sections IV–B, is employed for joint pilot selection and channel estimation. The ConcAE is well suited to this task because its encoder performs data-driven feature selection to identify informative pilot subcarriers, while its decoder learns a channel estimator tailored to the selected pilots, enabling end-to-end optimization. The ConcAE model is trained with a learning rate of $10^{-3}$ and a batch size of 256. After training, the selector layer determines the pilot pattern $\mathbf{L}$, and the decoder $\mathcal{Q}(\cdot)$ is used for channel estimation [21].

Generally, since the pilot configuration depends jointly on the allocated bandwidth, specified by the resource allocation matrix $\mathbf{C}$, and the pilot count $P$, the LUT is constructed offline to cover all $(P, \mathbf{C})$ combination scenarios. For each scenario, the LUT entry stores the corresponding pilot pattern $\mathbf{L}$ together with its associated decoder $\mathcal{Q}(\cdot)$. This offline design enables near-optimal pilot pattern and estimator selection during operation with negligible online computational complexity.

However, for wideband allocations and/or large pilot counts $P$, designing pilot patterns for the entire allocated bandwidth leads to a combinatorial growth of the LUT size. To mitigate this issue, we exploit the fact that resource allocation is performed at the RBG level, such that the allocated bandwidth consists of an integer number of RBGs. Based on this observation, pilot patterns can be designed for a single RBG and then uniformly replicated across all RBGs allocated to a given user. Fig. 9 illustrates this strategy and contrasts it with joint pilot design across multiple RBGs.

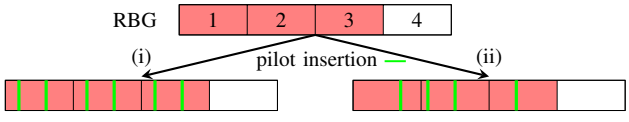


**FIGURE 9. Example pilot pattern designs at the RBG level: (i) one pilot pattern is defined for a single RBG and repeated identically across all RBGs; (ii) several adjacent RBGs are grouped and assigned coordinated pilot patterns within the group. Green bars mark pilot positions.**

As an illustrative example, consider a scenario in which the number of allocated RBGs $S'$ ranges in $[1, \ldots, 4]$ and the pilot count satisfies $P \in [1, \ldots, 8]$. Joint pilot design across all allocated RBGs would result in a LUT size of $8 \times (2^4 - 1) = 120$ entries, according to polynomial theory. In contrast, under the single-RBG design strategy, the maximum number of pilots per RBG is limited to $8/4 = 2$, and the LUT requires only $2 \times (2^1 - 1) = 2$ entries per RBG, yielding a substantial reduction in LUT complexity.

This simplification introduces a trade-off between LUT complexity and pilot efficiency. Following [42], the number of pilots required for a bandwidth of $F'$ subcarriers and channel sparsity $\rho$ scales as:

$$P \;\propto\; \rho \cdot \log\left(\frac{F'}{\rho}\right), \tag{18}$$

indicating that pilot requirements increase logarithmically with bandwidth. Consequently, repeating a single-RBG pilot pattern across all allocated RBGs may require a larger total number of pilots to achieve the same estimation accuracy as a joint wideband design. Nevertheless, this approach offers a favorable trade-off by significantly reducing LUT size while maintaining robust estimation performance.

*LUT construction*: Following the above pilot design principle, the LUT is constructed to cover all relevant configurations of pilot count $P$ and resource allocation $\mathbf{C}$. Let the training dataset be $\mathbf{S}_2 = \{\mathbf{h}^{\mathrm{S2}}\}_{r=1}^{R_2}$, where each $\mathbf{h} \in \mathbb{C}^K$ represents a channel realization and $R_2$ is the total number of observations. The design space of pilot patterns combines $D$ resource allocation configurations, $[\mathbf{C}_1, \ldots, \mathbf{C}_D]$, with $P_{\max}$ possible pilot counts, $[1, \ldots, P_{\max}]$. For each combination of $P$ and $\mathbf{C}$, a ConcAE network is trained independently using $\mathbf{S}_2$, and the resulting optimal pilot positions $\mathbf{L}$ along with the corresponding channel reconstruction function $\mathcal{Q}(\cdot)$ are stored in the LUT. Since the entire LUT is generated offline, it imposes no computational burden during real-time operation. Although the LUTs at the BS and UE are identical, the BS does not need to store the channel estimators, as channel reconstruction is performed solely at the UE. The LUT construction process is summarized in Algorithm 2.

**Algorithm 2:** Pseudocode for Generating the LUT

```
Initialize: Empty LUT = [ ]
Input: S_2 = {h_r^{S2}}_{r=1}^{R_2}, [1, ..., P_max],
  and [C_1, ..., C_D].
for P = 1 to P_max do
    for d = 1 to D do
        for each training batch from S_2 do
            Train ConcAE to Update L_{P,d} and Q(·)_{P,d}
        end for
        Insert L_{P,d} and Q(·)_{P,d} to LUT corresponding to
        the entry P and C_d.
    end for
end for
Output: Filled LUT L(·).
```

### 3) TRAINING OF THE PCC AND MLP NETWORKS

For the final training stage, the pre-trained STN-GAN $\mathcal{G}(\cdot)$ generator and the pre-defined LUT $\mathcal{L}(\cdot)$ are used. Consequently, only the parameters of the PCC and the MLP fusion layer are updated via backpropagation, corresponding to training the functions $\mathcal{W}(\cdot)$, $\mathcal{P}(\cdot)$, and $\mathcal{M}(\cdot)$.

The PCC is trained to determine the minimum number of pilot symbols required to satisfy a target channel estimation accuracy $\gamma$ with minimal pilot overhead, while the MLP is trained to fuse the channel estimates from Branch_1 and Branch_2. Training the PCC and MLP jointly is essential, as the pilot count selected by the PCC must ensure that the final channel estimate $\hat{\mathbf{H}}$, meets the target NMSE constraint $\gamma$ with respect to the true channel $\mathbf{H}$. Joint optimization allows the PCC to account for the reliability of the fusion process and enables consistent end-to-end minimization of the NMSE under the pilot overhead constraint.

To enable stable and differentiable training, the PCC predicts a continuous pilot fraction $\tilde{P} \in [0, 1]$ (see Fig. 8), representing the relative pilot budget required under the current channel and resource allocation conditions. This continuous formulation is used only during training, while the final discrete pilot count $P$ is obtained via rounding and clamping at inference. The training objective enforces the accuracy constraint while discouraging excessive pilot usage, as defined by the loss function:

$$\text{Loss} = \max(0, \text{NMSE} - \gamma)^2 + \lambda \tilde{P}, \tag{19}$$

where the first term penalizes violations of the target accuracy $\gamma$ and the second term promotes pilot efficiency. The squared penalty ensures smooth gradients when the estimation error exceeds $\gamma$. The regularization parameter $\lambda$ controls the trade-off between estimation accuracy and pilot overhead. If $\lambda$ is set too small, the model tends to allocate an excessive number of pilots, resulting in inefficient resource usage. Conversely, an overly large $\lambda$ encourages aggressive pilot reduction, which may lead to violations of the accuracy constraint. A properly chosen $\lambda$ enables adaptive pilot allocation by balancing estimation performance and resource efficiency. To detect the optimal $\lambda$, we use grid search on a validation set: we train the model with a range of $\lambda$ values and evaluate the resulting NMSE and pilot count $P$, and select the $\lambda$ that best balances accuracy (NMSE $\leq \gamma$) and pilot efficiency.

Training stability and generalization are enhanced by applying batch normalization after convolutional layers in $\mathcal{W}(\cdot)$ and bounding intermediate outputs using sigmoid activations, ensuring that the relevance weight $\omega$ and pilot fraction $\tilde{P}$ remain within valid ranges. Channel quality variations are implicitly captured through the CQI included in the historical information. The PCC is trained using the Adam optimizer with an initial learning rate of $10^{-3}$, a batch size of 64, and $\ell_2$ weight decay for regularization. During training, we use the dataset $\mathbf{S}_3 = \{(\bar{\mathbf{G}}_r^{\text{S3}}, \bar{\mathbf{Q}}_r^{\text{S3}}, \mathbf{C}_r^{\text{S3}}, \gamma_r^{\text{S3}}, \mathbf{H}_r^{\text{S3}})\}_{r=1}^{R_3}$, where $R_3$ denotes the number of realizations. Each tuple contains the historical information $\bar{\mathbf{G}}$ and $\bar{\mathbf{Q}}$, the resource allocation $\mathbf{C}$, the true channel $\mathbf{H}$, and the target NMSE $\gamma$. A candidate pilot set $\mathscr{P}$ is provided, from which the pilot symbols are selected. The functions $\mathcal{X}(\cdot)$ and $\mathcal{X}^{-1}(\cdot)$ handle pilot insertion and extraction, while $\mathcal{R}(\cdot)$ maps the estimated channel coefficients to their positions across the full OFDMA bandwidth according to $\mathbf{C}$. The final fused channel estimate $\hat{\mathbf{H}}$ is evaluated against $\mathbf{H}$ to compute the NMSE. The training workflow is summarized in Algorithm 3.

**Algorithm 3:** Pseudocode for Training PCC & MLP

**Input:** $\mathbf{S}_3 = \{(\bar{\mathbf{G}}_r^{\text{S3}}, \bar{\mathbf{Q}}_r^{\text{S3}}, \mathbf{C}_r^{\text{S3}}, \gamma_r^{\text{S3}}, \mathbf{H}_r^{\text{S3}})\}_{r=1}^{R_3}$, candidate pilot set, trained $\mathcal{L}(\cdot)$, and trained $\mathcal{G}(\cdot)$
**for** each training batch from $\mathbf{S}_1$ **do**
  Compute weight $\omega = \mathcal{W}(\bar{\mathbf{G}}, \bar{\mathbf{Q}}, \mathbf{C})$.
  Determine $K = \|\mathbf{C}[:, 0]\|_0$
  Calculate $P = \mathcal{P}(\omega, \gamma, K)$.
  Obtain the pilot pattern $(\mathbf{L}, \_) = \mathcal{L}(P, \mathbf{C})$.
  Using $P$, select $\mathbf{X}_\text{p}$ from the pilot set $\mathscr{P}$.
  Insert the pilots in the transmitted signal $\mathbf{X} = \mathcal{X}(\mathbf{X}_\text{p}, \mathbf{L})$
  Obtain $\mathbf{Y} = \mathbf{H}\mathbf{X} + \mathbf{Z}$, with $\mathbf{Z}$ is noise matrix
  Extract the received pilots $\mathbf{Y}_\text{p} = \mathcal{X}^{-1}(\mathbf{Y}, \mathbf{L})$
  Branch_1 estimation $\hat{\mathbf{H}}_1 = \mathcal{R}(\mathcal{Q}(\mathbf{Y}_\text{p}), \mathbf{C})$.
  Branch_2 estimation $\hat{\mathbf{H}}_2 = \mathcal{G}(\bar{\mathbf{G}}, \bar{\mathbf{Q}}, \mathbf{C})$.
  Final channel estimation $\hat{\mathbf{H}} = \mathcal{M}(\hat{\mathbf{H}}_1, \hat{\mathbf{H}}_2, P)$.
  Compute loss from (19)
  Update $\mathcal{P}(\cdot)$ and $\mathcal{W}(\cdot)$ via backpropagation.
**end for**
**Output:** Trained functions $\mathcal{P}(\cdot)$, $\mathcal{W}(\cdot)$, and $\mathcal{M}(\cdot)$.

### 4) TRAINING OF CHANNELFORMER

Channelformer is trained following the procedure described in its original work [38]. The network is trained in a supervised manner using LS channel estimates as input features, see Fig. 6, where the real and imaginary parts are treated as separate input channels. While Channelformer operates on time–frequency channel representations, we consider spatial–frequency representations to align the comparison with the proposed SBCE-AFD framework. Offline training exploits the availability of noise-free channel responses as regression labels, enabling the model to learn denoising and interpolation across the frequency and spatial domains. The Adam optimizer is used with an initial learning rate of $2 \times 10^{-3}$, reduced by a factor of $0.5$ at predefined epochs, and a mini-batch size of $128$ for up to 100 training epochs. The Huber loss is adopted to balance convergence speed and robustness to outliers, while light L2 regularization is applied and dropout is omitted, as channel noise provides inherent regularization. During evaluation, the trained parameters are fixed and no retraining or fine-tuning is performed.

### C. ADAPTIVE PILOT ALLOCATION THEORY

From (10), the LS estimation error is governed by the conditioning of the pilot observation matrix $\mathbf{X}_{\mathrm{p}}$. For practical pilot patterns, the trace term decreases inversely with the number of pilots $P$, yielding the well-known scaling $\mathrm{NMSE} \propto \frac{\sigma_z^2}{P}$. In the proposed framework, channel prediction reduces the estimation uncertainty. As described in Sections II-D, the weight $\omega \in [0,1]$ quantifies channel predictability by capturing temporal, frequency, and spatial correlations between past and current channel states. Consequently, $P$ is reduced by a factor $(1-\omega)$, leading to $\mathrm{NMSE} \propto \frac{\sigma_z^2(1-\omega)}{P}$. For a target NMSE $\gamma$, the required pilot count becomes

$$P \propto \frac{\sigma_z^2(1-\omega)}{\gamma}. \tag{20}$$

This provides the analytical justification of adaptive pilot allocation in time-varying MIMO-OFDM systems: the framework estimates channel predictability through $\omega$ and allocates $P$ necessary to satisfy the NMSE constraint $\gamma$.

## VI. SYSTEM COMPLEXITY ANALYSIS

This section presents an analysis of the computational complexity of the components within the proposed approach, as well as a summary of their memory consumption.

### A. COMPUTATIONAL COMPLEXITY

The computational complexity of the SBCE-APD framework can be divided into offline training and online deployment phases. The offline phase includes the training of the STN-GAN, ConcAE, PCC, and MLP modules, which is computationally intensive but performed only once. In contrast, the online phase consists of forward inference through the trained components, including the ConcAE decoder, the STN-GAN generator, LUT access, and the MLP fusion module. These operations involve only feedforward computations without iterative optimization or retraining, resulting in a fixed and predictable computational cost during deployment. The online computational complexity of all components, expressed in Big-O notation, is summarized in Table 4.

**TABLE 4. Computational Complexity of SBCE-APD**

| Process | Big-O |
|---|---|
| STN-GAN generator | $\mathcal{O}(MFNC_{\mathrm{in}}C_{\mathrm{out}}\kappa^3)$ |
| ConcAE decoder | $\mathcal{O}(NP+NK)$ |
| PCC | $\mathcal{O}(MFNC_{\mathrm{in}}C_{\mathrm{out}}\kappa^2)$ |
| MLP | $\mathcal{O}(2NF+NK)$ |

with $C_{\mathrm{in}}$ and $C_{\mathrm{out}}$ are the number of input and output channels for Conv layers of the intended module, respectively. $\kappa$ is the kernel size. For the offline training, the GAN discriminator complexity is $\mathcal{O}(KNC_{\mathrm{in}}C_{\mathrm{out}}\kappa^2)$, and the ConcAE selector is $\mathcal{O}(NK)$.

From a practical perspective, online processing at the BS is primarily confined to PCC inference and LUT retrieval. At the user equipment, LUT lookup and the ConcAE decoder are performed in parallel with the STN-GAN generator, while the subsequent execution of the MLP fusion network introduces comparatively lower computational complexity.

### B. MEMORY CONSUMPTION

Both the BS and the UE are required to store a subset of trained components to support the SBCE-APD framework, as illustrated in Fig. 7, while exhibiting different memory requirements. At the BS, this includes the parameters of the PCC functions $\mathcal{W}(\cdot)$ and $\mathcal{P}(\cdot)$, as well as a LUT that stores only the pilot pattern $\mathbf{L}$. In contrast, the UE maintains a larger LUT, as it additionally stores the parameters of the corresponding decoder functions $\mathcal{Q}(\cdot)$ associated with each $\mathbf{L}$. The UE further stores the parameters of the STN-GAN generator $\mathcal{G}(\cdot)$ and the MLP module $\mathcal{M}(\cdot)$. For both BS and UE, the pilot pattern $\mathbf{L}$ is represented as a binary vector of length $K$ under the single-stream transmission assumption. Moreover, both ends store historical information in the form of the complex-valued matrix $\bar{\mathbf{G}}$ and the real-valued matrix $\bar{\mathbf{Q}}$. An approximate breakdown of the memory requirements for the different components is summarized in Table 5.

**TABLE 5. Memory consumption of the trained SBCE-APD components**

| Component | Memory Consumption [$\times 8$ bits] |
|---|---|
| GAN generator | $M(\kappa^3 NF(M+1))$ |
| PCC Module | $N(M+1)(\kappa^3+F)$ |
| LUT (Estimator $\mathcal{Q}(\cdot)$) | $PN_1^{\mathcal{Q}} + N_1^{\mathcal{Q}}N_2^{\mathcal{Q}} + N_2^{\mathcal{Q}}S'K$ |
| MLP | $2KN_1^{\mathrm{MLP}} + N_1^{\mathrm{MLP}}N_2^{\mathrm{MLP}} + N_2^{\mathrm{MLP}}S'K$ |
| Historical information | $3FNM$ |

with $N_1^{\mathcal{Q}}, N_2^{\mathcal{Q}}$ represent the neurons in the first and second layers in the ConcAE Decoder, respectively, and $N_1^{\mathrm{MLP}}, N_2^{\mathrm{MLP}}$ in the first and second MLP layers.

## VII. SIMULATION SETUP AND RESULTS

In this section, we conduct numerical simulations to assess the effectiveness of the proposed approach and to compare its performance against channel estimation schemes that rely solely on pilots or only on channel prediction.

### A. SIMULATION SETUP

For the simulations, we utilize Sionna [45], an open-source library designed for simulating the physical layer of wireless communication systems, to generate the channel coefficients. The system parameters used are detailed in Table 6.

**TABLE 6. Simulation Parameters**

| System Parameters | Value |
|---|---|
| Channel model | TDL [46] |
| Carrier frequency $f_c$ | 2.5 GHz |
| UE velocity $v$ range | 60 to 130 Km/h |
| Delay spread $\Delta_{\mathrm{ds}}$ range | 100 to 300 ns |
| Subcarrier spacing | 15 KHz |
| Number of Subcarriers $F$, $F_{\mathrm{rbg}}$ | 128, 32 |
| Number of RBG $S = F/F_{\mathrm{rbg}}$ | 4 |
| Number of BS and UE antennas $N_{\mathrm{t}}$, $N_{\mathrm{r}}$ | 8, 2 |
| Explicit CSI Feedback | perfect |
| Channel noise | AWGN |

Based on the system parameters in Table 6, the channel correlation time derived from Section II-C lies in the range $d_{\text{tc}} \in [1.6, 3.6]$ ms. With a subcarrier spacing of 15 KHz, each OFDM symbol spans $66.7\,\mu$s. Thus, the number of symbols is computed within the correlation time as $M = \lceil \frac{d_{\text{tc}}}{66.7 \cdot 10^{-6}} \rceil \in [24, 54]$. A representative value within this interval is selected, and $M$ is set to 36.
For the datasets $\mathbf{S}_1$, $\mathbf{S}_2$, and $\mathbf{S}_3$, the $R_1 = 10^5$, $R_2 = 2 \times 10^4$, and $R_3 = 2 \times 10^4$ realizations are used, respectively.

### B. SIMULATION RESULTS

In this evaluation, we focus on comparing the proposed ConcAE-based joint pilot design, as a part of SCBE-APD framework, and channel estimation method with a baseline that uses Uniformly Spaced Pilot (USP) positions combined with linear Interpolation (Lin-Int).

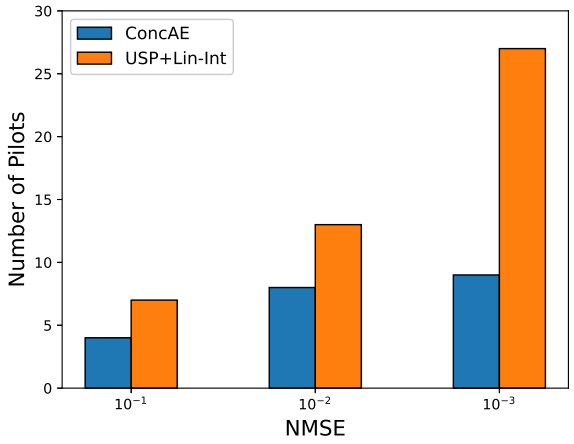


**FIGURE 10.** **Required number of pilots $P$ versus achieved NMSE in Branch_1, comparing ConcAE-based pilot design with uniform pilot spacing plus linear interpolation (USP+Lin-Int).**

Fig. 10 illustrates the required number of pilots $P$ per MIMO channel to achieve different levels of NMSE for the estimation process in Branch_1 only. It is important to note that the NMSE values reflect the performance of the intermediate output $\hat{\mathbf{H}}_1$, see Fig. 7. Compared with ConcAE-based joint pilot design and estimation, the baseline with uniformly spaced pilots and linear interpolation (USP+Lin-Int) needs substantially more pilots to reach the same NMSE, especially at low NMSE values. For instance, achieving NMSE levels of $10^{-1}, 10^{-2}$, and $10^{-3}$ requires 7, 13, and 27 pilots, respectively. In contrast, the ConcAE-based method achieves the same NMSE targets using only 4, 8, and 9 pilots, respectively. While a minor increase in the number of bits may be needed for higher-accuracy interpolation, the overall reduction in pilot overhead is substantial. This demonstrates that leveraging ConcAE for pilot design and channel estimation can significantly reduce pilot overhead while maintaining performance. Results are averaged over 100 independent Monte Carlo channel realizations at a fixed SNR of 30 dB.

We next evaluate the performance of SBCE-APD in Fig. 7, but with the No adaptive pilot design mechanism, denoted as SBCE-NoAPD. In this configuration, the weighting function $\mathcal{W}(\cdot)$ is retained to compute the historical relevance weight $\omega$, while the pilot control function $\mathcal{P}(\cdot)$ is deactivated. Instead, the number of pilots $P$ is manually varied to examine the behavior of the proposed method across different pilot values. For each selected value of $P$, a diverse set of historical information scenarios is considered, characterized by 1000 different realizations of $\mathbf{C}$, thereby ensuring coverage of the full range of the weight $\omega$. This evaluation setup is consistent with the illustrative example shown in Fig. 3.

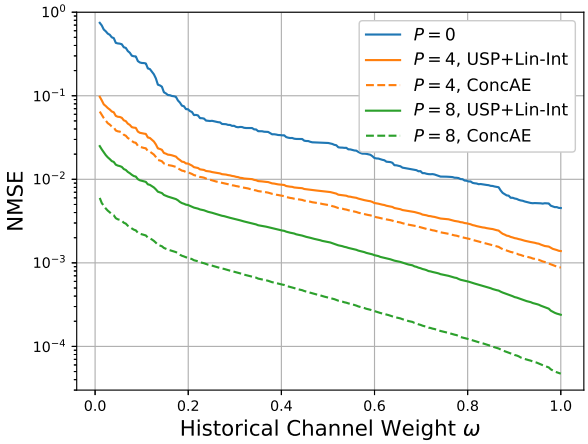


**FIGURE 11.** **NMSE of the proposed framework in terms of historical channel weights $\omega$. A comparison with the ConcAE and USP+Lin-Int with $P = 0, 4$, and 8.**

Fig. 11 presents the NMSE performance as a function of the historical channel weight $\omega$, as introduced in Section II-D. The plot includes results for three different pilot settings: $P = 0, 4$, and 8. When no pilots are used ($p = 0$), Branch_1 does not contribute in channel estimation, meaning the GAN generator in Branch_2 is solely responsible for the estimation. Whereas, for $P = 4$ and 8, we evaluate two types pilot pattern and estimator: ConcAE and USP+Lin-Int. The first key observation from the figure is that higher values of $\omega$ consistently result in lower NMSE across all cases. This is because the GAN generator in Branch_2 benefits from more informative historical input when $\omega$ is high. The results also show that increasing the number of pilots improves estimation accuracy. Moreover, the ConcAE-based approach consistently outperforms the USP+Lin-Int baseline, especially at higher pilot counts. The improvement observed at $P = 8$ is significantly greater than at $P = 4$, which aligns with the trends shown earlier in Fig. 10.
In the next analysis step, we evaluate the performance of the proposed framework with adaptive pilot design enabled. The system setup considers four users with equal resource allocation priority, such that each user is assigned, on average, 25% of the available RBGs over the past $M$ OFDM symbols. For each RBG previously allocated to a user,

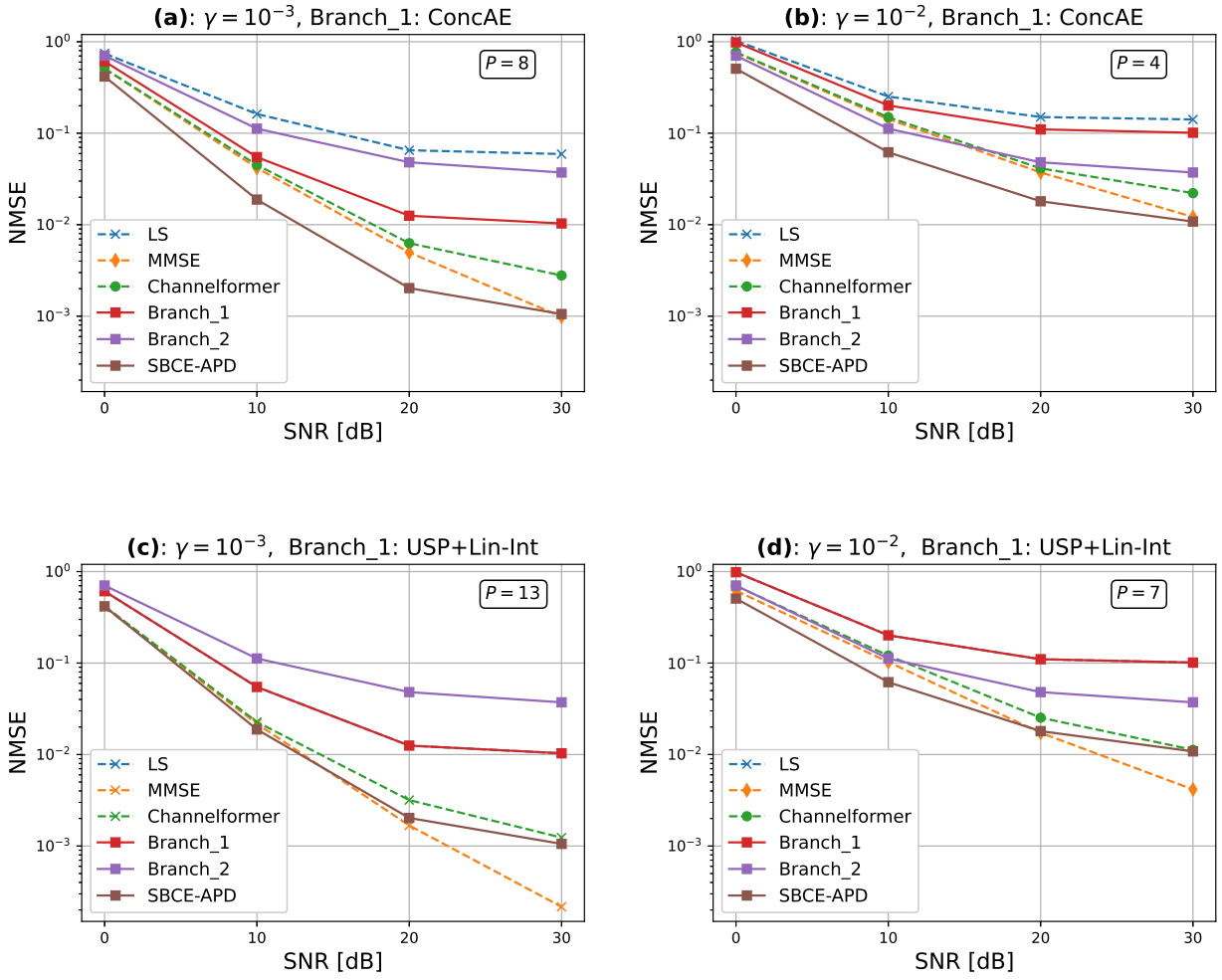


**FIGURE 12.** **NMSE versus SNR of the SBCE-APD method. For reference, the individual performances of Branch_1, Branch_2 of SCBE-APD are shown. For Benchmarking, the performance of the channel estimation methods LS, MMSE and also the attention-based Channelformer are illustrated. The figure consists of four subplots for different conditions. Two target accuracies are considered: $\gamma = 10^{-3}$ in (a,c) and $\gamma = 10^{-2}$ in (b,d). For Branch_1, two estimation approaches are evaluated: ConcAE in (a,b) and USP+Lin-Int in (c,d). Each subplot shows also the number of pilots $P$ determined by SBCE-APD to meet the target accuracy.**

the corresponding channel estimate and its associated CQI are computed at the UE and reported to the BS. During the initial phase of system operation, no historical channel information is available; therefore, channel estimation relies exclusively on pilot-based processing through Branch_1. Since the focus of this study is on steady-state behavior, performance evaluation is conducted only after the historical RBG buffers of all users have been sufficiently populated with channel estimates. From this point onward, Branch_2 contributes to the estimation process through its GAN-based generator, and the adaptive pilot design is activated and guided by the historical weight $\omega$.

Fig. 12 illustrates the NMSE performance of the proposed SBCE-APD method by evaluating the final channel estimate $\hat{\mathbf{H}}$ across different SNR levels. For each evaluation point, a fixed SNR value is assumed over the past $M$ channel realizations. To provide further insight into the estimation process, the intermediate outputs of Branch_1 and Branch_2, denoted by $\hat{\mathbf{H}}_1$ and $\hat{\mathbf{H}}_2$, respectively, are also reported. For benchmarking, SBCE-APD is compared with conventional LS and MMSE estimators, as well as the attention-based Channelformer method. Unlike SBCE-APD, these reference schemes do not exploit historical channel information and rely exclusively on pilot symbols arranged in a uniformly spaced pilot (USP) pattern. To ensure a fair comparison, LS, MMSE, and Channelformer are all configured with the same number of pilots as determined by SBCE-APD, while distributing them uniformly across the resource grid.

Four scenarios are evaluated based on two target NMSE values and the two approaches for pilot design and estimation:

- $\gamma = 10^{-3}$ in subfigures (a) and (c).
- $\gamma = 10^{-2}$ in subfigures (b) and (d).
- ConcAE in subfigures (a) and (b).
- USP+Lin-Int in subfigures (c) and (d).

Across all subfigures, we observe that Branch_2 exhibits consistent performance for all given scenarios. This is due to the fact that Branch_2 is a GAN-based generator that relies solely on historical information, without utilizing pilot information or being conditioned on the target NMSE.
Comparing subfigures (a) and (b), with Branch_1 is utilizing the ConcAE network in both scenarios but with different target NMSEs: $10^{-3}$ in (a) and $10^{-2}$ in (b). It is observed that Branch_1 shows better performance in (a) where the target NMSE is lower. This is because the lower target NMSE in (a) prompts the adaptive PCC module to allocate more pilots compared to (b). Specifically, scenario (a) uses $P = 8$ pilots, whereas scenario (b) uses only $P = 4$. Overall, the combined output of the two branches achieves the target NMSE successfully under stable channel conditions. For evaluating LS and MMSE, the same pilot count configured for SBCE-APD is applied, but with using the USP pilot pattern. As a result, both LS and MMSE demonstrate better performance in scenario (a) compared to scenario (b).
A similar discussion holds for subfigures (c) and (d), where Branch_1 now uses the USP+Lin-Int scheme. In this case, Branch_1 and LS yield identical results, as both methods share the same pilot pattern and estimation technique. The adaptive pilot allocation of SBCE-APD still ensures meeting the target NMSE, albeit with a higher number of pilots compared to the ConcAE-based scenarios in (a) and (b). Specifically, $P = 13$ pilots are used in (c) and $P = 7$ in (d). This increase is expected, as the USP+Lin-Int approach generally demands denser pilot placement to maintain estimation accuracy, consistent with results from Fig. 10.
Finally, the key strength of the proposed adaptive pilot allocation in SBCE-APD framework lies in its ability to flexibly meet NMSE targets using the minimum necessary pilot overhead. Regardless of the channel estimation method or pilot placement strategy used (ConcAE or USP+Lin-Int), the system consistently adjusts the number of pilot $P$ to ensure meeting a target estimation accuracy. This adaptability makes the approach especially attractive in varying channel conditions or when resources must be allocated dynamically across users in a multi-access environment.
An additional insight from Fig. 12 is that MMSE performs well, especially in (c) and (d), even surpassing SBCE-APD. This is because MMSE relies on accurate channel statistics, which are typically unavailable in practice, making it a theoretical reference. To evaluate the pilot reduction achieved by SBCE-APD, subfigure (a) shows that the adaptive design selects $P = 8$ pilots for a target accuracy of $\gamma = 10^{-3}$. With this pilot budget, SBCE-APD meets the target NMSE at an SNR of 30 dB, whereas LS and Channelformer fail to do so. For a fair comparison, we increased the number of pilots for the baseline methods until the target was satisfied. The results show that Channelformer requires 12 pilots and LS requires 22 pilots to achieve the same accuracy. This clearly demonstrates the pilot efficiency of SBCE-APD. Moreover, unlike SBCE-APD, the baseline methods do not natively adapt the pilot count to a target accuracy; instead, the required number of pilots must be determined manually through simulation.
All results of Fig. 12 are averaged over 250 independent Monte-Carlo realizations. The standard deviation of the NMSE remains below 1 dB for all evaluated scenarios, and the corresponding 95 % confidence intervals are omitted from the figures for clarity.
In the following, we evaluate the performance of the proposed SBCE-APD method and compare it to SBCE-NoAPD, where the adaptive mechanism is disabled. For Branch_1 we utilize the ConcAE topopulate the LUT. The analysis is based on 250 independent realizations, covering a diverse range of historical scenarios by varying $\mathbf{C}$ and $\mathbf{Q}$ ensuring broad coverage of the historical weight $\omega$. To focus solely on the impact of adaptivity, we assume a noise-free channel with no observation noise.

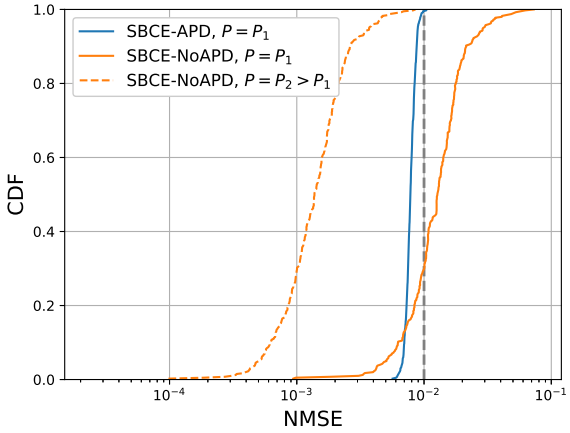


**FIGURE 13. CDF of NMSE for target NMSE $\gamma = 10^{-2}$ in three different scenarios: SBCE-APD with $P = P_1$, SBCE-NoAPD (non-adaptive) with $P = P_1$, and the SBCE-NoAPD with $P = P_2 > P_1$ to ensure NMSE requirement.**

Fig. 13 illustrates the Cumulative Distribution Function (CDF) of the achieved NMSE values under target NMSE of $\gamma = 10^{-2}$, comparing three variants of the proposed SBCE-APD estimation scheme:

- **SBCE-APD (adaptive)** uses the proposed adaptive pilot deployment. The CDF indicates that all realizations meet the NMSE requirement, with some achieving significantly lower values. This occurs particularly when the historical weight is high, prompting the system to bypass pilot usage for Branch_1 and rely solely on Branch_2 for estimation. This observation is consistent with the results shown in Fig. 11, where a high weight $\omega$ leads to improved estimation, with NMSE dropping below the threshold $10^{-2}$. The average number of pilots used is measured as $P = P_1 = 4$.

- **SBCE-NoAPD (non-adaptive)** employs a fixed number of pilots equal to the average from the adaptive approach, i.e., $P = P_1$. The CDF reveals a wide variation in NMSE outcomes, with about $70\%$ of estimations failing to meet the NMSE target. This broad performance range results from the diverse configurations of historical realizations utilized. The findings align with those in Fig. 11, where for $P = 4$ the NMSE varies between $10^{-1}$ and $10^{-3}$ as the weight $\omega$ ranges from $0$ to $1$. This underscores the limitation of relying on a fixed pilot count.
- **SBCE-NoAPD (non-adaptive, tuned pilot count)**: Here, the number of pilots is manually adjusted to ensure that all channel estimates meet the NMSE requirement. The average number of pilots used in this case is measured as $P = P_2 = 7$. As illustrated in the CDF, this configuration successfully satisfies the NMSE constraint, but at the cost of higher pilot overhead compared to the adaptive scheme.

Overall, the adaptive SBCE-APD scheme meets the NMSE target while minimizing pilot usage. Non-adaptive strategies either underperform or require significantly more pilots. The regularization parameter is set to $\lambda = 0.15$, balancing pilot efficiency and accuracy, as larger values reduce pilots but risk violating the NMSE target, and smaller values overly prioritize accuracy.

## VIII. LIMITATIONS AND PRACTICAL CONSIDERATIONS

The proposed SBCE-APD assumes perfect CSI feedback, which may be affected by practical impairments such as compression and quantization. Its performance also degrades at low SNR, where estimation becomes less reliable. In addition, the framework depends on representative training data, and mismatches between training and deployment environments may reduce performance. Finally, while LUT-based design enables low-complexity online operation, its offline construction introduces computational and storage trade-offs.

## IX. CONCLUSION

This paper introduced a Semi-Blind Channel Estimation with Adaptive Pilot Design (SBCE-APD) framework for MIMO-OFDMA systems, aimed at reducing pilot overhead while maintaining a given accurate requirement of channel estimation. The approach combines a GAN-based model that leverages historical channel information for non-pilot-based estimation with a pilot-based estimation adjusts both pilot count and positioning in response to estimation accuracy demands. This combination allows SBCE-APD to intelligently balance data-driven and pilot-based estimation, achieving target accuracy with fewer pilots. Notably, while the proposed framework involves multiple components, the ML-based models were deliberately kept simple to manage system complexity. Simulation results confirm that SBCE-APD consistently meets NMSE targets with significantly lower pilot symbols usage comparing to non-adaptive baselines, and performs robustly across a variety of channel and historical conditions. However, a limitation is that NMSE targets are met reliably only at moderate to high SNR.
Unlike LTE/NR, where pilot configurations are largely static, the proposed framework enables lightweight, learning-assisted adaptation of pilot resources while remaining compatible with existing OFDM-based physical layers, making it well suited for practical deployment in dynamic 6G systems. Overall, SBCE-APD offers a practical and scalable solution to pilot overhead challenges in MIMO-OFDMA systems. It represents a promising step toward intelligent, resource-efficient channel estimation strategies that align with the objectives of next-generation wireless networks.

## REFERENCES


[1] V. Dala Pegorara Souto, P. S. Dester, M. Soares Pereira Facina, D. Gomes Silva, F. A. P. de Figueiredo, G. Rodrigues de Lima Tejerina, J. C. Silveira Santos Filho, J. Silveira Ferreira, L. L. Mendes, R. D. Souza, *et al.*, "Emerging mimo technologies for 6g networks," *Sensors*, vol. 23, no. 4, p. 1921, 2023.

[2] M. S. J. Solaija, S. E. Zegrar, and H. Arslan, "Orthogonal frequency division multiplexing: The way forward for 6g physical layer design?," *IEEE Vehicular Technology Magazine*, vol. 19, no. 1, pp. 45–54, 2024.

[3] N. Saxena, E. Rastogi, and A. Rastogi, "6g use cases, requirements, and metrics," in *6G Mobile Wireless Networks*, pp. 7–24, Springer, 2021.

[4] X. Yin and X. Cheng, *Propagation channel characterization, parameter estimation, and modeling for wireless communications*. John Wiley & Sons, 2016.

[5] K. P. Bagadi and S. Das, "Mimo-ofdm channel estimation using pilot carries," *International Journal of computer applications*, vol. 2, no. 3, pp. 81–88, 2010.

[6] H. Harkat, P. Monteiro, A. Gameiro, F. Guiomar, and H. Farhana Thariq Ahmed, "A survey on mimo-ofdm systems: Review of recent trends," *Signals*, vol. 3, no. 2, pp. 359–395, 2022.

[7] F. Haddad, C. Bockelmann, and A. Dekorsy, "Adaptive pilot pattern design for ultra-low latency communication in beyond 5g and 6g systems," in *2024 IEEE 100th Vehicular Technology Conference (VTC2024-Fall)*, pp. 1–5, IEEE, 2024.

[8] A. S. Ahmed, M. M. Hamdi, M. S. Abood, A. M. Khaleel, M. Fathy, and S. H. Khaleefah, "Channel estimation using ls and mmse channel estimation techniques for mimo-ofdm systems," in *2022 international congress on human-computer interaction, optimization and robotic applications (HORA)*, pp. 1–6, IEEE, 2022.

[9] D. Donoho, "Compressed sensing," *IEEE Transactions on Information Theory*, vol. 52, no. 4, pp. 1289–1306, 2006.

[10] R. He, B. Ai, G. Wang, M. Yang, C. Huang, and Z. Zhong, "Wireless channel sparsity: Measurement, analysis, and exploitation in estimation," *IEEE Wireless Communications*, vol. 28, no. 4, pp. 113–119, 2021.

[11] F. Haddad, C. Bockelmann, and A. Dekorsy, "Channel estimation and pilot overhead reduction in ofdm systems using compressed sensing dynamic mode decomposition," *IEEE Communications Letters*, vol. 28, no. 5, pp. 1137–1140, 2024.

[12] H. A. Le, T. Van Chien, T. H. Nguyen, H. Choo, and V. D. Nguyen, "Machine learning-based 5g-and-beyond channel estimation for mimo-ofdm communication systems," *Sensors*, vol. 21, no. 14, p. 4861, 2021.

[13] B. Nithya, D. Brijesh, S. K. Kumar, and J. Pathmakarthik, "Pilot based channel estimation of ofdm systems using deep learning techniques," *International Journal of Information Technology*, vol. 15, no. 2, pp. 819–831, 2023.

[14] M. B. Mashhadi and D. Gündüz, "Pruning the pilots: Deep learning-based pilot design and channel estimation for mimo-ofdm systems," *IEEE Transactions on Wireless Communications*, vol. 20, no. 10, pp. 6315–6328, 2021.

[15] N. Van Huynh, J. Wang, H. Du, D. T. Hoang, D. Niyato, D. N. Nguyen, D. I. Kim, and K. B. Letaief, "Generative ai for physical layer communications: A survey," *IEEE Transactions on Cognitive Communications and Networking*, vol. 10, no. 3, pp. 706–728, 2024.

[16] A. F. Molisch, *Wireless communications: from fundamentals to beyond 5G*. John Wiley & Sons, 2022.

[17] T. Peken, G. Vanhoy, and T. Bose, "Blind channel estimation for massive mimo," *Analog Integrated Circuits and Signal Processing*, vol. 91, pp. 257–266, 2017.

[18] N. Shaik and P. K. Malik, "A comprehensive survey 5g wireless communication systems: open issues, research challenges, channel estimation, multi carrier modulation and 5g applications," *Multimedia Tools and Applications*, vol. 80, no. 19, pp. 28789–28827, 2021.

[19] M. R. Mehrabani, B. Abolhassani, and F. Haddadi, "Semi-blind channel estimation and pilot allocation for doubly-selective massive mimo-ofdm systems," *IEEE Access*, 2024.

[20] M. F. Balın, A. Abid, and J. Zou, "Concrete autoencoders: Differentiable feature selection and reconstruction," in *International conference on machine learning*, pp. 444–453, PMLR, 2019.

[21] M. Soltani, V. Pourahmadi, and H. Sheikhzadeh, "Pilot pattern design for deep learning-based channel estimation in ofdm systems," *IEEE Wireless Communications Letters*, vol. 9, no. 12, pp. 2173–2176, 2020.

[22] S. Feuerriegel, J. Hartmann, C. Janiesch, and P. Zschech, "Generative ai," *Business & Information Systems Engineering*, vol. 66, no. 1, pp. 111–126, 2024.

[23] Y. Wu, V. Singh, and A. Kapoor, "From image to video face inpainting: spatial-temporal nested gan (stn-gan) for usability recovery," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 2396–2405, 2020.

[24] C. Chen and X. Cheng, *Resource Allocation for OFDMA Systems*, p. 43–81. Springer International Publishing, June 2019.

[25] R. He and B. Ai, *Wireless Channel Measurement and Modeling in Mobile Communication Scenario: Theory and Application*. CRC Press, 2024.

[26] U. Karabulut, A. Awada, I. Viering, A. N. Barreto, and G. P. Fettweis, "Low complexity channel model for mobility investigations in 5g networks," in *2020 IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1–8, IEEE, 2020.

[27] Y. S. Cho, J. Kim, W. Y. Yang, and C. G. Kang, *MIMO Channel Models*, pp. 71–109. Singapore: Wiley, 2010.

[28] L. Wood and W. S. Hodgkiss, "Understanding the weichselberger model: A detailed investigation," in *MILCOM 2008-2008 IEEE Military Communications Conference*, pp. 1–7, IEEE, 2008.

[29] T. S. Rappaport, *Wireless communications: principles and practice*. Cambridge University Press, 2024.

[30] E. Björnson, C.-B. Chae, R. W. Heath Jr, T. L. Marzetta, A. Mezghani, L. Sanguinetti, F. Rusek, M. R. Castellanos, D. Jun, and Ö. T. Demir, "Towards 6g mimo: Massive spatial multiplexing, dense arrays, and interplay between electromagnetics and processing," *arXiv preprint arXiv:2401.02844*, 2024.

[31] "ETSI TS 138 214." https://www.etsi.org/deliver/etsi_ts/138200_138299/138214/17.01.00_60/ts_138214v170100p.pdf.

[32] R. Srividhya, T. Jayachandran, V. Rajmohan, *et al.*, "Channel estimation for ofdm systems using mmse and ls algorithms," in *2022 6th international conference on trends in electronics and informatics (ICOEI)*, pp. 1–5, IEEE, 2022.

[33] X. Dong, W.-S. Lu, and A. C. Soong, "Linear interpolation in pilot symbol assisted channel estimation for ofdm," *IEEE transactions on wireless communications*, vol. 6, no. 5, pp. 1910–1920, 2007.

[34] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *Communications of the ACM*, vol. 63, no. 11, pp. 139–144, 2020.

[35] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[36] H. Taud and J. Mas, *Multilayer Perceptron (MLP)*, pp. 451–455. Cham: Springer International Publishing, 2018.

[37] C. J. Maddison, A. Mnih, and Y. W. Teh, "The concrete distribution: A continuous relaxation of discrete random variables," *arXiv preprint arXiv:1611.00712*, 2016.

[38] D. Luan and J. S. Thompson, "Channelformer: Attention based neural solution for wireless channel estimation and effective online training," *IEEE Transactions on Wireless Communications*, vol. 22, no. 10, pp. 6562–6577, 2023.

[39] Z. Wei, D. Xu, S. Li, S. Song, D. W. K. Ng, and G. Caire, "Resource allocation design for next-generation multiple access: A tutorial overview," *Proceedings of the IEEE*, 2024.

[40] J. Guo, C.-K. Wen, S. Jin, and X. Li, "Ai for csi feedback enhancement in 5g-advanced," *IEEE Wireless Communications*, 2022.

[41] R. Mounika, A. Kumar, K. Kuchi, *et al.*, "Downlink resource allocation for 5g-nr massive mimo systems," in *2021 National Conference on Communications (NCC)*, pp. 1–6, IEEE, 2021.

[42] K. Manohar, B. W. Brunton, J. N. Kutz, and S. L. Brunton, "Data-driven sparse sensor placement for reconstruction: Demonstrating the benefits of exploiting known patterns," *IEEE Control Systems Magazine*, vol. 38, no. 3, pp. 63–86, 2018.

[43] F. Haddad, C. Bockelmann, and A. Dekorsy, "Optimized pilot pattern for sparse channel estimation: A low-complexity solution for future networks," in *GLOBECOM 2025 IEEE Global Communications Conference*, IEEE, 2024.

[44] A. Moslemi, "A tutorial-based survey on feature selection: Recent advancements on feature selection," *Engineering Applications of Artificial Intelligence*, vol. 126, p. 107136, 2023.

[45] J. Hoydis, S. Cammerer, F. Ait Aoudia, A. Vem, N. Binder, G. Marcus, and A. Keller, "Sionna: An open-source library for next-generation physical layer research," *arXiv preprint*, Mar. 2022.

[46] "ETSI TR 138 901." https://www.etsi.org/deliver/etsi_tr/138900_138999/138901/17.00.00_60/tr_138901v170000p.pdf.


**Fayad Haddad** (Graduate Student Member, IEEE) received his bachelor's degree in electronic and communication engineering from the University of Al-Baath, Homs, Syria, in 2008, and his master's degree in information and communication engineering from the University of Bremen, Germany, in 2019. He is currently a research assistant and has authored or coauthored multiple publications in flagship IEEE conferences and organized workshops. His research interests focus on channel estimation, pilot placement optimization and CSI compression and prediction for 5G and 6G networks.

**Carsten Bockelmann** (Member, IEEE) received Dipl.-Ing. and Ph.D. degrees in electrical engineering from the University of Bremen, Germany, in 2006 and 2012, respectively. Since 2012, he has been a Senior Research Group Leader with the University of Bremen, coordinating research activities regarding the application of various frameworks like compressive sensing, DMD, Event-based sampling, and machine learning to communication problems. His research interests include 6G, massive machine-type communication, ultra-reliable low latency communications and industry 4.0 as well as health communications.

**ARMIN DEKORSY** (Senior Member, IEEE) is currently the Head of the Department of Communications Engineering, University of Bremen. He has more than ten years of industrial experience in leading research positions, such as an DMTS with Bell Labs Europe and the Head of Research Europe Qualcomm, Nuremberg, and by conducting international research projects (more than 25 BMBF/BMWI/EU Projects) in affiliation with his scientific expertise shown by more than 200 journals and conference publications and holds more than 19 patents. He investigates new lines of research in wireless communication and signal processing for the baseband of transceivers of future communication systems, the results of which are transferred to the pre-development of industry through political and strategic activities. His current research focuses on distributed signal processing,compressed sampling, information bottleneck method, and machine learning leading to the further development of communication technologies for5G/6G, industrial wireless communications, and New Space satellite communications. He is a Senior Member of the IEEE Communications and Signal Processing Society, the Head of VDE/ITG Expert Committee "Information and System Theory", and a member of Executive Board of the Technologie-Zentrum Informatik und Informationstechnik, University of Bremen