Generative Diffusion Model Driven Massive Random Access in Massive MIMO Systems

Keke Ying, *Graduate Student Member, IEEE*, Zhen Gao, *Senior Member, IEEE*, Sheng Chen, *Life Fellow, IEEE*, Tony Q. S. Quek, *Fellow, IEEE*, and H. Vincent Poor, *Life Fellow, IEEE*

***Abstract*—Massive random access is an important technology for achieving ultra-massive connectivity in next-generation wireless communication systems. It aims to address key challenges during the initial access phase, including active user detection (AUD), channel estimation (CE), and data detection (DD). This paper examines massive access in massive multiple-input multiple-output (MIMO) systems, where deep learning is used to tackle the challenging AUD, CE, and DD functions. First, we introduce a Transformer-AUD scheme tailored for variable pilot-length access. This approach integrates pilot length information and a spatial correlation module into a Transformer-based detector, enabling a single model to generalize across various pilot lengths and antenna numbers. Next, we propose a generative diffusion model (GDM)-driven iterative CE and DD framework. The GDM employs a score function to capture the posterior distributions of massive MIMO channels and data symbols. Part of the score function is learned from the channel dataset via neural networks, while the remaining score component is derived in a closed form by applying the symbol prior constellation distribution and known transmission model. Utilizing these posterior scores, we design an asynchronous alternating CE and DD framework that employs a predictor-corrector sampling technique to iteratively generate channel estimation and data detection results during the reverse diffusion process. Simulation results demonstrate that our proposed approaches significantly outperform baseline methods with respect to AUD, CE, and DD.**

Received 11 May 2025; revised 23 September 2025; accepted 14 November 2025. Date of publication 4 December 2025; date of current version 22 December 2025. The work of Zhen Gao was supported in part by the Natural Science Foundation of China (NSFC) under Grant U2233216 and Grant 62471036, in part by Shandong Province Natural Science Foundation under Grant ZR2025QA30 and Grant ZR2022YQ62, in part by Beijing Natural Science Foundation under Grant L242011, and in part by Beijing Nova Program under Grant 20220484025. The work of Tony Q. S. Quek was supported in part by the National Research Foundation, Singapore and Infocomm Media Development Authority under its Communications and Connectivity Bridging Funding Initiative. The work of H. Vincent Poor was supported in part by the Innovation Grant from Princeton NextG. The associate editor coordinating the review of this article and approving it for publication was Z. Cai. *(Corresponding author: Zhen Gao.)*

Keke Ying is with the School of Information and Electronics, Beijing Institute of Technology (BIT), Beijing 100081, China (e-mail: ykk@bit.edu.cn).

Zhen Gao is with the State Key Laboratory of CNS/ATM, Beijing 100081, China, also with the MIIT Key Laboratory of Complex-Field Intelligent Sensing, Beijing 100081, China, also with BIT, Zhuhai 519088, China, also with the Advanced Technology Research Institute, BIT, Jinan 250307, China, and also with the Yangtze Delta Region Academy, BIT, Jiaxing 314019, China (e-mail: gaozhen16@bit.edu.cn).

Sheng Chen is with the School of Computer Science and Technology, Ocean University of China, Qingdao 266100, China (e-mail: sqc@ecs.soton.ac.uk).

Tony Q. S. Quek is with Singapore University of Technology and Design, Singapore 487372 (e-mail: tonyquek@sutd.edu.sg).

H. Vincent Poor is with the Department of Electrical and Computer Engineering, Princeton University, Princeton, NJ 08544 USA (e-mail: poor@princeton.edu).

Digital Object Identifier 10.1109/TWC.2025.3636585

***Index Terms*—Deep learning, massive MIMO, massive random access, transformer, generative models.**

I. Introduction

IN WIRELESS communications, multi-user random access is a critical step in establishing initial communication links. This process involves techniques such as activity user detection (AUD), channel estimation (CE), and multi-user data detection (DD). Traditional cellular networks often use scheduling-based access protocols and orthogonal multiple access methods [1]. As networks evolve, the 6G vision outlines more stringent performance requirements for scenarios involving massive communication. This vision necessitates a connection density ranging from 10^6 to 10^8 devices per square kilometer [2]. As the number of users increases, dependence on orthogonal multiple access will lead to excessive overhead, substantially affecting communication efficiency.

To overcome the limitations of existing scheduling-based random access protocols, the grant-free non-orthogonal multiple access scheme has been proposed [1]. By eliminating the need for complex handshaking between users and the base station (BS), terminals can directly transmit uplink signals composed of non-orthogonal preamble sequences and payload data. This approach effectively reduces communication delays while enabling more devices to access the network simultaneously. However, it also increases the complexity of designing receiver detection algorithms. Previous studies have proposed several advanced receiver schemes, notably sourced random access and unsourced random access [3]. Sourced random access [4] employs pre-assigned preamble sequences to distinguish user identities and facilitates user activity detection and channel estimation using these preambles. In contrast, unsourced random access employs a common codebook, utilizing codewords to convey information and select coding, interleaving, and spreading techniques [5]. Moreover, reference [6] investigates a massive random access scheme utilizing individual user codebooks. This scheme recognizes the uniqueness of user codewords and encodes bits during the codeword selection process, integrating features from both sourced and unsourced random access mechanisms previously described. The reference [6] provides an in-depth performance analysis of massive random access under multiple-input multiple-output (MIMO) Rayleigh channels, taking into account the presence or absence of channel state information (CSI) at the receiver and whether the number of active users is known, thereby

1536-1276 © 2025 IEEE. All rights reserved, including rights for text and data mining, and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: UNIVERSITY OF SOUTHAMPTON. Downloaded on January 09,2026 at 17:30:07 UTC from IEEE Xplore. Restrictions apply.

offering theoretical insights into performance limits. This paper mainly focuses on sourced random access schemes as in [4], tackling algorithm design challenges related to AUD, CE, and DD. For simplicity, we continue to refer to the discussed approach as massive random access.

A. Related Work

For massive random access, identifying the set of active users is a prerequisite for executing subsequent transmission tasks. Studies in [7], [8], and [9] have developed a covariance-based AUD method. This approach initially derives the likelihood function of user activity from received signals using a transmission model, framing AUD as a maximum likelihood estimation problem. The coordinate descent method is then employed to iteratively estimate each user's activity. However, the algorithm's computational complexity is high due to the need for computing the inverse of the covariance matrix in each iteration and performing serial iterative estimation for multiple users. To address this challenge, the authors of [10] and [11] proposed advanced neural network-based AUD methods, which adopt binary cross-entropy as the loss function and utilize a heterogeneous Transformer to directly map received signals to the active user support set, achieving superior active user detection performance and low inference latency.

The aforementioned research assumes that the channel between users and the BS is characterized by a multi-antenna Rayleigh channel, and it does not tackle the scenarios of non-Rayleigh channels. In non-Rayleigh environments, a common strategy involves the joint execution of AUD and CE tasks. This is often framed as a compressed sensing (CS) problem, which is solved using traditional greedy algorithms, Bayesian approaches, or deep learning methods. For example, the existing algorithms, such as orthogonal matching pursuit (OMP) [12], approximate message passing (AMP) [13], [14], and orthogonal AMP (OAMP) [15], exploit the inherent sparsity of user activity to recover the user channel matrix while simultaneously identifying active users. To reduce computational complexity and enhance overall performance, model-driven deep learning approaches [16], [17] have been developed based on traditional CS algorithm frameworks. By integrating learnable parameters with the conventional backbones, such networks demonstrate superior performance under low pilot overhead conditions while significantly accelerating convergence. However, since the primary aim of CS is CE, with activity detection as a secondary outcome, joint AUD and CE methods typically necessitate higher pilot overhead than covariance-based techniques, which directly target the active user set detection [7].

Once the AUD and CE processes are completed, the massive MIMO DD problem at BS often becomes overdetermined. Various advanced detection algorithms have been developed to address this challenge. In addition to traditional methods like least squares (LS) and linear minimum mean square error (LMMSE), algorithms such as OAMP [18], expectation propagation (EP) [19], and deep learning-based techniques [20], [21], [22], [23] have demonstrated excellent performance. However, these DD methods rely on the CSI estimated in the earlier stage, and errors in CSI estimation will degrade the detection performance.

Performing joint CE and DD (JCEDD) can enhance overall achievable performance. The work [21] presents an iterative JCEDD scheme that accounts for channel estimation errors during data detection and employs a model-driven OAMP unfolding algorithm to improve detection. In the CE phase, demodulated data are used as pilots to augment LMMSE CE, resulting in superior performance. Additionally, several studies have addressed the joint AUD, CE, and DD problems, where message-passing algorithms or optimization-based methods have been developed to tackle these tasks [24], [25], [26]. Despite their strong performance, these algorithms require elaborate system configurations and dedicated algorithm design, which may not generalize effectively to practical systems or complex channel environments that are challenging to model, thus limiting their applicability. Consequently, they are not included in the comparisons of this paper.

In recent years, generative model-based methods have been extensively applied to address inverse problems in communications. Unlike Bayesian algorithms, which require manual assumptions of prior signal distributions, generative models employ neural networks to model the signal prior, offering a more effective representation of complex signal distributions. As detailed in [27], the score matching with Langevin dynamics (SMLD) framework is used to tackle the CE problem. This approach involves estimating the score function, defined as the gradient of a log-probability density function [28]. By combining the channel prior's score function, learned from a neural network, with the likelihood score derived from observations, the posterior estimate of high-dimensional channels can be incrementally sampled from Gaussian noise using annealed Langevin dynamics (ALD) [28], [29]. The SMLD framework can also address the MIMO detection problem [30], [31], [32]. To handle the non-differentiable symbol constellation prior, a noise-perturbed version of the symbol's distribution serves as a proxy function to derive the score function [31]. Furthermore, the SMLD framework has been effectively applied to the JCEDD problem [33] as well as the quantized CE problem [34], demonstrating improved performance over existing deep learning and traditional methods.

Another category of generative model frameworks is the denoising diffusion probabilistic model (DDPM) [35]. DDPM methods have been widely applied to denoising tasks in communication systems. For example, the study [36] proposed a lightweight diffusion model that first obtains an initial CE using LS and then progressively denoises it to enhance CE performance. Additionally, the work [37] applied diffusion models to denoise semantic features at the receiver, thereby improving semantic transmission performance.

It has been indicated in [38] that the DDPM process implicitly calculates score function for each noise level, and both DDPM and SMLD can be unified under the framework of score-based generative models. These models can be described using stochastic differential equations (SDEs), which model both the forward noise-injection and reverse generation processes. This SDE approach offers greater flexibility in sampling algorithms, resulting

Authorized licensed use limited to: UNIVERSITY OF SOUTHAMPTON. Downloaded on January 09,2026 at 17:30:07 UTC from IEEE Xplore. Restrictions apply.

in improved performance compared to existing generative models.

B. Our Contributions

This paper proposes a new two-stage algorithm to solve the AUD, CE, and DD tasks in massive random access systems. The first stage identifies the active users set (AUS) through a Transformer-based detector, while the second stage utilizes generative diffusion models (GDMs) to perform JCEDD for active user equipment (UEs). The existing Transformer-based deep learning AUD scheme [10] fails to effectively address the challenges posed by variable pilot lengths in massive random access problems. Furthermore, the performance of current JCEDD schemes relying on ALD-based generative methods [33] has not been comprehensively evaluated and is susceptible to converging to local optima. To tackle these issues, our research makes the following key contributions:

- We design a variable pilot length AUD network (VPL-AUDNet) for AUD task. Specifically: 1) We propose the variable pilot length transmission framework and corresponding adaption mechanism that allows a single network to be applicable to different pilot lengths without retraining. 2) We introduce a spatial correlation mechanism to extract spatial correlation features from the received signals, thereby improving network-based AUD performance in complex correlated channels.
- We model the CE and DD tasks as reverse diffusion processes that sample from a joint distribution. These processes gradually transform initial noise into the original channel and data symbols. By employing advanced predictor-corrector samplers, our scheme outperforms existing JCEDD schemes.
- To accommodate the distinct characteristics of unknown continuous channel distributions and known discrete data constellation distributions, we propose an asynchronous iterative CE and DD framework. This framework integrates an LMMSE-based CE initialization method, which reduces the number of sampling steps and enhances convergence performance.

The remainder of this paper is organized as follows. Section II briefly introduces the massive access system model, and Section III formulates the problems of AUD and JCEDD. Section IV details the proposed Transformer-based AUD scheme, while Section V elaborates on our generative models-based JCEDD scheme. Section VI presents the simulation results, and Section VII concludes the paper.

*Notations:*Matrices and column vectors are denoted by uppercase and lowercase boldface letters, respectively. $\boldsymbol{I}_N$ is the $N \times N$ identity matrix, $\boldsymbol{0}_{N\times M}$ is the $N \times M$ all-zero matrix, and $\boldsymbol{1}_N$ is the $N \times 1$ all-one vector. $(\cdot)^{\mathrm{T}}$, $(\cdot)^*$, $(\cdot)^{\mathrm{H}}$, $(\cdot)^{-1}$, and $\mathbb{E}\{\cdot\}$ denote the transpose, conjugate, Hermitian transpose, inversion, and expectation operations, respectively. $|\mathcal{A}|_c$ indicates the cardinality of the set $\mathcal{A}$. $\mathrm{diag}(\boldsymbol{a})$ transforms vector $\boldsymbol{a}$ into the corresponding diagonal matrix. $\mathcal{N}(\boldsymbol{x};\boldsymbol{a},\boldsymbol{A})$ denotes the probability density function (PDF) of a Gaussian random vector $\boldsymbol{x}$ with mean $\boldsymbol{a}$ and covariance matrix $\boldsymbol{A}$, and the PDF of a complex-valued Gaussian random vector is denoted by $\mathcal{N}_c(\cdot;\cdot,\cdot)$. $\boldsymbol{X}_{[:,\mathcal{A}]}$ ($\boldsymbol{X}_{[\mathcal{A},:]}$) refers to the sub-matrix comprising the columns (rows) of $\boldsymbol{X}$ identified by the index set $\mathcal{A}$, while $\otimes$ and $\odot$ denotes Kronecker and Hadamard product operations, respectively. The operator $\mathrm{vec}(\cdot)$ stacks a matrix column by column into a vector, and $[\boldsymbol{X}]_{i,j}$ is the (i,j)-th element of $\boldsymbol{X}$. $\lfloor\cdot\rfloor$ denotes the integer floor operator.

II. System Model

In this study, we examine a typical narrowband uplink random access scenario where K potential UEs are served by a single BS. The BS is equipped with a uniform planar array (UPA) of $M = M_x \times M_y$ antennas, with M_x and M_y antennas positioned along the x-axis and y-axis, respectively. For simplicity, each UE is assumed to use a single antenna. In a specific transmission frame, only $K_a \ll K$ UEs are active due to the sporadic transmission characteristic of massive random access. The activity of a UE is represented by the indicator α_k, where $\alpha_k = 1$ indicates that the k-th UE is active, and $\alpha_k = 0$ otherwise. By defining the AUS as $\mathcal{A} = \{k \mid \alpha_k = 1, 1 \le k \le K\}$, we have $K_a = |\mathcal{A}|_c$.

For the k-th active UE, the uplink frame consists of L_p pilot symbols $\boldsymbol{p}_k \in \mathbb{C}^{L_p}$ followed by L_d data symbols $\boldsymbol{x}_k \in \mathbb{C}^{L_d}$. These pilot and data symbols form a frame $\boldsymbol{s}_k = \left[\boldsymbol{p}_k^{\mathrm{T}}, \boldsymbol{x}_k^{\mathrm{T}}\right]^{\mathrm{T}} \in \mathbb{C}^L$ of length $L = L_p + L_d$. The channel vector between the k-th UE and the BS is denoted by $\boldsymbol{h}_k \in \mathbb{C}^M$, which is assumed to remain constant throughout the entire transmission frame. The received signal matrix $\boldsymbol{Y} \in \mathbb{C}^{L\times M}$ over a frame from all the active UEs is expressed as:

$$\boldsymbol{Y} = \sum_{k=1}^{K} \alpha_k \boldsymbol{s}_k \boldsymbol{h}_k^{\mathrm{T}} + \boldsymbol{W}, \tag{1}$$

where $\boldsymbol{W} \in \mathbb{C}^{L\times M}$ is the noise matrix, whose elements are symmetric complex additive Gaussian white noise (AWGN) with zero mean and variance σ_n^2.

Define $\boldsymbol{S} \triangleq [\boldsymbol{s}_1, \boldsymbol{s}_2, \ldots, \boldsymbol{s}_K] \in \mathbb{C}^{L\times K}$, $\boldsymbol{A} \triangleq \mathrm{diag}(\boldsymbol{\alpha}) = \mathrm{diag}\left([\alpha_1, \alpha_2, \ldots, \alpha_K]^{\mathrm{T}}\right) \in \mathbb{C}^{K\times K}$, $\bar{\boldsymbol{H}} \triangleq [\boldsymbol{h}_1, \boldsymbol{h}_2, \ldots, \boldsymbol{h}_K]^{\mathrm{T}} \in \mathbb{C}^{K\times M}$, and $\boldsymbol{H} \triangleq \boldsymbol{A}\bar{\boldsymbol{H}}$. Then, the received signal matrix can be expressed as:

$$\boldsymbol{Y} = \boldsymbol{S}\boldsymbol{A}\bar{\boldsymbol{H}} + \boldsymbol{W} = \boldsymbol{S}\boldsymbol{H} + \boldsymbol{W}. \tag{2}$$

By splitting the pilot and data components in $\boldsymbol{S}$ into two parts, i.e., $\boldsymbol{P} \triangleq [\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_K] \in \mathbb{C}^{L_p\times K}$ and $\boldsymbol{X} \triangleq [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_K] \in \mathbb{C}^{L_d\times K}$, we have $\boldsymbol{S} = \left[\boldsymbol{P}^{\mathrm{T}}, \boldsymbol{X}^{\mathrm{T}}\right]^{\mathrm{T}} \in \mathbb{C}^{L\times K}$ and $\boldsymbol{Y} = \left[\boldsymbol{Y}_p^{\mathrm{T}}, \boldsymbol{Y}_d^{\mathrm{T}}\right]^{\mathrm{T}} \in \mathbb{C}^{L\times M}$, where $\boldsymbol{Y}_p \in \mathbb{C}^{L_p\times M}$ and $\boldsymbol{Y}_d \in \mathbb{C}^{L_d\times M}$ correspond to the received pilot and data components, respectively. We also denote $\boldsymbol{W} = \left[\boldsymbol{W}_p^{\mathrm{T}}, \boldsymbol{W}_d^{\mathrm{T}}\right] \in \mathbb{C}^{L\times M}$ with $\boldsymbol{W}_p \in \mathbb{C}^{L_p\times M}$ and $\boldsymbol{W}_d \in \mathbb{C}^{L_d\times M}$.

III. Problem Formulation

The proposed massive random access system is depicted in Fig. 1, where the BS needs to solve the problems of AUD and JCEDD. Specifically, we divide the AUD, CE, and DD tasks into the two stages. In the first stage, we use a single model to derive the AUS information from the received pilots of variable lengths. In the second stage, we perform the JCEDD using the powerful generation capability of diffusion models.

Authorized licensed use limited to: UNIVERSITY OF SOUTHAMPTON. Downloaded on January 09,2026 at 17:30:07 UTC from IEEE Xplore. Restrictions apply.

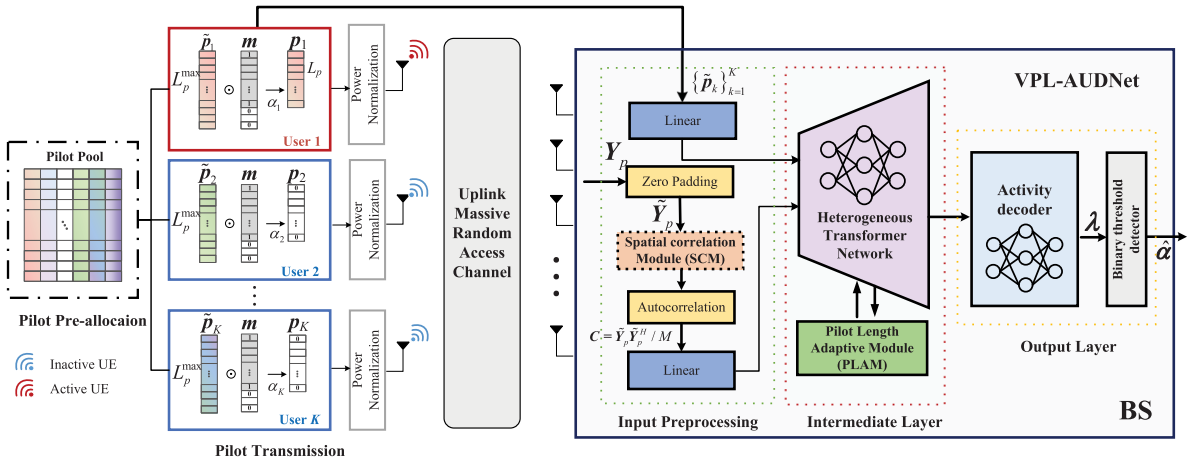


Fig. 1. System diagram of the massive random access with variable-length pilot transmission and corresponding VPL-AUDNet at the receiver.

A. Stage 1-AUD

Due to activity sparsity, existing CS methods [12], [13], [15], [16], [17] recover the entire sparse channel matrix, which requires a large pilot overhead to accurately estimate the CSI. As discussed in Section I, covariance-based methods [7], [8], [9] have not been successfully generalized to non-Rayleigh channels. Additionally, current deep learning-based methods [10], [11] fail to generalize to variable pilot lengths once the network is trained with a fixed pilot length. To address these issues, we propose a neural network-based method that can generalize to variable pilot lengths and perform well with low pilot overhead in complex correlated channel environments.

We assume that the original pilot sequence for each UE, denoted as $\tilde{\boldsymbol{p}}_k$, is selected from a large pilot pool $\tilde{\boldsymbol{P}} = [\tilde{\boldsymbol{p}}_1, \tilde{\boldsymbol{p}}_2, \ldots, \tilde{\boldsymbol{p}}_K] \in \mathbb{C}^{L_p^{\max} \times K}$ and is pre-allocated by the BS when the UE initially registers with the BS. The UE identity is then bound to this pilot sequence until it un-registers from the network. During the transmission procedure as shown in the left part of Fig. 1, only the first L_p symbols of each pilot sequence are transmitted. That is, each pilot sequence is masked by a binary vector $\boldsymbol{m} = [1, \ldots, 1, 0, \ldots, 0]^{\mathrm{T}} \in \{0,1\}^{L_p^{\max}}$, whose first L_p elements are ones and the remaining $\left(L_p^{\max} - L_p\right)$ elements are zeros. Here, L_p can take a value in the range $[L_p^{\min}, L_p^{\max}]$. Let $\boldsymbol{M} = \mathbf{1}_K^{\mathrm{T}} \otimes \boldsymbol{m} \in \mathbb{B}^{L_p^{\max} \times K}$. Then, at the AUD stage, the BS identifies the active UEs from the received pilot signals, expressed as:

$$\tilde{\boldsymbol{Y}}_p = \left(\tilde{\boldsymbol{P}} \odot \boldsymbol{M}\right) \boldsymbol{H} + \tilde{\boldsymbol{W}}_p. \tag{3}$$

Here, $\tilde{\boldsymbol{Y}}_p = \left[\boldsymbol{Y}_p^{\mathrm{T}}, \boldsymbol{0}_{(L_p^{\max} - L_p) \times M}\right] \in \mathbb{C}^{L_p^{\max} \times M}$ is the equivalent received pilot after zero-padding it to the maximum pilot sequence length. Similarly, $\tilde{\boldsymbol{W}}_p = \left[\boldsymbol{W}_p^{\mathrm{T}}, \boldsymbol{0}_{(L_p^{\max} - L_p) \times M}\right] \in \mathbb{C}^{L_p^{\max} \times M}$ is the equivalent noise at the receiver. Since the equivalent channel matrix can be written as $\boldsymbol{H} = [\alpha_1 \boldsymbol{h}_1, \alpha_2 \boldsymbol{h}_2, \ldots, \alpha_K \boldsymbol{h}_K]^{\mathrm{T}} \in \mathbb{C}^{K \times M}$, estimating the activity $\hat{\boldsymbol{\alpha}} = [\hat{\alpha}_1, \hat{\alpha}_2, \ldots, \hat{\alpha}_K]^{\mathrm{T}} \in \mathbb{B}^{K \times 1}$ is equivalent to recognizing the support set of $\boldsymbol{H}$.

The AUD problem can then be formulated as follows:

$$\begin{aligned} \min_{\boldsymbol{\Phi}} \quad & \mathbb{E}\left\{\mathcal{L}\left(\boldsymbol{\alpha}, \hat{\boldsymbol{\alpha}}\right)\right\}, \\ \text{s.t.} \quad & \hat{\boldsymbol{\alpha}} = f_{\boldsymbol{\Phi}}\left(\tilde{\boldsymbol{Y}}_p, \tilde{\boldsymbol{P}}, L_p\right), \ \forall L_p \in \left[L_p^{\min}, L_p^{\max}\right], \end{aligned} \tag{4}$$

where $f_{\boldsymbol{\Phi}}\left(\tilde{\boldsymbol{Y}}_p, \tilde{\boldsymbol{P}}, L_p\right)$ is a neural network which maps the received signal and pilot sequences to $\hat{\boldsymbol{\alpha}}$, $\boldsymbol{\Phi}$ represents the parameters of the neural network, and $\mathcal{L}\left(\boldsymbol{\alpha}, \hat{\boldsymbol{\alpha}}\right)$ is the loss function that quantifies the disparity between $\boldsymbol{\alpha}$ and $\hat{\boldsymbol{\alpha}}$.

The advantage of variable pilot-length transmission is not only to avoid training and storing multiple neural networks for different pilot lengths but also to allow the UE to transmit different pilot lengths according to varying traffic densities at different times of the day. This reduces transmission delay while maintaining flexibility. The proposed AUD scheme is detailed Section IV.

B. Stage 2-JCEDD

Given the estimated AUS $\hat{\mathcal{A}} = \{k \,|\, \hat{\alpha}_k = 1, 1 \le k \le K\}$, the transmission model can be equivalently formulated as:

$$\boldsymbol{Y} = \boldsymbol{S}_{[:,\hat{\mathcal{A}}]} \boldsymbol{H}_{[\hat{\mathcal{A}},:]} + \tilde{\boldsymbol{W}}, \tag{5}$$

where $\boldsymbol{S}_{[:,\hat{\mathcal{A}}]} = \left[\boldsymbol{P}_{[:,\hat{\mathcal{A}}]}^{\mathrm{T}}, \boldsymbol{X}_{[:,\hat{\mathcal{A}}]}^{\mathrm{T}}\right]^{\mathrm{T}} \in \mathcal{X}^{L \times \hat{K}_a}$, with $\mathcal{X}$ denoting the quadrature amplitude modulation (QAM) constellation set, and $\boldsymbol{H}_{[\hat{\mathcal{A}},:]} \in \mathbb{C}^{\hat{K}_a \times M}$. For convenience, define $\boldsymbol{S}_a \triangleq \boldsymbol{S}_{[:,\hat{\mathcal{A}}]}$, $\boldsymbol{P}_a \triangleq \boldsymbol{P}_{[:,\hat{\mathcal{A}}]}$, $\boldsymbol{X}_a \triangleq \boldsymbol{X}_{[:,\hat{\mathcal{A}}]}$, and $\boldsymbol{H}_a \triangleq \boldsymbol{H}_{[\hat{\mathcal{A}},:]}$. Given the known observations $\boldsymbol{Y}$ and pilots $\boldsymbol{P}_a$, we find the maximum a posteriori estimation of both the data symbols $\boldsymbol{X}_a$ and channels $\boldsymbol{H}_a$, which can be formulated as:

$$\left\{\hat{\boldsymbol{X}}_a, \hat{\boldsymbol{H}}_a\right\} = \arg \max_{\boldsymbol{X}_a \in \mathcal{X}^{L_d \times \hat{K}_a}, \boldsymbol{H}_a} p\left(\boldsymbol{X}_a, \boldsymbol{H}_a | \boldsymbol{Y}, \boldsymbol{P}_a\right). \tag{6}$$

The above problem can be divided into the two sub-problems when one of the two variables is assumed to be known:

$$\hat{\boldsymbol{X}}_a = \arg \max_{\boldsymbol{X}_a \in \mathcal{X}^{L_d \times \hat{K}_a}} p\left(\boldsymbol{X}_a | \boldsymbol{Y}, \hat{\boldsymbol{H}}_a, \boldsymbol{P}_a\right), \tag{7a}$$

Authorized licensed use limited to: UNIVERSITY OF SOUTHAMPTON. Downloaded on January 09,2026 at 17:30:07 UTC from IEEE Xplore. Restrictions apply.

$$\hat{\boldsymbol{H}}_a = \arg \max_{\boldsymbol{H}_a} p\left(\boldsymbol{H}_a | \boldsymbol{Y}, \hat{\boldsymbol{X}}_a, \boldsymbol{P}_a\right). \tag{7b}$$

Different from typical passing algorithms that factorize the joint distribution (6) into a factor graph and then inferring the posterior distribution of each element through message passing, our proposed scheme directly samples the signals from the posterior distributions (7a) and (7b). This approach was first implemented in [27], [31], [32], and [33], where the ALD sampler is used to sample from the posterior distribution in the CE or DD problem.

The ALD sampler provides a viable method for sampling from complex distributions. Recent advances highlight the significant role of the score function in the ALD sampler, revealing a strong connection to the denoiser in the widely recognized DDPM model. Both concepts can be clarified through the perspective of SDE [38]. Specifically, sampling from a distribution can be interpreted as solving the reverse SDE of a corresponding forward SDE in the diffusion process. Based on this insight, we address the shortcomings of the existing ALD sampler [33] to provide more reliable estimation results for the JCEDD problem. The proposed JCEDD scheme is detailed in Section V.

IV. Transformer-Based AUD for Variable Pilot Lengths

As illustrated in the right part of Fig. 1, the receiver's primary function is to extract correlations between the received signal and the user pilot sequences, facilitating the identification of active users whose pilots contribute to the received signal. We propose a VPL-AUDNet consisting of three main blocks. First, in the preprocessing block, we design a unified method for preprocessing received pilot signals with varying lengths. In the second block, the feature sequences from the previous block are concurrently fed into a neural network for correlation extraction, utilizing a heterogeneous Transformer network as the backbone [10] to handle the different physical meanings of the received signal and pilot sequences. To accommodate variable pilot lengths, we incorporate an additional pilot length adaptive module (PLAM) to make the Transformer network cognizant of key information from the input sequence. In the output layer, the activity decoder calculates the correlation between the feature sequences of $\tilde{\boldsymbol{Y}}_p$ and the multi-user pilot sequences $\{\tilde{\boldsymbol{p}}_k\}_{k=1}^K$ to estimate the user activity vector $\boldsymbol{\lambda}$. The detailed AUD signal processing flow at the receiver is presented below.

A. Preprocessing

Before inputting the user pilots and received signals into the Transformer-network block, preprocessing is required to ensure dimensional consistency. The user pilots need to be converted into real-valued representations, resulting in K sequences, denoted as $\tilde{\boldsymbol{z}}_k \triangleq \left[\Re\{\tilde{\boldsymbol{p}}_k\}^{\mathrm{T}}, \Im\{\tilde{\boldsymbol{p}}_k\}^{\mathrm{T}}\right]^{\mathrm{T}} \in \mathbb{R}^{2L_p^{\max}}$, $1 \le k \le K$, each of length $2L_p^{\max}$. These sequences are then processed through a linear layer to produce the pilot feature sequences $\{\boldsymbol{z}_k^0\}_{k=1}^K$, each of length d. Concurrently, to handle varying pilot transmission lengths L_p at the receiver, zero-padding is applied to the received signal $\boldsymbol{Y}_p \in \mathbb{C}^{L_p \times M}$ to convert it into $\tilde{\boldsymbol{Y}}_p \in \mathbb{C}^{L_p^{\max} \times M}$ (see (3)). Subsequently, to leverage the multi-antenna observations at the BS, the covariance matrix [8], [10] of the received signals is calculated as $\boldsymbol{C} \triangleq \tilde{\boldsymbol{Y}}_p \tilde{\boldsymbol{Y}}_p^{\mathrm{H}} / M \in \mathbb{C}^{L_p^{\max} \times L_p^{\max}}$. This covariance matrix is transformed into a real-valued vectorized form, denoted as $\tilde{\boldsymbol{z}}_{K+1} \triangleq \left[\Re\{\mathrm{vec}(\boldsymbol{C})\}^{\mathrm{T}}, \Im\{\mathrm{vec}(\boldsymbol{C})\}^{\mathrm{T}}\right]^{\mathrm{T}} \in \mathbb{R}^{2\left(L_p^{\max}\right)^2}$. The resulting vector undergoes a linear layer transformation to achieve a dimensionality of d, denoted as $\boldsymbol{z}_{K+1}^0$. Finally, these K pilot sequences and the received signal sequence, $\{\boldsymbol{z}_k^0\}_{k=1}^{K+1}$, are collectively fed into the subsequent heterogeneous Transformer for further feature extraction.

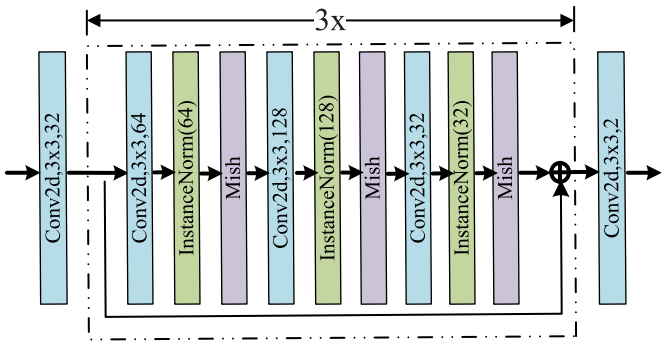


Fig. 2. Block diagram of the proposed spatial correlation module.

In scenarios involving spatially correlated channels, directly mapping the covariance matrix onto a sequence may fail to effectively capture the antenna correlations. To address this limitation, we introduce a spatial correlation module (SCM) between the zero-padding and autocorrelation processes. Specifically, the received signal $\tilde{Y}_p \in \mathbb{C}^{L_{\max} \times M}$ is first transformed into a real tensor with dimensions $L_{\max} \times 2 \times M_x \times M_y$ and then inputted into the SCM. To prevent the network parameters from being dependent on the number of antennas, the SCM uses a fully convolutional residual network to extract features along the vertical and horizontal antenna dimensions, as depicted in Fig. 2. The output of the SCM is reshaped to dimensions $L_{\max} \times 2 \times M$ for subsequent autocorrelation calculation. For Rayleigh channels with independent identical distribution (i.i.d.), this module can be omitted. Simulation results in Section VI demonstrate that incorporating the SCM significantly enhances the performance of AUD in correlated channels such as 3GPP channels [39].

B. Heterogeneous Transformer and Activity Decoder

Unlike the traditional Transformer, the heterogeneous Transformer introduced in [10] employs distinct sets of weight matrices for input embedding, multi-head attention (MHA), and the feed-forward network (FFN) to account for the different physical meanings of pilot sequences and received signals. To facilitate understanding, we illustrate the heterogeneous Transformer structure and MHA in Appendix A. Since the relative positional order of different sequences is not meaningful in our AUD task, we do not apply positional encoding as in conventional Transformers.

Assuming that the heterogeneous Transformer network consists of N encoder layers, we conduct activity estimation for

Authorized licensed use limited to: UNIVERSITY OF SOUTHAMPTON. Downloaded on January 09,2026 at 17:30:07 UTC from IEEE Xplore. Restrictions apply.

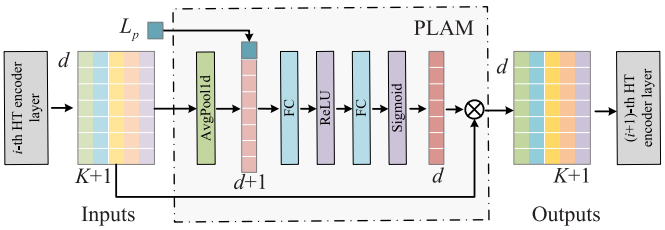


Fig. 3. Block diagram of the proposed pilot length adaptive module.

K users based on the $K+1$ sequences output by the N-th encoder layer. This task is performed by the activity decoder, which computes the heterogeneous MHA between the final output feature sequence of the received signal $\boldsymbol{z}_{K+1}^{N+1}$ and pilot sequences $\{\boldsymbol{z}_k^{N+1}\}_{k=1}^K$. The resulting MHA output, $\boldsymbol{z}_c$, represents the synthesized contributions of all pilot sequences to the received signal. Next, the user activity is estimated as the weighted correlation between $\boldsymbol{z}_c$ and each pilot feature sequence. It is given by:

$$\lambda_k' = C\tanh\left(\frac{\boldsymbol{z}_c^{\mathrm{T}}\boldsymbol{W}^o\boldsymbol{z}_k^{N+1}}{\sqrt{d}}\right), \lambda_k = \frac{1}{1+e^{-\lambda_k'}},\ \forall k. \quad (8)$$

Here, $\{\lambda_k\}_{k=1}^K$ is the output of the activity decoder, $\boldsymbol{W}^o \in \mathbb{C}^{d\times d}$ is a learnable parameter matrix, and C is a hyperparamter empirically set to $C=10$. User activity is then determined by a binary threshold detector: $\hat{\alpha}_k = \mathbb{I}(\lambda_k > 0.5),\ \forall k$, where the indicator $\mathbb{I}(x)=1$ if x is true, and 0 otherwise.

C. Pilot Length Adaptive Module

In variable pilot-length transmission, the input sequence at the receiver undergoes zero-padding, causing some elements to carry no useful information. As a result, the significance of different elements in the input sequence varies. Allowing the neural network to recognize this variability enables adaptive optimization of network parameters based on the corresponding pilot lengths.

Inspired by the attention feature module in [40], we propose a PLAM. By inserting a PLAM module between each pair of adjacent Transformer encoder layers, we provide additional pilot length information to the receiver. The structure of the proposed PLAM module is illustrated in Fig. 3. Its input consists of $(K+1)$ sequences from the output of the previous Transformer encoder layer, and its output comprises $(K+1)$ sequences of the same dimension, which serve as the input to the next layer. First, an average pooling operation is applied to the $(K+1)$ sequences along the sequence length dimension, producing a sequence of length d. This d-dimensional sequence is then concatenated with the pilot length information L_p and processed through a two-layer fully connected network, generating a d-dimensional weight sequence. Finally, this weight sequence is used for position-wise dot multiplication with all input sequences, redistributing the weights of the input feature sequence and allowing the receiver to recognize the varying importance of different elements in the inputs.

D. Loss Function

We use binary cross-entropy as the loss function. The final loss is computed as the average of the loss values across different pilot lengths, given by:

$$\mathcal{L} = \frac{1}{|\mathcal{D}|K}\sum_{i=1}^{|\mathcal{D}|}\sum_{k=1}^{K}\Big(\alpha_k^{(i)}\log\lambda_k^{(i)} + \left(1-\alpha_k^{(i)}\right)\left(1-\log\lambda_k^{(i)}\right)\Big). \quad (9)$$

Here, $\mathcal{D}$ denotes the dataset, which consists of training samples $\left\{\boldsymbol{Y}_p^{(i)}, \tilde{\boldsymbol{P}}^{(i)}, \boldsymbol{\alpha}^{(i)}\right\}$ with different pilot lengths L_p, where $L_p \in \left[L_p^{\min}, L_p^{\max}\right]$.

V. Generative Diffusion Models Driven JCEDD Using Predictor-Corrector Sampler

Given the active UE set, we need to recover the channel and corresponding data symbols of the active UEs. Before introducing the proposed JCEDD schemes, we briefly review the relationship between existing generative models and their SDE descriptions. We then address the CE problem, detailing the calculation of posterior score functions within the predictor-corrector (PC) sampler framework. Subsequently, we extend the application of the PC sampler to the DD problem. Finally, we combine the CE and DD methods to propose an iterative algorithm for JCEDD and analyze its complexity.

A. Preliminary: Diffusion Process and SDE

Existing diffusion-based generative models involve corrupting data with multiple noise scales to form a known prior distribution. These models then learn to transform this prior distribution back into the data distribution by gradually removing noise. SDE considers a continuum of distributions that evolve over time according to a diffusion process. The forward diffusion process $\boldsymbol{x}(t)$, indexed by the continuous time variable $t\in(0,T]$, progressively diffuses a data point $\boldsymbol{x}_0\in\mathbb{R}^d$ sampled from a dataset with an unknown distribution $p_0(\boldsymbol{x}) = p_{\text{data}}(\boldsymbol{x})$ into random noise $\boldsymbol{x}(T)$ with distribution $p_T(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x};\boldsymbol{0},\sigma_T^2\boldsymbol{I})$. For simplicity in notation, we define $p_t(\boldsymbol{x}) \triangleq p(\boldsymbol{x}(t))$. This process can be described by an SDE that has no trainable parameters [38], [41], as follows:

$$\mathrm{d}\boldsymbol{x} = \boldsymbol{f}(\boldsymbol{x},t)\mathrm{d}t + g(t)\mathrm{d}\boldsymbol{w}, \quad (10)$$

where $\boldsymbol{f}(\boldsymbol{x},t)\colon \mathbb{R}^d\times\mathbb{R}\to\mathbb{R}^d$ is a vector function called the drift coefficient, and the scalar function $g(t)\colon \mathbb{R}\to\mathbb{R}$ is the diffusion coefficient. Additionally, $\boldsymbol{w}$ is a standard Wiener process, defined as $\mathrm{d}\boldsymbol{w} = \boldsymbol{z}(t)\mathrm{d}t$, where $\boldsymbol{z}(t)\sim\mathcal{N}(\boldsymbol{z}(t);\boldsymbol{0},\boldsymbol{I})$.

In the reverse process, starting from a sample $\boldsymbol{x}_T\sim p_T(\boldsymbol{x})$, random noise is smoothly transformed into a data sample $\boldsymbol{x}_0\sim p_0(\boldsymbol{x})$. This reverse process satisfies a time-reversed SDE, which can be derived from the forward SDE given the score function $\nabla_{\boldsymbol{x}}\log p_t(\boldsymbol{x})$ for the marginal probability of $\boldsymbol{x}$ as a function of time, i.e.,

$$\mathrm{d}\boldsymbol{x} = \left(\boldsymbol{f}(\boldsymbol{x},t) - g(t)^2\nabla_{\boldsymbol{x}}\log p_t(\boldsymbol{x})\right)\mathrm{d}t + g(t)\mathrm{d}\bar{\boldsymbol{w}}. \quad (11)$$

Here, $p_t(\boldsymbol{x})$ denotes the probability of $\boldsymbol{x}$ at time t, and $\bar{\boldsymbol{w}}$ denotes the Wiener process when time flows backward from

Authorized licensed use limited to: UNIVERSITY OF SOUTHAMPTON. Downloaded on January 09,2026 at 17:30:07 UTC from IEEE Xplore. Restrictions apply.

T to 0. Existing generative models like SMLD [28] and DDPM [35] can be considered specific numerical discretization solvers of the SDE. For instance, in SMLD, the coefficients are $\boldsymbol{f}(\boldsymbol{x},t)=\mathbf{0}$ and $g(t)=\sqrt{\frac{\mathrm{d}(\sigma^2(t))}{\mathrm{d}t}}$, with $\sigma(t)$ serving as a predefined noise scheduling function. For DDPM, the drift and diffusion coefficients are given by $\boldsymbol{f}(\boldsymbol{x},t)=-\frac{\sigma(t)\boldsymbol{x}}{2}$ and $g(t)=\sqrt{\sigma(t)}$, respectively. By applying the discrete form of both the forward and reverse SDEs, we can derive the corresponding iteration equations used in SMLD and DDPM.

B. Channel Estimation

As discussed in [38], the generation process involves solving the reversed SDE given in Eq. (11), where the unconditional generation from this equation yields samples that adhere to the prior probability distribution $p_0(\boldsymbol{x})$. However, considering Eq. (5) in the JCEDD problem, since the BS receives signal $\boldsymbol{Y}$, we need to consider the conditional generation process where channel should be drawn from the posterior distribution $p\left(\boldsymbol{H}_a|\boldsymbol{Y},\hat{\boldsymbol{X}}_a,\boldsymbol{P}_a\right)$ to produce the CE result that align with the constraint $\boldsymbol{Y}=\boldsymbol{S}_a\boldsymbol{H}_a+\boldsymbol{W}$.

Specifically, we assume that the data symbols $\hat{\boldsymbol{X}}_a$ are known and represent the initial posterior distribution as $p_0\left(\boldsymbol{H}_a|\boldsymbol{Y}\right)\triangleq p\left(\boldsymbol{H}_a(0)|\boldsymbol{Y}\right)$, where $\boldsymbol{H}_a(0)=\boldsymbol{H}_a$ represents the ground truth channel to be estimated. The conditional distribution of $\boldsymbol{H}_a(t)$ at time t is expressed as $p_t\left(\boldsymbol{H}_a|\boldsymbol{Y}\right)\triangleq p\left(\boldsymbol{H}_a(t)|\boldsymbol{Y}\right)$. According to [42], the conditional generation process involves sampling from $p_0\left(\boldsymbol{H}_a|\boldsymbol{Y}\right)$, which is equivalent to solving a reverse SDE represented as follows:

$$\begin{aligned}\mathrm{d}\boldsymbol{H}_a=&\left(\boldsymbol{f}(\boldsymbol{H}_a,t)-g(t)^2\nabla_{\boldsymbol{H}_a}\log p_t(\boldsymbol{H}_a|\boldsymbol{Y})\right)\mathrm{d}t\\&+g(t)\mathrm{d}\bar{\boldsymbol{W}}.\end{aligned}\tag{12}$$

Due to the equivalence between DDPM and SMLD during the inference process [43], the remainder of this paper will examine the SDE in the SMLD form. Discussions on DDPM are similar and are therefore omitted.

To solve the reversed SDE (12), it is essential to derive the posterior score function $\nabla_{\boldsymbol{H}_a}\log p_t\left(\boldsymbol{H}_a|\boldsymbol{Y}\right)$. By applying Bayes' theorem, the posterior score function is given by:

$$\begin{aligned}\nabla_{\boldsymbol{H}_a}\log p_t\left(\boldsymbol{H}_a|\boldsymbol{Y}\right)=&\nabla_{\boldsymbol{H}_a}\log p_t\left(\boldsymbol{Y}|\boldsymbol{H}_a\right)\\&+\nabla_{\boldsymbol{H}_a}\log p_t\left(\boldsymbol{H}_a\right),\end{aligned}\tag{13}$$

where the *likelihood score function* $\nabla_{\boldsymbol{H}_a}\log p_t\left(\boldsymbol{Y}|\boldsymbol{H}_a\right)$ is determined by the signal transmission model (5), and the *prior score function* $\nabla_{\boldsymbol{H}_a}\log p_t\left(\boldsymbol{H}_a\right)$ is determined by the prior distribution of the channel dataset.

1) Calculation of Prior Score Function: Denote $\boldsymbol{h}_k\triangleq[\boldsymbol{H}_a]_{[k,:]}^{\mathrm{T}}\in\mathbb{C}^M,\ \forall k\in\mathcal{A}$. Since the prior distributions of each active UE's channel are assumed to be independent of each other, we have $\log p_t\left(\boldsymbol{H}_a\right)=\log\prod_{k=1}^{K_a}p_t(\boldsymbol{h}_k)=\sum_{k=1}^{K_a}\log p_t(\boldsymbol{h}_k)$. Therefore, the prior score function can be calculated for each UE separately, i.e.,

$$\begin{aligned}\nabla_{\boldsymbol{H}_a}\log p_t\left(\boldsymbol{H}_a\right)=&\Big[\nabla_{\boldsymbol{h}_1}\log p_t\left(\boldsymbol{h}_1\right),\nabla_{\boldsymbol{h}_2}\log p_t\left(\boldsymbol{h}_2\right),\\&\cdots,\ \nabla_{\boldsymbol{h}_{K_a}}\log p_t\left(\boldsymbol{h}_{K_a}\right)\Big]^{\mathrm{T}}.\end{aligned}\tag{14}$$

Therefore, when all the UEs' channels are generated from the same dataset distribution $\boldsymbol{p}_{\text{data}}(\boldsymbol{h})$, we only need to focus on the prior score function of a single UE.

For simplicity, omit the UE index k and denote $\boldsymbol{p}_0(\boldsymbol{h})\triangleq\boldsymbol{p}_{\text{data}}(\boldsymbol{h})$. Our objective is to derive the score function $\nabla_{\boldsymbol{h}}\log p_t\left(\boldsymbol{h}\right)$, where $p_t\left(\boldsymbol{h}\right)$ is the data distribution diffused by noise at time t. For complex channel distributions, the prior score function lacks a closed-form expression. Fortunately, the study [44] has shown that using a neural network to learn the score function $\nabla_{\boldsymbol{h}}\log p_t\left(\boldsymbol{h}\right)$ is equivalent to denoising score matching (DSM). This can be achieved by training a time-dependent neural network $\boldsymbol{s}_{\boldsymbol{\theta}}(\boldsymbol{h}(t),t)$ using the following loss function:

$$\mathcal{L}_s=\mathbb{E}_t\left\{\lambda(t)\mathbb{E}\left\{\left\|\boldsymbol{s}_{\boldsymbol{\theta}}(\boldsymbol{h}(t),t)-\nabla_{\boldsymbol{h}(t)}\log p(\boldsymbol{h}(t)|\boldsymbol{h}(0))\right\|_2^2\right\}\right\},\tag{15}$$

where the inner expectation is taken over the joint distribution $p(\boldsymbol{h}(t),\boldsymbol{h}(0))$ with $\boldsymbol{h}(0)$ sampled from the dataset distribution $p_0(\boldsymbol{h})$, and the time t is uniformly sampled from $(0,1]$ with $T=1$, while $\lambda(t)\propto 1/\mathbb{E}[\|\nabla_{\boldsymbol{h}(t)}\log p(\boldsymbol{h}(t)|\boldsymbol{h}(0))\|_2^2]$ is a positive weight function, and $\boldsymbol{\theta}$ collects the neural network parameters.

Specifically, $p(\boldsymbol{h}(t)|\boldsymbol{h}(0))$ denotes the transition kernel from $\boldsymbol{h}(0)$ to $\boldsymbol{h}(t)$, typically defined by a forward noise diffusion process given by Eq. (10). For a typical generation model in the SMLD form [38], with a known drift coefficient $\boldsymbol{f}(\boldsymbol{h},t)=\mathbf{0}$ and a diffusion coefficient $g(t)=\sqrt{\frac{\mathrm{d}(\sigma^2(t))}{\mathrm{d}t}}$, the transition probability function from the original data point $\boldsymbol{h}(0)$ to $\boldsymbol{h}(t)$ can be derived using *Ito's Lemma* [41]. This is given by $p\left(\boldsymbol{h}\left(t\right)|\boldsymbol{h}\left(0\right)\right)=\mathcal{N}\left(\boldsymbol{h}(t);\boldsymbol{h}(0),\left(\sigma^2(t)-\sigma^2(0)\right)\boldsymbol{I}\right)\approx\mathcal{N}\left(\boldsymbol{h}(t);\boldsymbol{h}(0),\sigma^2(t)\boldsymbol{I}\right)$, where $\sigma(t)=\sigma_{\min}\left(\frac{\sigma_{\max}}{\sigma_{\min}}\right)^t$ is a predefined noise scheduling function, and the approximation holds when the initial noise variance $\sigma^2(0)=\sigma_{\min}\approx 0$. Consequently, we have $\nabla_{\boldsymbol{h}(t)}\log p(\boldsymbol{h}(t)|\boldsymbol{h}(0))=-\frac{\boldsymbol{h}(t)-\boldsymbol{h}(0)}{\sigma^2(t)}$. The final loss function can thus be simplified as:

$$\mathcal{L}_s=\mathbb{E}_t\left\{\lambda(t)\mathbb{E}\left(\left\|\boldsymbol{s}_{\boldsymbol{\theta}}(\boldsymbol{h}(t),t)+\frac{\boldsymbol{h}(t)-\boldsymbol{h}(0)}{\sigma^2(t)}\right\|_2^2\right)\right\},\tag{16}$$

which is equivalent to training a denoising neural network that can recover the scaled noise superposed on the channel. With sufficient data and model capacity, DSM ensures that the optimal solution $\boldsymbol{s}_{\boldsymbol{\theta}^*}(\boldsymbol{h}(t),t)$ approximates $\nabla_{\boldsymbol{h}}\log p_t(\boldsymbol{h})$ for almost all $\boldsymbol{h}(t)$ and t.

Given the powerful expressive capability of the noise conditional score network (NCSN++) [38] in image generation tasks, and considering that the channel can be viewed as a special kind of image, we use NCSN++ to construct $\boldsymbol{s}_{\boldsymbol{\theta}^*}(\boldsymbol{h}(t),t)$. To adapt the channel dimension to the input of NCSN++, the original M-dimensional channel vector sample is transformed into a real representation and reshaped into dimensions $2\times M_x\times M_y$. NCSN++ then takes the channel of these dimensions and a time step index t as inputs. The network outputs the learned channel prior score at time t. It should be noted that during the prior score function learning, no information on the pilot symbols, data symbols, received signal, or transmission noise variance is involved in the training process. This means that once the neural network is trained, it can be applied to

Authorized licensed use limited to: UNIVERSITY OF SOUTHAMPTON. Downloaded on January 09,2026 at 17:30:07 UTC from IEEE Xplore. Restrictions apply.

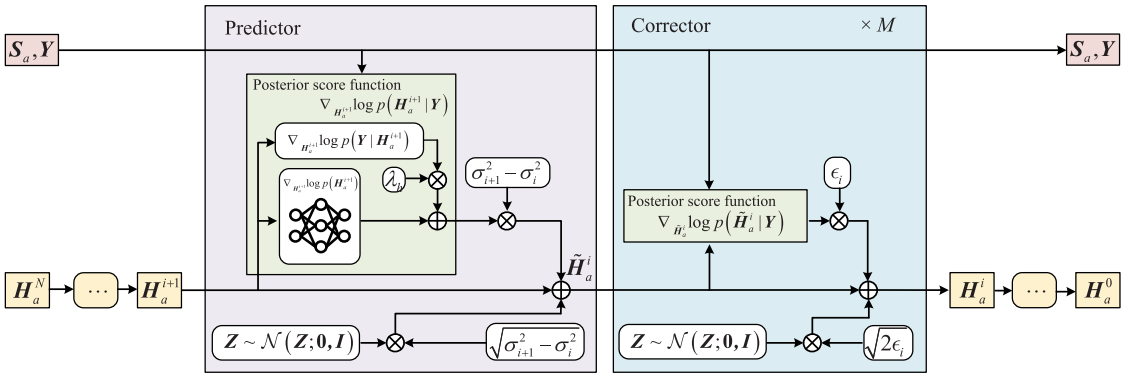


Fig. 4. Block diagram of the proposed PC sampler-based CE algorithm.

any dimension of the JCEDD problem as long as the channel distribution remains unchanged.

One special case where the channel prior score function does not need to be learned is when the channel distribution has a known closed form, for example, $\boldsymbol{h}(0) \sim \mathcal{N}(\boldsymbol{h}(0); \boldsymbol{\mu}_h, \boldsymbol{R}_{hh})$. Then, the prior score function of the channel in this case can be derived by Tweedie's formula [45]

$$\begin{aligned}\nabla_{\boldsymbol{h}(t)} \log p(\boldsymbol{h}(t)) &= \frac{\mathbb{E}\{\boldsymbol{h}(0)|\boldsymbol{h}(t)\} - \boldsymbol{h}(t)}{\sigma^2(t)} \\ &= -\left(\boldsymbol{R}_{hh} + \sigma(t)^2 \boldsymbol{I}\right)^{-1} (\boldsymbol{h}(t) - \boldsymbol{\mu}_h),\end{aligned} \tag{17}$$

where the posterior expectation can be derived according to Bayes linear Gaussian model, which is given by$\mathbb{E}\{\boldsymbol{h}(0)|\boldsymbol{h}(t)\} = \boldsymbol{\mu}_h + \boldsymbol{R}_{hh}\left(\boldsymbol{R}_{hh} + \sigma^2(t)\boldsymbol{I}\right)^{-1}(\boldsymbol{h}(t) - \boldsymbol{\mu}_h)$.

2) Calculation of Likelihood Score Function: We now consider how to derive $\nabla_{\boldsymbol{H}_a} \log p_t(\boldsymbol{Y}|\boldsymbol{H}_a)$. Given that the channels are viewed as three-dimensional real tensors in prior score learning, we can also transform the transmission model (5) into its corresponding real-valued representation. For simplicity, we retain the notation $\boldsymbol{Y} = \boldsymbol{S}_a \boldsymbol{H}_a + \boldsymbol{W}$ throughout this paper. Note that in the SMLD framework, $p(\boldsymbol{H}_a(t)|\boldsymbol{H}_a(0)) \approx \mathcal{N}\left(\boldsymbol{H}_a(t); \boldsymbol{H}_a(0), \sigma^2(t)\boldsymbol{I}\right)$. Consequently, the transmission model can equivalently be expressed as:

$$\boldsymbol{Y} = \boldsymbol{S}_a\left(\boldsymbol{H}_a(t) + \sigma(t)^2 \boldsymbol{Z}\right) + \boldsymbol{W}, \tag{18}$$

where $\boldsymbol{Z} \sim \mathcal{N}(\boldsymbol{Z}; \boldsymbol{0}, \boldsymbol{I})$ is the diffusion noise superimposed on the original channel $\boldsymbol{H}_a(0)$. Therefore, the corresponding transition probability is given by:

$$p_t(\boldsymbol{Y}|\boldsymbol{H}_a) \sim \mathcal{N}\left(\boldsymbol{Y}; \boldsymbol{S}_a \boldsymbol{H}_a(t), \sigma_n^2 \boldsymbol{I} + \sigma^2(t) \boldsymbol{S}_a \boldsymbol{S}_a^{\mathrm{T}}\right). \tag{19}$$

Taking the gradient of the log-likelihood, we obtain:

$$\nabla_{\boldsymbol{H}_a} \log p_t(\boldsymbol{Y}|\boldsymbol{H}_a) = \boldsymbol{S}_a^{\mathrm{T}}\left(\sigma_n^2 \boldsymbol{I} + \sigma^2(t)\boldsymbol{S}_a \boldsymbol{S}_a^{\mathrm{T}}\right)^{-1}(\boldsymbol{Y} - \boldsymbol{S}_a \boldsymbol{H}_a(t)). \tag{20}$$

In the considered JCEDD problem, $\boldsymbol{S}_a = \left[\boldsymbol{P}_a^{\mathrm{T}}, \hat{\boldsymbol{X}}_a^{\mathrm{T}}\right]^{\mathrm{T}}$ contains both the pilot symbol $\boldsymbol{P}_a$ and estimated data symbol $\hat{\boldsymbol{X}}_a$ in the iteration process, and the estimated data symbol can enhance the CE. When no data symbol is used, the problem degrades to the conventional pure pilot-aided CE problem.

3) Channel Sampling Process: Utilizing the derived prior score function and likelihood score function, we numerically solve the reverse SDE (12). Specifically, by dividing the interval $t \in (0, 1]$ into N discrete steps, indexed as $i = 0, 1, \ldots, N-1$, we obtain the following iterative rule for the reverse diffusion sampler (which is also called *predictor*):

$$\begin{aligned}\boldsymbol{H}_a^i = \boldsymbol{H}_a^{i+1} - \boldsymbol{f}_{i+1}(\boldsymbol{H}_a^{i+1}) + g_{i+1}^2 \nabla_{\boldsymbol{H}_a^{i+1}} \log p(\boldsymbol{H}_a^{i+1}|\boldsymbol{Y}) \\ + g_{i+1}\boldsymbol{Z},\end{aligned} \tag{21}$$

where $\boldsymbol{H}_a^i = \boldsymbol{H}_a(t)|_{t=\frac{i}{N}}$, $\boldsymbol{f}_i(\boldsymbol{H}_a^i) = \boldsymbol{f}(\boldsymbol{H}_a(t), t)|_{t=\frac{i}{N}}$, $g_i = g(t)|_{t=\frac{i}{N}}$, and $\boldsymbol{Z} \sim \mathcal{N}(\boldsymbol{Z}; \boldsymbol{0}, \boldsymbol{I})$. Given the known drift coefficient $\boldsymbol{f}(\boldsymbol{H}_a, t) = \boldsymbol{0}$ and diffusion coefficient $g(t) = \sqrt{\frac{\mathrm{d}(\sigma^2(t))}{\mathrm{d}t}}$ in SMLD, Eq. (21) is equivalent to:

$$\begin{aligned}\tilde{\boldsymbol{H}}_a^i = \boldsymbol{H}_a^{i+1} + \left(\sigma_{i+1}^2 - \sigma_i^2\right) \nabla_{\boldsymbol{H}_a^{i+1}} \log p(\boldsymbol{H}_a^{i+1}|\boldsymbol{Y}) \\ + \sqrt{\sigma_{i+1}^2 - \sigma_i^2}\boldsymbol{Z},\end{aligned} \tag{22}$$

where $\sigma_i = \sigma(t)|_{t=\frac{i}{N}}$. Then, starting from a random initialization $\boldsymbol{H}_a^N \sim \mathcal{N}(\boldsymbol{H}_a^N; \boldsymbol{0}, \sigma_{\max}^2 \boldsymbol{I})$, the iteration process for the SMLD-based predictor is shown in Fig. 4. The corresponding discretized form of the score function is given as:

$$\nabla_{\boldsymbol{H}_a^i} \log p(\boldsymbol{H}_a^i) = \left[\boldsymbol{s}_{\boldsymbol{\theta}^*}\left(\boldsymbol{h}_1^i, \frac{i}{N}\right), \ldots, \boldsymbol{s}_{\boldsymbol{\theta}^*}\left(\boldsymbol{h}_{K_a}^i, \frac{i}{N}\right)\right]^{\mathrm{T}}, \tag{23a}$$

$$\begin{aligned}\nabla_{\boldsymbol{H}_a^i} \log p\left(\boldsymbol{Y}|\boldsymbol{H}_a^i\right) = \boldsymbol{V}\boldsymbol{\Sigma}^{\mathrm{T}}\left(\sigma_n^2 \boldsymbol{I} + \sigma_i^2 \boldsymbol{\Sigma}\boldsymbol{\Sigma}^{\mathrm{T}}\right)^{-1} \\ \times \left(\boldsymbol{U}^{\mathrm{T}}\boldsymbol{Y} - \boldsymbol{\Sigma}\boldsymbol{V}^{\mathrm{T}}\boldsymbol{H}_a^i\right),\end{aligned} \tag{23b}$$

$$\begin{aligned}\nabla_{\boldsymbol{H}_a^i} \log p\left(\boldsymbol{H}_a^i|\boldsymbol{Y}\right) = \lambda_h \nabla_{\boldsymbol{H}_a^i} \log p\left(\boldsymbol{Y}|\boldsymbol{H}_a^i\right) \\ + \nabla_{\boldsymbol{H}_a^i} \log p\left(\boldsymbol{H}_a^i\right).\end{aligned} \tag{23c}$$

In Eq. (23b), we apply singular value decomposition (SVD) to $\boldsymbol{S}_a = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^{\mathrm{T}}$ within Eq. (20) to avoid non-diagonal matrix inversions, thereby reducing computational complexity. Additionally, a scaling parameter $\lambda_h = 2.5$, as suggested in [46], is empirically chosen to balance the effects of the likelihood score and prior score, enhancing convergence performance.

However, the numerical predictor in Eq. (22) may introduce approximation errors when solving the continuous SDE in a discrete manner. To correct the errors, a score-based Markov Chain Monte Carlo (MCMC) method is used as the *corrector*.

Authorized licensed use limited to: UNIVERSITY OF SOUTHAMPTON. Downloaded on January 09,2026 at 17:30:07 UTC from IEEE Xplore. Restrictions apply.

This method corrects the marginal distribution of the estimated sample during iterations. A well-known MCMC approach is the ALD, which employs the following iterative process:

$$\boldsymbol{H}_a^i = \tilde{\boldsymbol{H}}_a^i + \epsilon_i \nabla_{\tilde{\boldsymbol{H}}_a^i} \log p(\tilde{\boldsymbol{H}}_a^i|\boldsymbol{Y}) + \sqrt{2\epsilon_i}\boldsymbol{Z}, \tag{24}$$

where $\epsilon_i = 2\sigma_i \left(r\|\boldsymbol{Z}\|_2 / \|\nabla_{\tilde{\boldsymbol{H}}_a^i} \log p(\tilde{\boldsymbol{H}}_a^i|\boldsymbol{Y})\|_2\right)^2$ is the step size in the i-th iteration of the ALD algorithm [38]. Additionally, r is an empirically chosen hyperparameter, set to $r = 0.3$ in our simulations. Existing ALD-based studies on CE [27], DD [31], and JCEDD [33] can be regarded as corrector-only samplers with potential for performance improvement. Therefore, we address the JCEDD problem using the PC sampler illustrated in Fig. 4.

Algorithm 1 PC Sampler-Based Channel Estimation

Input: $\boldsymbol{S}_a$, $\boldsymbol{Y}$, σ_n^2, pre-trained denoising network $\boldsymbol{s}_{\boldsymbol{\theta}^*}$, step number N, noise scheduling function $\sigma(t)$, and hyperparameter λ_h;

1: **Initialize:** $\boldsymbol{H}_a^N \sim \mathcal{N}(\boldsymbol{H}_a^N; \boldsymbol{0}, \sigma_N^2 \boldsymbol{I})$;
2: **Compute:** $\boldsymbol{S}_a = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^{\mathrm{T}}$;
3: **for** $i = N$ to 1 **do**
4: Calculate $\nabla_{\boldsymbol{H}_a^i} \log p\left(\boldsymbol{H}_a^i|\boldsymbol{Y}\right)$ via (23);
5: Draw $\boldsymbol{Z} \sim \mathcal{N}(\boldsymbol{Z}; \boldsymbol{0}, \boldsymbol{I})$;
6: $\boldsymbol{H}_a^{i-1} = \boldsymbol{H}_a^i + \left(\sigma_i^2 - \sigma_{i-1}^2\right) \nabla_{\boldsymbol{H}_a^i} \log p(\boldsymbol{H}_a^i|\boldsymbol{Y}) + \sqrt{\sigma_i^2 - \sigma_{i-1}^2}\boldsymbol{Z}$; *% Predictor: line 6*
7: **for** $j = 1$ to Q **do**
8: Draw $\boldsymbol{Z} \sim \mathcal{N}(\boldsymbol{Z}; \boldsymbol{0}, \boldsymbol{I})$;
9: Calculate $\nabla_{\boldsymbol{H}_a^{i-1}} \log p\left(\boldsymbol{H}_a^{i-1}|\boldsymbol{Y}\right)$ via (23);
10: $\epsilon_{i-1} = 2\sigma_{i-1}\left(r\|\boldsymbol{Z}\|_2 / \|\nabla_{\boldsymbol{H}_a^{i-1}} \log p(\boldsymbol{H}_a^{i-1}|\boldsymbol{Y})\|_2\right)^2$;
11: $\boldsymbol{H}_a^{i-1} = \boldsymbol{H}_a^{i-1} + \epsilon_{i-1} \nabla_{\boldsymbol{H}_a^{i-1}} \log p(\boldsymbol{H}_a^{i-1}|\boldsymbol{Y}) + \sqrt{2\epsilon_{i-1}}\boldsymbol{Z}$;
12: **end for** *% Corrector: lines 7-12*
13: **end for**

Output: Estimated channel $\hat{\boldsymbol{H}}_a = \boldsymbol{H}_a^0$.

4) Overall CE Algorithm: In practical implementations, each iteration of the PC sampler includes one prediction step and Q correction steps, where Q ranges from 1 to 3. The complete CE algorithm is listed as Algorithm 1. Compared to traditional Bayesian algorithms, our method derives prior distribution knowledge through network learning, which avoids the mismatches associated with manually specified prior distributions. Moreover, unlike purely data-driven or model-driven deep learning algorithms based on supervised learning, this method does not require training separate networks for different numbers of active UEs and varying pilot lengths, offering improved generalization and interpretability.

C. Data Detection

Upon reviewing the formulas (2) and (5), the DD problem can be characterized as $\boldsymbol{Y}_d = \boldsymbol{X}_a\boldsymbol{H}_a + \boldsymbol{W}_d$. By transposing the formula and representing the model in a real-valued form, the signal transmission model can be reformulated as:

$$\mathring{\boldsymbol{Y}}_d = \mathring{\boldsymbol{H}}_a\mathring{\boldsymbol{X}}_a + \mathring{\boldsymbol{W}}_d, \tag{25}$$

where $\mathring{\boldsymbol{Y}}_d$, $\mathring{\boldsymbol{H}}_a$, $\mathring{\boldsymbol{X}}_a$ and $\mathring{\boldsymbol{W}}_d$ are the transposed counterparts of the corresponding variables in real-valued form. Similar to the CE problem, assuming that the channel $\mathring{\boldsymbol{H}}_a$ is known, data symbols can be sampled from the posterior distribution $p(\mathring{\boldsymbol{X}}_a|\mathring{\boldsymbol{Y}}_d)$ by solving the SDE, which also involves calculating the prior score and the likelihood score of the data. Since the data symbols are uniformly sampled from a QAM constellation, the prior score $\nabla_{\mathring{\boldsymbol{X}}_a^i} \log p\left(\mathring{\boldsymbol{X}}_a^i\right)$ in the i-th iteration step can be calculated using Tweedie's formula [45], i.e.,

$$\nabla_{\mathring{\boldsymbol{X}}_a^i} \log p(\mathring{\boldsymbol{X}}_a^i) = \frac{\mathbb{E}\left\{\mathring{\boldsymbol{X}}_a^0|\mathring{\boldsymbol{X}}_a^i\right\} - \mathring{\boldsymbol{X}}_a^i}{\tau_i^2}, \tag{26}$$

where $\tau_i = \tau(t)|_{t=\frac{i}{N}}$ is the scheduled perturbation noise for data symbol, and $\tau(t) = \tau_{\min}\left(\frac{\tau_{\max}}{\tau_{\min}}\right)^t$ is a predefined scheduling function. Additionally, the posterior distribution can be derived using Bayes' rule as: $p(\mathring{\boldsymbol{X}}_a^0|\mathring{\boldsymbol{X}}_a^i) = \frac{p(\mathring{\boldsymbol{X}}_a^i|\mathring{\boldsymbol{X}}_a^0)p(\mathring{\boldsymbol{X}}_a^0)}{\sum_{\mathring{\boldsymbol{X}}_a^0 \sim \mathcal{X}} p(\mathring{\boldsymbol{X}}_a^i|\mathring{\boldsymbol{X}}_a^0)p(\mathring{\boldsymbol{X}}_a^0)}$, where $p(\mathring{\boldsymbol{X}}_a^0) = \frac{1}{|\mathcal{X}|_c}$. Then, the posterior mean is derived as

$$\mathbb{E}\left\{\mathring{\boldsymbol{X}}_a^0|\mathring{\boldsymbol{X}}_a^i\right\} = \frac{\sum_{\mathring{\boldsymbol{X}}_a^0 \sim \mathcal{X}} \mathring{\boldsymbol{X}}_a^0 p(\mathring{\boldsymbol{X}}_a^i|\mathring{\boldsymbol{X}}_a^0)}{\sum_{\mathring{\boldsymbol{X}}_a^0 \sim \mathcal{X}} p(\mathring{\boldsymbol{X}}_a^i|\mathring{\boldsymbol{X}}_a^0)}. \tag{27}$$

By substituting (27) into (26), the final prior score the data symbols at time step i can be expressed as

$$\nabla_{\mathring{\boldsymbol{X}}_a^i} \log p(\mathring{\boldsymbol{X}}_a^i) = \frac{1}{\tau_i^2}\left(\frac{\sum_{\mathring{\boldsymbol{X}}_a^0 \sim \mathcal{X}} \mathring{\boldsymbol{X}}_a^0 e^{-\frac{\|\mathring{\boldsymbol{X}}_a^i - \mathring{\boldsymbol{X}}_a^0\|^2}{2\tau_i^2}}}{\sum_{\mathring{\boldsymbol{X}}_a^0 \sim \mathcal{X}} e^{-\frac{\|\mathring{\boldsymbol{X}}_a^i - \mathring{\boldsymbol{X}}_a^0\|^2}{2\tau_i^2}}} - \mathring{\boldsymbol{X}}_a^i\right). \tag{28}$$

For the likelihood score, the equivalent likelihood function can be derived from the transmission model (25) as:

$$\nabla_{\mathring{\boldsymbol{X}}_a^i} \log p\left(\mathring{\boldsymbol{Y}}_d|\mathring{\boldsymbol{X}}_a^i\right) = \boldsymbol{V}_h\boldsymbol{\Sigma}_h^{\mathrm{T}}\left(\sigma_n^2\boldsymbol{I}_{2M} + \tau_i^2\boldsymbol{\Sigma}_h\boldsymbol{\Sigma}_h^{\mathrm{T}}\right)^{-1} \times \left(\boldsymbol{U}_h^T\mathring{\boldsymbol{Y}}_d - \boldsymbol{\Sigma}_h\boldsymbol{V}_h^{\mathrm{T}}\mathring{\boldsymbol{X}}_a^i\right), \tag{29}$$

where we have used the SVD $\mathring{\boldsymbol{H}}_a = \boldsymbol{U}_h\boldsymbol{\Sigma}_h\boldsymbol{V}_h^{\mathrm{T}}$ to reduce the complexity during the iteration. Finally, the posterior score of the data symbols can be calculated as

$$\nabla_{\mathring{\boldsymbol{X}}_a^i} \log p\left(\mathring{\boldsymbol{X}}_a^i|\mathring{\boldsymbol{Y}}_d\right) = \lambda_x \nabla_{\mathring{\boldsymbol{X}}_a^i} \log p\left(\mathring{\boldsymbol{Y}}_d|\mathring{\boldsymbol{X}}_a^i\right) + \nabla_{\mathring{\boldsymbol{X}}_a^i} \log p(\mathring{\boldsymbol{X}}_a^i), \tag{30}$$

where $\lambda_x = 2.5$ is an empirically chosen hyperparameter. The DD algorithm can also be derived based on PC-sampler. Due to space constraint, the detailed DD algorithm is omitted, but its basic iterative procedure is similar to that of Algorithm 1.

D. Overall Algorithm

In the CE and DD algorithms above, the data $\boldsymbol{X}_a$ is assumed to be known during CE, while the channel $\boldsymbol{H}_a$ is assumed to be known during DD. In practical transmission, only the pilot $\boldsymbol{P}_a$ and the received signals $\boldsymbol{Y}$ are known. Therefore, we design an iterative algorithm to perform JCEDD. The

Authorized licensed use limited to: UNIVERSITY OF SOUTHAMPTON. Downloaded on January 09,2026 at 17:30:07 UTC from IEEE Xplore. Restrictions apply.

TABLE I
COMPLEXITY OF ALGORITHM 2

Operation		Lines	Complexity/Each Time	Number of Times
Initialization		Line 1	$\mathcal{O}\left(K_a L_p^2 M^3 + L_p K_a^2 M^3 + \frac{2}{3} L_p^3 M^3 + 2K_a M^2 L_p^2\right)$	1
SVD	Channel	Lines 2 and 13	$\mathcal{O}\left(2LK_a^2 + K_a^3\right)$	$1 + \lfloor i^*/N_{\text{update}} \rfloor$
	Data	Line 7	$\mathcal{O}\left(2K_a M^2 + M^3\right)$	$\lfloor i^*/N_{\text{update}} \rfloor$
Posterior Score Calculation	Channel	Lines 4-5	$\mathcal{O}(L^2 M + K_a^2 M + K_a F_{\text{net}})$	$i^*(1+Q)$
	Data	Lines 10-11	$\mathcal{O}(M^2 L_d + K_a^2 L_d + 2K_a L_d \lvert\mathcal{X}\rvert_c)$	$\lfloor i^*/N_{\text{update}} \rfloor (1+Q)N_X$
Overall Complexity			Dominated by Lines 4-5 and Lines 10-11.	

Algorithm 2 PC Sampler-Based JCEDD Algorithm

Input: $\boldsymbol{S}_a$, $\boldsymbol{Y}$, σ_n^2, pre-trained denoising network $\boldsymbol{s}_{\boldsymbol{\theta}^*}$, SDE noise scheduling functions $\sigma(t)$ and $\tau(t)$, step numbers N_H and N_D, hyper-parameters λ_h and λ_x;

1: **Initialize:** Acquire initial CE $\hat{\boldsymbol{H}}_a$ via LMMSE, initial DD $\hat{\boldsymbol{X}}_a$ via ZF, and start point i^* via (31);
2: **Calculate:** $\boldsymbol{S}_a = \left[\boldsymbol{P}_a^{\mathrm{T}}, \hat{\boldsymbol{X}}_a^{\mathrm{T}}\right]^{\mathrm{T}} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^{\mathrm{T}}$;
3: **for** $i = i^*$ to 1 **do**
4: Calculate $\nabla_{\boldsymbol{H}_a^i} \log p\left(\boldsymbol{H}_a^i | \boldsymbol{Y}\right)$ via (23);
5: $\boldsymbol{H}_a^{i-1} = \text{PC-sampler}\left(i, \boldsymbol{H}_a^i, \nabla_{\boldsymbol{H}_a^i} \log p\left(\boldsymbol{H}_a^i | \boldsymbol{Y}\right)\right)$;
6: **if** $i \% N_{\text{update}} == 0$ **then**
7: Calculate $\boldsymbol{H}_a^{i-1} = \boldsymbol{U}_h \boldsymbol{\Sigma}_h \boldsymbol{V}_h^{\mathrm{T}}$;
8: Initialize $\mathring{\boldsymbol{X}}_a^{N_X} \sim \mathcal{N}\left(\mathring{\boldsymbol{X}}_a^{N_X}; \boldsymbol{0}, \tau_{\max}^2 \boldsymbol{I}\right)$;
9: **for** $j = N_X$ to 1 **do**
10: Calculate $\nabla_{\mathring{\boldsymbol{X}}_a^j} \log p\left(\mathring{\boldsymbol{X}}_a^j | \mathring{\boldsymbol{Y}}_d\right)$ via (30);
11: $\boldsymbol{X}_a^{j-1} = \text{PC-sampler}\left(j, \boldsymbol{X}_a^j, \nabla_{\mathring{\boldsymbol{X}}_a^j} \log p\left(\mathring{\boldsymbol{X}}_a^j | \mathring{\boldsymbol{Y}}_d\right)\right)$;
12: **end for**
13: Calculate $\boldsymbol{S}_a = \left[\boldsymbol{P}_a^{\mathrm{T}}, \left(\boldsymbol{X}_a^0\right)^{\mathrm{T}}\right]^{\mathrm{T}} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^{\mathrm{T}}$;
14: **end if**
15: **end for**

Output: Estimated channel and data $\hat{\boldsymbol{H}}_a = \boldsymbol{H}_a^0$ and $\hat{\boldsymbol{X}}_a = \boldsymbol{X}_a^0$.

proposed PC sampler-based JCEDD method is depicted in Fig. 5, and the algorithmic framework is summarized as Algorithm 2.

The issues we need to address are: 1) A typical SDE sampling algorithm starts by sampling from random noise, resulting in a high number of iterations. 2) In the existing ALD-based JCEDD [33], each CE iteration is followed by one DD iteration. Due to the different distributions of the channel and data symbols, their convergence speeds are inconsistent, causing the DD to fall into local optima before the completion of the CE iterations. We address these issues by the following improvements in the proposed sampling mechanism.

- We conduct the LMMSE estimation [21] of the channel $\boldsymbol{H}_a$ using $\boldsymbol{Y}_p = \boldsymbol{P}_a \boldsymbol{H}_a + \boldsymbol{W}_p$ and determine the perturbation noise variance that matches the estimated channel variance as the starting point for SDE sampling. Let the estimated covariance matrix of the channel $\boldsymbol{h}_a = \text{vec}\left(\boldsymbol{H}_a^{\mathrm{T}}\right) \in \mathbb{C}^{K_a M}$ be $\boldsymbol{R}_\Delta$. Then the starting point i^* of the SDE iteration is obtained by identifying σ_i that is the closest to the average channel variance according to:

$$i^* = \arg\min_{\forall i} \left| \sigma_i - \frac{1}{K_a M} \sum_{l=1}^{K_a M} [\boldsymbol{R}_\Delta]_{l,l} \right|. \tag{31}$$

By reshaping the estimated $\hat{\boldsymbol{h}}_a$ into matrix $\hat{\boldsymbol{H}}_a$, the initial DD result can be obtained by applying zero forcing (ZF) to (25), i.e., $\hat{\boldsymbol{X}}_a = \boldsymbol{Y}_d \boldsymbol{H}_a^{\mathrm{H}} \left(\boldsymbol{H}_a \boldsymbol{H}_a^{\mathrm{H}}\right)^{-1}$.
- We design an asynchronous alternating iteration mechanism. For channel datasets with complex distributions, the calculation of posterior score is more complicated than DD that relies solely on model-based iterations with closed-form expression. We update the DD results after every N_{update} channel iterations (Line 6 in Algorithm 2). To ensure the optimal DD results under the current channel error, we conduct N_X steps of PC sampler each time the data symbols are updated (Lines 9-12).

E. Computational Complexity Analysis

The complexity analysis of Algorithm 2 is detailed in Tab. I. The channel is initialized using the LMMSE method before the iterative process begins. For the channel PC-sampler, $i^\star$ iterations are necessary. Each PC iteration consists of one predictor step and Q corrector steps, resulting in a total of $i^\star(1+Q)$ computations of posterior scores, as described in Lines 4-5 of Algorithm 2. Furthermore, data is updated after every N_{update} channel iterations, as outlined in Lines 6-14.

During each data update, N_X data iterations are conducted using another PC sampler. Each iteration includes one predictor step and Q corrector steps, leading to a total of $\left\lfloor \frac{i^\star}{N_{\text{update}}} \right\rfloor (1+Q) N_X$ computations of posterior scores for data. Additionally, SVD is utilized to reduce computational complexity. Specifically, $1 + \left\lfloor \frac{i^\star}{N_{\text{update}}} \right\rfloor$ SVD operations are performed for the channel in Lines 2 and 13, and $\left\lfloor \frac{i^\star}{N_{\text{update}}} \right\rfloor$ SVD operations are needed for the data in Line 7.

Specifically, computational complexity of calculating each channel likelihood score function is $\mathcal{O}(L^2 M + K_a^2 M)$, while the complexity of the channel prior score function is $\mathcal{O}(K_a F_{\text{net}})$, where F_{net} represents the computational cost of one time inference of the score network $\boldsymbol{s}_{\boldsymbol{\theta}}(\boldsymbol{h}(t), t)$. The data likelihood score's complexity is $\mathcal{O}(M^2 L_d + K_a^2 L_d)$, and the data prior score's complexity is $\mathcal{O}(2K_a L_d \lvert\mathcal{X}\rvert_c)$.

For typical simulation parameters, e.g., $N_{\text{update}} = 50$, $\left\lfloor \frac{i^\star}{N_{\text{update}}} \right\rfloor = 5 \sim 15$, $Q = 3$, $N_X = 1500$ and $N_H = 1500$, the

Authorized licensed use limited to: UNIVERSITY OF SOUTHAMPTON. Downloaded on January 09,2026 at 17:30:07 UTC from IEEE Xplore. Restrictions apply.

overall complexity of Algorithm 2 is primarily determined by the score function computations in lines 4-5 and 10-11. Specifically, the neural network NCSN++ [38], employed to calculate the channel's prior score function, markedly impacts the inference complexity. For complex channel distributions, utilizing a moderately complex network can balance performance with computational demands, providing an avenue for algorithm optimization. Additionally, given that the proposed approach includes multiple PC-sampler iterations, the use of higher-order SDE or ordinary differential equation (ODE) solvers [47], [48] is essential for further reducing complexity in engineering applications.

VI. Numerical Results

In this section, we first detail the experimental setup, including simulation parameters, network configurations, channel datasets, training conditions, and comparative algorithms. We then assess the performance of the Transformer-based AUD as well as the generative diffusion models driven JCEDD.

A. Experimental Setups

We consider a typical uplink random access scenario with a total of $K=128$ potential users,[1] where each user's activity follows a Bernoulli distribution with an active probability of 0.1. The BS employs a UPA with $M_x = M_y = 8$ antennas on each dimension, while each user operates with a single antenna. Each element of the pilot sequence $\tilde{\boldsymbol{p}}_k,\ \forall k$, and the data symbols $\boldsymbol{x}_k$ are randomly sampled from 4QAM constellations. The maximum pilot sequence length is $L_p^{\max} = 28$. We consider a 3GPP-compliant channel model [39] generated using Quadriga [49], simulating the 3GPP 38.901 UMa NLOS scenario at a carrier frequency of 3.5 GHz. The minimum pilot sequence length is $L_p^{\min} = 8$ for Rayleigh channel, and $L_p^{\min} = 10$ for 3GPP channel, since 3GPP channel is more complicated and it needs more pilots to estimate AUS.

1) VPL-AUDNet Setup: A total of 64,000 channel samples are generated using Quadriga. For each training batch, the pilot lengths for the UEs are randomly selected from $[L_p^{\min}, L_p^{\max}]$. In addition to the input preprocessing layer and output layer, the VPL-AUDNet contains 5 heterogeneous Transformer encoder layers, each featuring a hidden state sequence length of $d = 512$ and 8 attention heads. Training is conducted over 200 epochs, with the learning rate warming up to 10^{-4} during the initial 15 epochs and then decaying to 0.1 of this original value every 75 epochs. The batch size is set to 128.

[1]Future networks are expected to support millions of devices per square kilometer. However, such high user density can only be achieved within specific time-frequency resources. For anticipated 6G massive communications, it is generally necessary to divide limited resources into different time-frequency resource groups. For example, within the coverage area of a single macro base station (approximately $S = \pi r^2 = \pi \times 0.5^2 = 0.7854\,\text{km}^2$), 128 users can be grouped to share the same resource block, such as 12 subcarriers and 14 OFDM symbols. Considering a system with 1 frame = 10 ms = 140 OFDM symbols and 512 resource blocks (providing approximately 92.16 MHz bandwidth for 15 kHz subcarrier spacing), the system can theoretically support up to $512\times10\times128 = 655,360$ UEs, corresponding to a user density of $\frac{655,360\ \text{UEs/sector}\times3\ \text{sectors}}{0.7854\,\text{km}^2} \approx 2.5\times10^6\ \text{UEs/km}^2$ for the $100\,\text{MHz}\times10\,\text{ms}$ time-frequency resources, effectively meeting the massive communication requirement.

2) Channel Prior Score Network Setup: The NSCN++ architecture [38] is employed to model the prior score function $\boldsymbol{s}_{\boldsymbol{\theta}}(\boldsymbol{h}(t), t)$. A 3GPP training dataset consisting of 160,000 channel samples is utilized. The time step t is uniformly sampled from the interval $[0, 1]$, and the perturbation noise variance is determined using $\sigma(t) = \sigma_{\min}\left(\frac{\sigma_{\max}}{\sigma_{\min}}\right)^t$, with $\sigma_{\max} = 30$ and $\sigma_{\min} = 0.01$. The network is trained with a learning rate of 10^{-5} and a batch size of 64, over 20 epochs.

During the testing phase, the reverse SDE process for CE is discretized into $N_H = 1,500$ steps. Given that an LMMSE initialization is employed, only i^* of the N_H steps are executed. The reverse SDE process for DD is also discretized into $N_X = 1,500$ steps, using a noise scheduling function defined as $\tau(t) = \tau_{\min}\left(\frac{\tau_{\max}}{\tau_{\min}}\right)^t$, where $\tau_{\max} = 1$ and $\tau_{\min} = 0.01$. Additionally, the hyperparameters are set to $\lambda_h = \lambda_x = 2.5$.

3) Evaluation Metrics: We utilize activity error probability (AEP), normalized mean square error (NMSE) and bit error rate (BER) to evaluate the performance of AUD, CE and DD, respectively. These metrics are defined respectively as $\text{AEP} = \mathbb{E}\left\{\frac{1}{K}\sum_{k=1}^{K}|\alpha_k - \hat{\alpha}_k|\right\}$, $\text{NMSE} = \mathbb{E}\left\{\frac{\|\boldsymbol{H}_a - \hat{\boldsymbol{H}}_a\|^2}{\|\boldsymbol{H}_a\|^2}\right\}$ and $\text{BER} = \mathbb{E}\left\{\frac{E_b}{K_a L_d \log_2|\mathcal{X}|_c}\right\}$, where E_b denotes the total number of error bits for active UEs.

4) Comparative Schemes: For AUD, the proposed method is compared with two baseline algorithms: AMP [13] and OMP [12]. Additionally, we also assess the performance of the VPL-AUDNet scheme trained with a fixed pilot length. For CE and DD, the following algorithms are included in the comparisons:

- **LS-CE & Perfect-Data:** The data $\boldsymbol{X}_a$ is perfectly known, and jointly utilize pilots $\boldsymbol{P}_a$ and data $\boldsymbol{X}_a$ for LS CE;
- **Perfect-CSI & ZF-DD:** The CSI $\boldsymbol{H}_a$ is perfectly known, and employ ZF for DD;
- **Perfect-CSI & OAMP-DD:** The CSI $\boldsymbol{H}_a$ is perfectly known, and utilize OAMP [21] for DD;
- **Pilot LS-CE & ZF-DD:** Utilize pilots to perform LS CE, and use the estimated channel for ZF DD;
- **Pilot LS-CE & OAMP-DD:** Utilize pilots to perform LS CE, and use the estimated channel for OAMP [21] DD;
- **Iter LMMSE-CE & OAMP-DD:** Employ LMMSE for CE and OAMP for DD, with CE and DD iteratively executed to enhance the performance of each other [21];
- **Iter Langevin CE & DD:** Conduct iterative CE and DD using ALD [33];
- **SDE-CE & Perfect-Data:** The data $\boldsymbol{X}_a$ is perfectly known, and apply SDE solver (i.e., PC sampler) for CE, which corresponds to Algorithm 1;
- **Perfect-CSI & SDE-DD:** The CSI $\boldsymbol{H}_a$ is known, and apply SDE solver for DD;
- **Our Iter SDE CE & DD:** The proposed Algorithm 2.

B. Performance of AUD

Fig. 6(a) illustrates the AUD performance of various schemes as the functions of the transmission pilot length, L_p^{test}, under i.i.d. Rayleigh channels, where the channel between each potential UE and the BS is modeled as $\boldsymbol{h}_k \sim \mathcal{N}_c(\boldsymbol{h}_k; \boldsymbol{0}, \boldsymbol{I}_M),\ \forall k$. The simulation is conducted under the

Authorized licensed use limited to: UNIVERSITY OF SOUTHAMPTON. Downloaded on January 09,2026 at 17:30:07 UTC from IEEE Xplore. Restrictions apply.

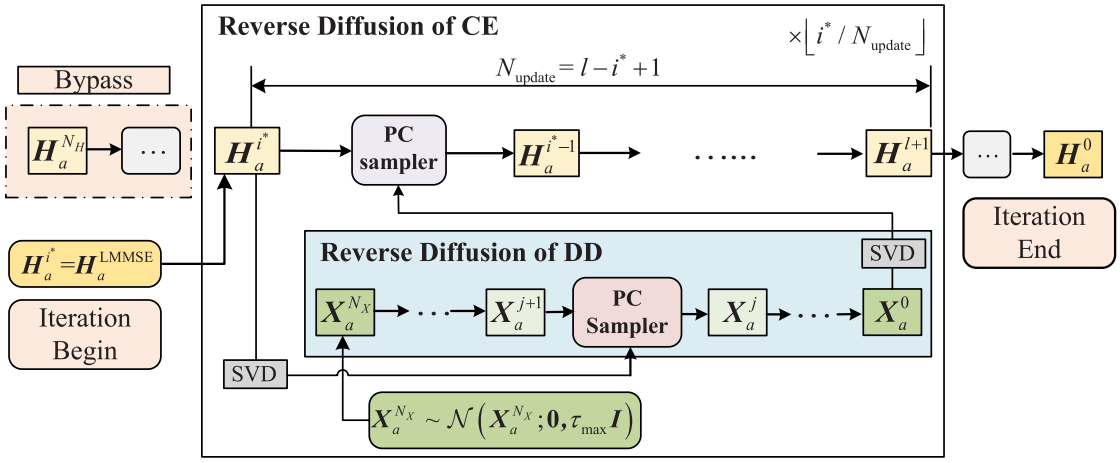
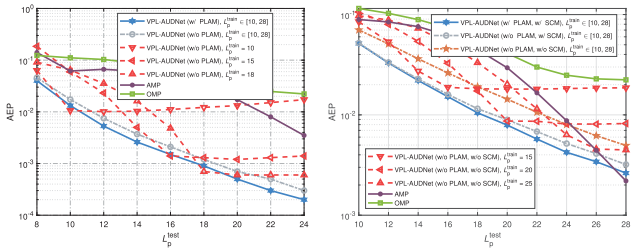


Fig. 5. Block diagram of the proposed PC sampler-based JCEDD algorithm.



(a) Rayleigh channel (b) 3GPP channel

Fig. 6. AEP performance with respect to the transmission pilot length of different AUD schemes.

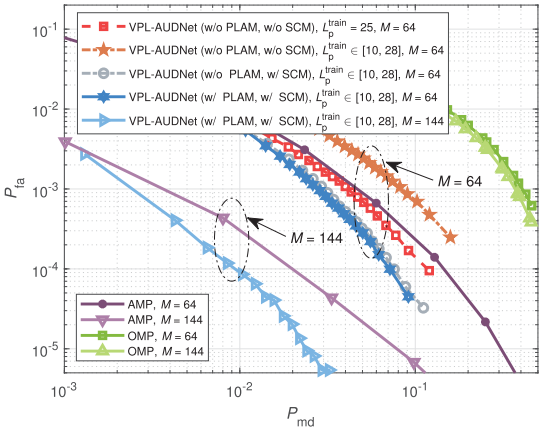


Fig. 7. P_{fa}-P_{md} performance of different AUD schemes under 3GPP channel.

received SNR $=\frac{\mathbb{E}\{\|\boldsymbol{SH}\|^2\}}{\mathbb{E}\{\|\boldsymbol{W}\|^2\}}=10$ dB. The results indicate that that the VPL-AUDNet-based schemes outperform traditional CS algorithms based on AMP and OMP, in terms of AEP. Furthermore, the VPL-AUDNet models trained with fixed pilot lengths and without the PLAM module, e.g., VPL-AUDNet (w/o PLAM), $L_p^{\text{train}}=15$, exhibit degraded performance when tested on pilot lengths different from the training pilot length.[2] Conversely, VPL-AUDNet (w/o PLAM), $L_p^{\text{train}}\in[10,28]$ exhibit better generalization performance. Moreover, the proposed VPL-AUDNet (w/PLAM), $L_{\text{train}}\in[8,24]$, outperforms VPL-AUDNet (w/o PLAM), $L_{\text{train}}\in[8,24]$, demonstrating the effectiveness of the PLAM. Additionally, under i.i.d. Rayleigh channels, the proposed VPL-AUDNet scheme achieves an AEP performance close to 10^{-2} even when the pilot number is less than the average number of UEs ($L_p^{\text{test}}\leq 12$), whereas the AMP and OMP algorithms exhibit a AEP of 10^{-1}. These results demonstrate that the proposed scheme achieves optimal overall performance across various pilot lengths within a single model, thereby confirming its effectiveness.

Fig. 6(b) compares the AUD performance of various schemes as the functions of the number of transmission pilots L_p^{test} under 3GPP channels, given a received SNR of 10 dB. The results show that the performance curves of the different schemes are similar to the trends observed under Rayleigh channels. Compared to traditional CS algorithms like AMP and OMP, the VPL-AUDNet-based scheme achieves better performance with significantly reduced pilot length by avoiding completely CE. Additionally, a comparison between the curves of VPL-AUDNet (w/o PLAM, w/SCM), $L_p^{\text{train}}=[10,28]$, and VPL-AUDNet (w/o PLAM, w/o SCM), $L_p^{\text{train}}=[10,28]$, reveals that the SCM significantly enhances AEP performance under 3GPP channels. Moreover, VPL-AUDNet (w/PLAM, w/SCM), $L_p^{\text{train}}=[10,28]$ scheme attains optimal performance at low pilot numbers.

Fig. 7 depicts the curves for the probability of false alarm (P_{fa}) and the probability of missed detection (P_{md}) for various schemes under 3GPP channels. Simulations are conducted with a received SNR of 10 dB and a test pilot length of $L_p^{\text{test}}=25$. All VPL-AUDNet models are trained with an antenna number of $M=64$. The results indicate that the proposed VPL-AUDNet (w/PLAM, w/SCM), $L_p^{\text{train}}=[10,28]$, achieves an optimal trade-off between P_{fa} and P_{md}. Moreover, testing the model trained with $M=64$ antennas on the channels with $M=144$ antennas results in further improvements in P_{fa}-P_{md} performance, demonstrating the scheme's robust

[2]For models trained with a pilot length of L_p^{train}, if the test length is shorter than the training length, the received signals are padded with zeros to align with L_p^{train} before being input into the VPL-AUDNet network. Conversely, if the test length exceeds the training length, the received signals are truncated to L_p^{train} before network input.

Authorized licensed use limited to: UNIVERSITY OF SOUTHAMPTON. Downloaded on January 09,2026 at 17:30:07 UTC from IEEE Xplore. Restrictions apply.

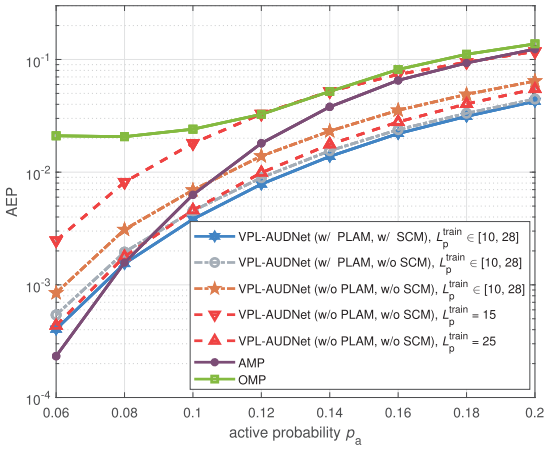


Fig. 8. AEP performance with respect to the active probability of different AUD schemes under the 3GPP channel.

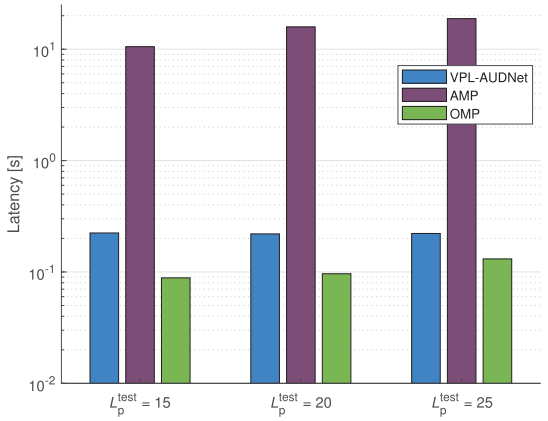
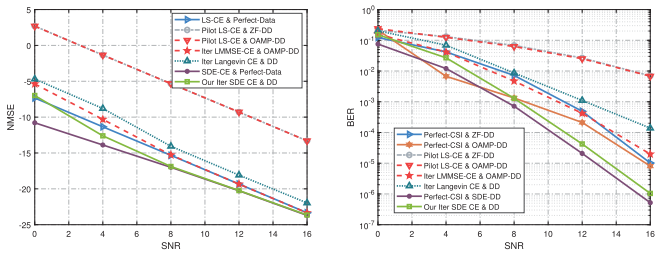
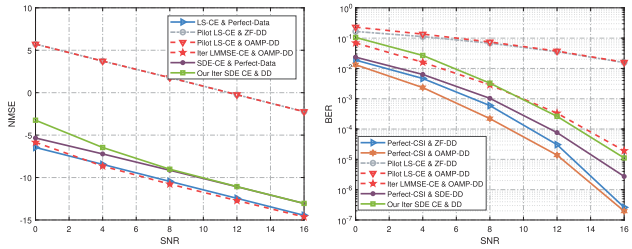


Fig. 9. Computation latency of AUD schemes under different pilot lengths.



(a) NMSE (b) BER

Fig. 10. NMSE and BER performance with respect to the received SNR for different CE and DD algorithms under 3GPP channels.



(a) (b)

Fig. 11. NMSE and BER performance with respect to the received SNR for different CE and DD algorithms under the MIMO Rayleigh channel.

generalization capability across varying antenna configurations. In contrast, the performance gains for OMP related to an increased number of antennas are less significant.

Fig. 8 compares the AEP performance curves of different schemes as a function of active probability of UEs p_a under the 3GPP channel. The simulation assumes a received SNR of 10 dB, a total potential number of UEs of 128, a user activation probability of $[0.06, 0.2]$, and a test pilot length of $L_p^{\text{test}} = 25$. The simulation results indicate that although AMP achieves marginally better AEP performance when the active probability is low, its performance deteriorates more rapidly as the active probability increases. In contrast, our scheme based on VPL-AUDNet exhibits a more gradual performance degradation, indicating better generalization capabilities in adapting to variations in activity probability.

Fig. 9 illustrates the computational latency required to process a batch of 128 samples across different schemes with varying pilot lengths. The results show that the computational latency of the VPL-AUDNet scheme remains approximately constant regardless of the pilot length due to a specialized receiver structure accommodating variable pilot length inputs. In contrast, as the pilot length increases, compressed sensing algorithms based on AMP or OMP experience increased computational complexity because of the corresponding growth in the dimension of the sensing matrix. The overall computational delay of the proposed VPL-AUDNet scheme is lower than that of the AMP scheme but higher than that of the OMP scheme. Nevertheless, the AUD performance of the VPL-AUDNet scheme is generally superior to that of OMP, highlighting the advantages of a neural network-based approach.

C. Performance of JCEDD

In this subsection's simulations, unless otherwise specified, the BS is equipped with $M = 64$ antennas, and a total of $K_a = 12$ active UEs access the system simultaneously. Moreover, apart from Fig. 11, which utilizes a Rayleigh channel model, all other simulations are conducted using the 3GPP channel model.

Fig. 10 illustrates the performance curves of CE and DD under varying SNRs for different schemes. In the simulations, each user is allocated a pilot length of $L_p = 15$ and a data length of $L_d = 50$. As shown in Fig. 10(a), the iterative CE and DD schemes (i.e., **Iter Langevin CE & DD**, **Iter LMMSE-CE & OAMP-DD**, and **Our Iter SDE CE & DD**) achieve an NMSE gain of approximately 7 and 10 dB when compared to the non-iterative **Pilot LS-CE & ZF-DD**, and **Pilot LS-CE & OAMP-DD** schemes. This indicates a significant gain in CE performance by using detected data to refine CE. As SNR increases, **Iter LMMSE-CE & OAMP-DD**'s performance approaches that of **LS-CE & Perfect-Data**. Similarly, the performance of **Our Iter SDE CE & DD** (Algorithm 2) matches **SDE-CE & Perfect-Data** (Algorithm 1) for SNR ≥ 8 dB. When the data is not perfectly known, **Our Iter SDE CE & DD** outperforms **Iter LMMSE-CE & OAMP-DD** because it more effectively utilizes the prior distribution of the channel

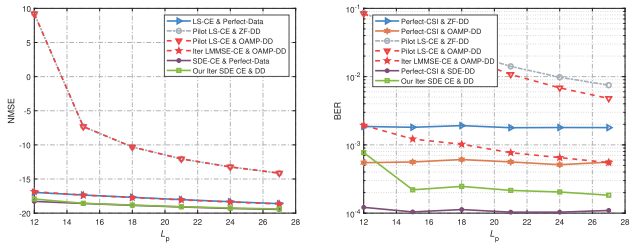
Authorized licensed use limited to: UNIVERSITY OF SOUTHAMPTON. Downloaded on January 09,2026 at 17:30:07 UTC from IEEE Xplore. Restrictions apply.

through a learned prior score function, whereas **Iter LMMSE-CE & OAMP-DD** relies only on the channel's second-order statistical information via LMMSE. Additionally, **Our Iter SDE CE & DD** implements a PC sampler to enhance SDE solution performance beyond what **Iter Langevin CE & DD** algorithm can achieve with only a corrector.

Fig. 10(b) compares the BER performance of various algorithms. Similar to Fig. 10(a), the non-iterative **Pilot LS-CE & ZF-DD** and **Pilot LS-CE & OAMP-DD** algorithms perform worse than the iterative algorithms. **Our Iter SDE CE & DD** scheme achieves performance close to that of **Perfect-CSI & SDE-DD** and surpasses the **Iter LMMSE-CE & OAMP-DD** and **Iter Langevin CE & DD** algorithms. At a BER of 10^{-4}, **Our Iter SDE CE & DD** offers a 3 dB SNR gain over **Iter LMMSE-CE & OAMP-DD**. Furthermore, **Perfect-CSI & SDE-DD** outperforms **Perfect-CSI & OAMP-DD** and **Perfect-CSI & ZF-DD** algorithms. This is likely because the channel matrix elements for the 3GPP scenario do not meet the i.i.d. distribution assumptions for DD tasks, causing OAMP algorithms to underperform. On the other hand, the ZF algorithm does not exploit the prior distribution information of the constellation. In contrast, **Perfect-CSI & SDE-DD** thoroughly leverages the prior distribution of the data and the likelihood information of the received signals, resulting in superior performance.

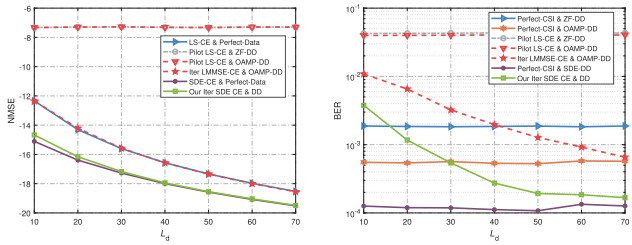
In Fig. 11, we conducted simulations using the MIMO Rayleigh channel model, where the channel prior is assumed to follow $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$. Instead of utilizing a neural network-learned score, we substituted the derived prior score (17) into the process of Algorithm 2. The results indicate that **Our Iter SDE CE & DD** scheme outperforms traditional two-stage algorithms such as **Pilot LS-CE & ZF-DD** and **Pilot LS-CE & OAMP-DD**. However, it is less effective compared to the **Iter LMMSE-CE & OAMP-DD** algorithms. This is because, in a simple Rayleigh channel distribution, the mean and covariance matrix—representing first and second-order statistical information—are adequate to capture the channel's distribution characteristics. Consequently, the LMMSE-CE and OAMP-DD achieve near-optimal performance, while the prior score offers limited additional information. Additionally, numerical errors in solving the SDE process can affect the scheme's final performance. This suggests that the proposed scheme excels with more complex channel distributions, whereas existing methods already perform well in simple Rayleigh channels.

Fig. 12 compares the performance curves of CE and DD as the functions of pilot length L_p for different schemes. In the simulation, each user's data length is $L_d = 50$, and the received SNR at the BS is set to 10 dB. As shown in Fig. 12(a), when $L_p > 12$, **Our Iter SDE CE & DD** can achieve the performance of **SDE-CE & Perfect-Data**, whereas the LS algorithm, which relies solely on pilots, exhibits a notable performance gap relative to iterative CE and DD algorithms. This suggests that additional computation can potentially reduce communication pilot overhead. Also **Iter LMMSE-CE & OAMP-DD** and **LS-CE & Perfect-Data** exhibit similar performance, indicating that the estimated data is sufficiently robust to enhance CE. Fig. 12(b) shows that the



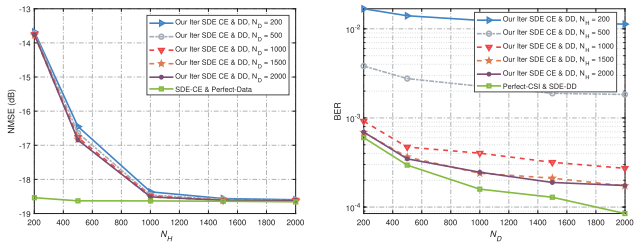
(a) NMSE (b) BER

Fig. 12. NMSE and BER performance with respect to the pilot symbol length L_p for different CE and DD algorithms.



(a) NMSE (b) BER

Fig. 13. NMSE and BER performance with respect to the data symbol length L_d for different CE and DD algorithms.



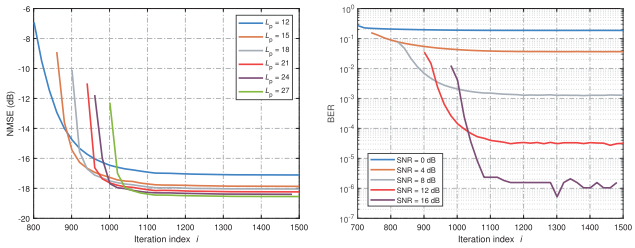
(a) NMSE (b) BER

Fig. 14. NMSE and BER performance with respect to the step numbers N_H and N_X of our Iter SDE CE & DD algorithm.

BER performance of **Our Iter SDE CE & DD** is considerably better than that of **Perfect-CSI & OAMP-DD**, which relies on the perfect CSI. As expected, **Perfect-CSI & SDE-DD** attains the best performance.

Fig. 13 depicts the performance curves of CE and DD across various schemes as the functions of data length L_d. In the simulation, the pilot length for each UE is fixed at $L_p = 15$, with the received SNR = 10 dB at the BS. As can be seen from Fig. 13(a), the iterative algorithm improves CE performance as data length increases. For $L_d \geq 30$, the NMSE of **Our Iter SDE CE & DD** matches that of **SDE-CE & Perfect-Data**, offering an NMSE gain of 1 to 2 dB over **Iter LMMSE-CE & OAMP-DD**. As depicted in Fig. 13(b), increasing L_d improves the accuracy of CE, thereby enhancing the BER performance of **Iter LMMSE-CE & OAMP-DD** and **Our Iter SDE CE & DD**. Moreover, when $L_d = 70$, the BER of **Iter LMMSE-CE & OAMP-DD** approximates that of **Perfect-CSI & OAMP-DD**, while **Our Iter SDE CE & DD** surpasses **Perfect-CSI & OAMP-DD**, in terms of BER.

Authorized licensed use limited to: UNIVERSITY OF SOUTHAMPTON. Downloaded on January 09,2026 at 17:30:07 UTC from IEEE Xplore. Restrictions apply.



(a) (b)

Fig. 15. Convergence performance of (a) NMSE with varying pilot lengths L_p and (b) BER under different SNRs, both as a function of the iteration index i.

Fig. 14 compares the performance of CE and DD of **Our Iter SDE CE & DD** scheme as the function of the maximum algorithm iteration numbers N_H and N_X. In the simulation, each UE's pilot length is set to $L_p = 15$ and the data length is $L_d = 50$, with the received SNR fixed at 10 dB. For the predefined noise diffusion functions $\sigma(t)$ and $\tau(t)$, the interval $t \in [0, 1]$ can be divided into segments with different values of N_X and N_H to control the noise variance gap between adjacent steps, thereby balancing generation speed and performance. As shown in Fig. 14(a), as the channel sampling steps N_H increase, the NMSE performance improves gradually due to a more elaborate denoising process, resulting in more accurate SDE solutions. When $N_H = 1000$, the NMSE of **Our Iter SDE CE & DD** approximates that of **SDE-CE & Perfect-Data**. Moreover, under the same N_H, CE performance remains similar for different N_D, possibly due to SDE-based CE not being sensitive to DD errors. Additionally, Fig. 14(b) shows that as the data sampling steps N_D increase, the BER performance of DD gradually improves. However, the BER curves vary for different N_H, indicating that for DD problems, SDE-based DD methods are more sensitive to CE errors.

Fig. 15(a) illustrates the NMSE performance curve of **Our Iter SDE CE & DD** relative to the iteration index i with varying pilot lengths L_p. The maximum number of iterations is set to $N_H = 1500$ ($i \leq 1500$). The simulation results indicate that as the pilot length L_p increases, the convergence iteration count for the proposed algorithm decreases from approximately 400 iterations at $L_p = 12$ to about 100 iterations at $L_p = 27$. Additionally, the final converged NMSE improves from -17.1 dB to -18.5 dB. Likewise, Fig. 15(b) demonstrates that with different SNRs, both the convergence speed and the final DD outcome of our algorithm improve as the SNR increases. Despite employing a maximum of 1500 iterations, the simulation results reveal that under favorable conditions, such as $L_p = 27$ and SNR $= 16$ dB, the proposed scheme requires only about 50 to 100 iterations to achieve satisfactory results.

In Tab. II, we present the average computation latency for one single realization of **Our Iter SDE CE & DD** algorithm, both with and without LMMSE initialization, across various pilot lengths. Simulations were performed using an NVIDIA GeForce GTX 4090 GPU and an AMD EPYC 7J13 CPU. As the pilot length L_p increases, the computational latency of the method with LMMSE initialization gradually

TABLE II

COMPUTATION LATENCY OF ALGORITHM 2 AT DIFFERENT PILOT LENGTHS

Metrics	Methods	$L_p = 12$	$L_p = 15$	$L_p = 18$	$L_p = 21$	$L_p = 24$	$L_p = 27$
Latency [s]	w/ LMMSE init	2.32	2.14	1.99	1.87	1.76	1.70
	w/o LMMSE init	4.89	4.88	4.88	4.87	4.86	4.88
	relative time cost	47.41%	43.86%	40.84%	38.42%	36.14%	34.79%

TABLE III

COMPUTATION LATENCY OF ALGORITHM 2 AT DIFFERENT SNRS

Metrics	Methods	SNR = 0 dB	SNR = 4 dB	SNR = 8 dB	SNR = 12 dB	SNR = 16 dB
Latency [s]	w/ LMMSE init	2.62	2.39	2.16	1.84	1.44
	w/o LMMSE init	4.87	4.89	4.88	4.89	4.88
	relative time cost	53.76%	48.82%	44.37%	37.61%	29.52%

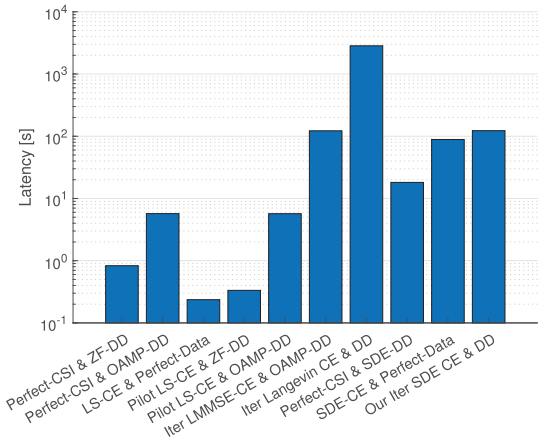
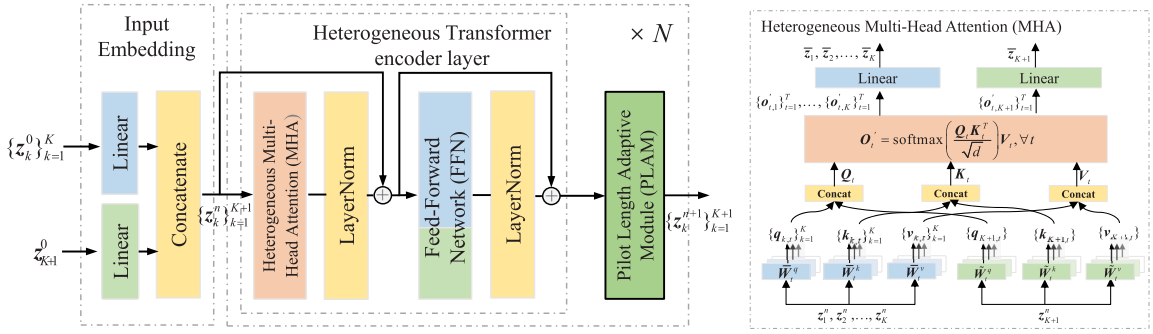


Fig. 16. Computation latency of different CE and DD schemes.

decreases, whereas the latency without LMMSE initialization remains relatively constant. This occurs because a longer pilot results in more accurate initial channel estimation, which corresponds to a smaller diffusion noise-level in the iteration process, resulting in fewer required iteration steps. Similarly, Tab. III illustrates the differences in computational latency across various SNRs when LMMSE initialization is applied or not. Notably, our approach requires only 29%-53% of the original scheme's complexity, underscoring the effectiveness of the LMMSE initialization strategy.

Fig. 16 compares the computational latency of various methods for processing a batch size of 64 in CE and DD tasks. Due to its asynchronous update mechanism and LMMSE initialization, **Our Iter SDE CE & DD** algorithm demonstrates significantly lower computational latency compared to **Iter Langevin CE & DD** algorithm. Furthermore, the latency of **Our Iter SDE CE & DD** is comparable to that of **Iter LMMSE-CE & OAMP-DD**, while the previous simulations have shown it outperforming **Iter LMMSE-CE & OAMP-DD** in both CE and DD. Although the latency of iterative CE and DD algorithms is higher than that of two-stage algorithms, such as **Pilot LS-CE & ZF-DD** and **Pilot LS-CE & OAMP-DD**, existing researches indicate that diffusion-based algorithms can reduce the number of steps from thousands to tens, or even to as few as 1 to 2 steps, through acceleration methods such as higher-order algorithms

Authorized licensed use limited to: UNIVERSITY OF SOUTHAMPTON. Downloaded on January 09,2026 at 17:30:07 UTC from IEEE Xplore. Restrictions apply.



(a) (b)

Fig. 17. Block diagrams of (a) the heterogeneous transformer network and the PLAM in the intermediate layer of the proposed VPL-AUDNet, and (b) the heterogeneous MHA within the heterogeneous transformer encoder layer.

[48] and consistency models [50]. Therefore, accelerating diffusion algorithms still holds significant potential for addressing CE and DD problems.

VII. Conclusion

We have proposed techniques for addressing the challenges of AUD and JCEDD in massive random access, where standard deep learning often struggles to generalize across various problem dimensions using a single model. Our VPL-AUDNet architecture has been designed for variable pilot length transmission, enabling a single model to adapt to different pilot lengths and antennas, thereby achieving superior AUD performance in both 3GPP and Rayleigh channels. Furthermore, we have introduced a generative diffusion models-driven JCEDD scheme using PC samplers, which significantly enhances CE and DD performance through an asynchronous iterative algorithm. These proposed schemes have demonstrated the potential of utilizing deep learning and generative models to address signal detection and estimation challenges in massive random access. A future research direction of interest is to consider accelerating generative model methods to further reduce inference latency.

Appendix A
Structure of the Heterogeneous Transformer Network

Fig. 17(a) illustrates the structure of the intermediate layer in the proposed VPL-AUDNet network. First, for the input pilot feature sequences $\{\boldsymbol{z}_k^0\}_{k=0}^K$, and the received signal feature sequence $\boldsymbol{z}_{K+1}^0$, two distinct linear layers are employed for linear embedding. This process results in the sequences, $\{\boldsymbol{z}_k^1\}_{k=1}^K$, with dimensions $(B, K+1, d)$, where B is the batch size, $K+1$ is the sequence length, and d is the feature dimension. Subsequently, the sequence undergoes processing through N alternating heterogeneous Transformer encoder layers and PLAM networks, and finally produce $\{\boldsymbol{z}_k^{N+1}\}_{k=1}^K$ for the output layer of the VPL-AUDNet. Fig. 17(a) also demonstrates that the unique aspect of the heterogeneous Transformer network, in contrast to the traditional Transformer, is its use of distinct linear matrix weights for input embedding, MHA, and FFN components.

Fig. 17(b) details the computation process of heterogeneous MHA. In the traditional Transformer, the query matrices $\boldsymbol{Q}_t$ for the t-th head are calculated using the same weight matrices $\boldsymbol{W}_t^q$ across all $K+1$ input sequences, that is, $\boldsymbol{Q}_t = \boldsymbol{W}_t^q \boldsymbol{z}_k$, where $k \in \{1, \ldots, K+1\}$. In contrast, we employ two distinct sets of matrices, $\bar{\boldsymbol{W}}_t^q$ and $\tilde{\boldsymbol{W}}_t^q$, to separately derive the query results for the pilot sequence and the received signal sequence, as illustrated at the bottom of Fig. 17(b): $\boldsymbol{q}_{k,t} = \bar{\boldsymbol{W}}_t^q \boldsymbol{z}_k$, for $k \in \{1, \ldots, K\}$, and $\boldsymbol{q}_{K+1,t} = \tilde{\boldsymbol{W}}_t^q \boldsymbol{z}_{K+1}$. Subsequently, we concatenate the sequences $\{\boldsymbol{q}_{k,t}\}_{k=1}^K$ and $\boldsymbol{q}_{K+1,t}$ into $\boldsymbol{Q}_t = \left[\boldsymbol{q}_{1,t}, \ldots, \boldsymbol{q}_{K,t}, \boldsymbol{q}_{K+1,t}\right]$ for further calculations. A similar procedure is applied to compute the key matrices $\boldsymbol{K}_t$ and value matrices $\boldsymbol{V}_t$. Finally, after matrix multiplication and the softmax operation, the output across different heads is divided into the first K sequences and the $(K+1)$-th sequence. Different linear layers are utilized for multi-head fusion, resulting in $\{\tilde{\boldsymbol{z}}_k\}_{k=1}^{K+1}$, which are then forwarded to the subsequent module.

Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not reflect the views of National Research Foundation, Singapore.

References

[1] Z. Gao et al., "Compressive-sensing-based grant-free massive access for 6G massive communication," *IEEE Internet Things J.*, vol. 11, no. 5, pp. 7411–7435, Mar. 2024.
[2] *The ITU-R Framework for IMT-2030*, ITU-R WP 5D, M.2516, Jul. 2023.
[3] J. Che, Z. Zhang, Z. Yang, X. Chen, and C. Zhong, "Massive unsourced random access for NGMA: Architectures, opportunities, and challenges," *IEEE Netw.*, vol. 37, no. 1, pp. 28–35, Jan. 2023.
[4] L. Liu and W. Yu, "Massive connectivity with massive MIMO—Part I: Device activity detection and channel estimation," *IEEE Trans. Signal Process.*, vol. 66, no. 11, pp. 2933–2946, Jun. 2018.
[5] M. Gkagkos, K. R. Narayanan, J.-F. Chamberland, and C. N. Georghiades, "FASURA: A scheme for quasi-static fading unsourced random access channels," *IEEE Trans. Commun.*, vol. 71, no. 11, pp. 6391–6401, Nov. 2023.
[6] J. Gao, Y. Wu, S. Shao, W. Yang, and H. V. Poor, "Energy efficiency of massive random access in MIMO quasi-static Rayleigh fading channels with finite blocklength," *IEEE Trans. Inf. Theory*, vol. 69, no. 3, pp. 1618–1657, Mar. 2023.
[7] X. Shao, X. Chen, D. W. K. Ng, C. Zhong, and Z. Zhang, "Cooperative activity detection: Sourced and unsourced massive random access paradigms," *IEEE Trans. Signal Process.*, vol. 68, pp. 6578–6593, 2020.
[8] Z. Chen, F. Sohrabi, and W. Yu, "Sparse activity detection in multi-cell massive MIMO exploiting channel large-scale fading," *IEEE Trans. Signal Process.*, vol. 69, pp. 3768–3781, 2021.

Authorized licensed use limited to: UNIVERSITY OF SOUTHAMPTON. Downloaded on January 09,2026 at 17:30:07 UTC from IEEE Xplore. Restrictions apply.

[9] W. Jiang, Y. Jia, and Y. Cui, "Statistical device activity detection for OFDM-based massive grant-free access," *IEEE Trans. Wireless Commun.*, vol. 22, no. 6, pp. 3805–3820, Jun. 2023.

[10] Y. Li, Z. Chen, Y. Wang, C. Yang, B. Ai, and Y.-C. Wu, "Heterogeneous transformer: A scale adaptable neural network architecture for device activity detection," *IEEE Trans. Wireless Commun.*, vol. 22, no. 5, pp. 3432–3446, May 2023.

[11] Z. Sheng, P. Zhu, and X. You, "UADFormer: A transformer-based deep learning method for user activity detection in cell-free massive MIMO systems," *IEEE Trans. Wireless Commun.*, vol. 23, no. 12, pp. 18516–18531, Dec. 2024.

[12] L. Qiao, J. Zhang, Z. Gao, S. Chen, and L. Hanzo, "Compressive sensing based massive access for IoT relying on media modulation aided machine type communications," *IEEE Trans. Veh. Technol.*, vol. 69, no. 9, pp. 10391–10396, Sep. 2020.

[13] M. Ke, Z. Gao, Y. Wu, X. Gao, and R. Schober, "Compressive sensing-based adaptive active user detection and channel estimation: Massive access meets massive MIMO," *IEEE Trans. Signal Process.*, vol. 68, pp. 764–779, 2020.

[14] G. Sun et al., "Massive grant-free OFDMA with timing and frequency offsets," *IEEE Trans. Wireless Commun.*, vol. 21, no. 5, pp. 3365–3380, May 2022.

[15] K. Ying et al., "Quasi-synchronous random access for massive MIMO-based LEO satellite constellations," *IEEE J. Sel. Areas Commun.*, vol. 41, no. 6, pp. 1702–1722, Jun. 2023.

[16] Y. Cui, S. Li, and W. Zhang, "Jointly sparse signal recovery and support recovery via deep learning with applications in MIMO-based grant-free random access," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 3, pp. 788–803, Mar. 2021.

[17] X. Shao, X. Chen, Y. Qiang, C. Zhong, and Z. Zhang, "Feature-aided adaptive-tuning deep learning for massive device detection," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 7, pp. 1899–1914, Jul. 2021.

[18] J. Ma and L. Ping, "Orthogonal AMP," *IEEE Access*, vol. 5, pp. 2020–2033, 2017.

[19] J. Céspedes, P. M. Olmos, M. Sánchez-Fernández, and F. Perez-Cruz, "Expectation propagation detection for high-order high-dimensional MIMO systems," *IEEE Trans. Commun.*, vol. 62, no. 8, pp. 2840–2849, Aug. 2014.

[20] K. Pratik, B. D. Rao, and M. Welling, "RE-MIMO: Recurrent and permutation equivariant neural MIMO detection," *IEEE Trans. Signal Process.*, vol. 69, pp. 459–473, 2021.

[21] H. He, C.-K. Wen, S. Jin, and G. Y. Li, "Model-driven deep learning for MIMO detection," *IEEE Trans. Signal Process.*, vol. 68, pp. 1702–1715, 2020.

[22] A. Kosasih et al., "Graph neural network aided expectation propagation detector for MU-MIMO systems," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Apr. 2022, pp. 1212–1217.

[23] H. He, X. Yu, J. Zhang, S. Song, and K. B. Letaief, "Message passing meets graph neural networks: A new paradigm for massive MIMO systems," *IEEE Trans. Wireless Commun.*, vol. 23, no. 5, pp. 4709–4723, May 2024.

[24] S. Zhang, Y. Cui, and W. Chen, "Joint device activity detection, channel estimation and signal detection for massive grant-free access via BiGAMP," *IEEE Trans. Signal Process.*, vol. 71, pp. 1200–1215, 2023.

[25] M. Ke, Z. Gao, M. Zhou, D. Zheng, D. W. K. Ng, and H. V. Poor, "Next-generation URLLC with massive devices: A unified semi-blind detection framework for sourced and unsourced random access," *IEEE J. Sel. Areas Commun.*, vol. 41, no. 7, pp. 2223–2244, Jul. 2023.

[26] G. Sun, M. Cao, W. Wang, W. Xu, and C. Studer, "Deep-unfolded massive grant-free transmission in cell-free wireless communication systems," *IEEE Trans. Signal Process.*, vol. 73, pp. 1094–1109, 2025.

[27] M. Arvinte and J. I. Tamir, "MIMO channel estimation using score-based generative models," *IEEE Trans. Wireless Commun.*, vol. 22, no. 6, pp. 3698–3713, Jun. 2023.

[28] Y. Song and S. Ermon, "Generative modeling by estimating gradients of the data distribution," in *Proc. Adv. Neural Inf. Process. Syst.*, Vancouver, BC, Canada, Dec. 2019, pp. 11918–11930.

[29] Y. Song and S. Ermon, "Improved techniques for training score-based generative models," in *Proc. NIPS*, vol. 33, Vancouver, BC, Canada, 2020, pp. 12438–12448.

[30] L. He, Z. Wang, and Y. Huang, "A massive MIMO sampling detection strategy based on denoising diffusion model," in *Proc. Int. Wireless Commun. Mobile Comput. (IWCMC)*, Ayia Napa, Cyprus, May 2024, pp. 1443–1448.

[31] N. Zilberstein, C. Dick, R. Doost-Mohammady, A. Sabharwal, and S. Segarra, "Annealed Langevin dynamics for massive MIMO detection," *IEEE Trans. Wireless Commun.*, vol. 22, no. 6, pp. 3762–3776, Jun. 2023.

[32] N. Zilberstein, A. Sabharwal, and S. Segarra, "Solving linear inverse problems using higher-order annealed Langevin diffusion," *IEEE Trans. Signal Process.*, vol. 72, pp. 492–505, 2024.

[33] N. Zilberstein, A. Swami, and S. Segarra, "Joint channel estimation and data detection in massive MIMO systems based on diffusion models," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Seoul, South Korea, Apr. 2024, pp. 13291–13295.

[34] X. Zhou, L. Liang, J. Zhang, P. Jiang, Y. Li, and S. Jin, "Generative diffusion models for high dimensional channel estimation," 2024, *arXiv:2408.10501*.

[35] J. Ho, A. N. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, Dec. 2024, pp. 6840–6851.

[36] B. Fesl, M. Baur, F. Strasser, M. Joham, and W. Utschick, "Diffusion-based generative prior for low-complexity MIMO channel estimation," *IEEE Wireless Commun. Lett.*, vol. 13, no. 12, pp. 3493–3497, Dec. 2024.

[37] T. Wu et al., "CDDM: Channel denoising diffusion models for wireless semantic communications," *IEEE Trans. Wireless Commun.*, vol. 23, no. 9, pp. 11168–11183, Sep. 2024.

[38] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, "Score-based generative modeling through stochastic differential equations," in *Proc. 9th Int. Conf. Learn. Represent.*, May 2021, pp. 1–36.

[39] 3GPP, *Study on Channel Model for Frequencies From 0.5 to 100 GHz*, 3rd Generation Partnership Project (3GPP), document (TR) 38.901, May 2017, version 14.0.0.

[40] J. Xu, B. Ai, W. Chen, A. Yang, P. Sun, and M. Rodrigues, "Wireless image transmission using deep source channel coding with attention modules," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 4, pp. 2315–2328, Apr. 2022.

[41] S. H. Chan, "Tutorial on diffusion models for imaging and vision," 2024, *arXiv:2403.18103*.

[42] B. D. O. Anderson, "Reverse-time diffusion equation models," *Stochastic Processes Appl.*, vol. 12, no. 3, pp. 313–326, May 1982.

[43] B. Kawar, M. Elad, S. Ermon, and J. Song, "Denoising diffusion restoration models," in *Proc. Adv. Neural Inf. Process. Syst.*, Nov. 2022, pp. 23593–23606.

[44] P. Vincent, "A connection between score matching and denoising autoencoders," *Neural Comput.*, vol. 23, no. 7, pp. 1661–1674, Jul. 2011.

[45] B. Efron, "Tweedie's formula and selection bias," *J. Amer. Stat. Assoc.*, vol. 106, no. 496, pp. 1602–1614, Dec. 2011.

[46] X. Meng and Y. Kabashima, "Quantized compressed sensing with score-based generative models," in *Proc. Int. Conf. Learn. Represent.*, Kigali, Rwanda, 2022, pp. 1–30.

[47] J. Song, C. Meng, and S. Ermon, "Denoising diffusion implicit models," in *Proc. Int. Conf. Learn. Represent.*, 2020, pp. 1–20.

[48] K. Zheng, C. Lü, J. Chen, and J. Zhu, "DPM-solver-v3: Improved diffusion ODE solver with empirical model statistics," in *Proc. Adv. Neural Inf. Process. Syst.*, Dec. 2023, pp. 1–41.

[49] S. Jaeckel et al., "QuaDRiGa-quasi deterministic radio channel generator, user manual and documentation," Heinrich Hertz Inst., Berlin, Germany, Tech. Rep. v2.6.1, 2021.

[50] Y. Song, P. Dhariwal, M. Chen, and I. Sutskever, "Consistency models," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2023, pp. 32211–32252.

Keke Ying (Graduate Student Member, IEEE) received the B.S. degree from Beijing Institute of Technology, China, in 2020, where he is currently pursuing the Ph.D. degree with the School of Information and Electronics. His research interests include reconfigurable massive MIMO, massive machine type communication, and deep learning-based signal processing.

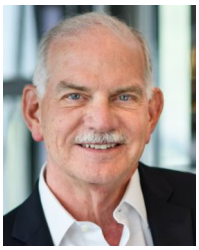
Authorized licensed use limited to: UNIVERSITY OF SOUTHAMPTON. Downloaded on January 09,2026 at 17:30:07 UTC from IEEE Xplore. Restrictions apply.

Zhen Gao (Senior Member, IEEE) received the B.S. degree in information engineering from Beijing Institute of Technology, Beijing, China, in 2011, and the Ph.D. degree in communication and signal processing from Tsinghua National Laboratory for Information Science and Technology, Department of Electronic Engineering, Tsinghua University, China, in 2016.

He is currently a Professor with Beijing Institute of Technology. His research interests include wireless communications, with a focus on multi-carrier modulations, multiple antenna systems, and sparse signal processing. He was a recipient of the IEEE Broadcast Technology Society 2016 Scott Helt Memorial Award (Best Paper), the Exemplary Reviewer of IEEE COMMUNICATION LETTERS in 2016, IET Electronics Letters Premium Award (Best Paper) 2016, and the Young Elite Scientists Sponsorship Program (2018–2021) from China Association for Science and Technology.

Sheng Chen (Life Fellow, IEEE) received the B.Eng. degree from East China Petroleum Institute, Dongying, China, in January 1982, the Ph.D. degree from the City St George's, University of London, in September 1986, both in control engineering, and the Doctor of Sciences (D.Sc.) degree from the University of Southampton, Southampton, U.K., in August 2005. From 1986 to 1999, he held research and academic appointments at The University of Sheffield, The University of Edinburgh, and The University of Portsmouth; U.K. Since 1999, he has been with the School of Electronics and Computer Science, University of Southampton, where he holds the post of a Professor of intelligent systems and signal processing. In 2023, he was appointed as a Distinguish Professor with the Faculty of Information Science and Engineering, Ocean University of China, Qingdao, China. He has published over 700 research articles. He has more than 22,000 Web of Science citations with an H-index of 64 and more than 42,000 Google Scholar citations with an H-index of 87. His research interests include adaptive signal processing, wireless communications, modeling and identification of nonlinear systems, neural networks, and machine learning. He is a fellow of United Kingdom Royal Academy of Engineering, Asia–Pacific Artificial Intelligence Association, and IET. He is one of the original 200 ISI highly cited researchers in engineering (March 2004).

Tony Q. S. Quek (Fellow, IEEE) received the B.E. and M.E. degrees in electrical and electronics engineering from the Tokyo Institute of Technology in 1998 and 2000, respectively, and the Ph.D. degree in electrical engineering and computer science from the Massachusetts Institute of Technology in 2008. Currently, he is the Associate Provost (AI & Digital Innovation) and Cheng Tsang Man Chair Professor with Singapore University of Technology and Design (SUTD). He also serves as the Director of the Future Communications Research and Development Programme, and the ST Engineering Distinguished Professor. He is a co-founder of Silence Laboratories and NeuroRAN. His current research topics include wireless communications and networking, network intelligence, non-terrestrial networks, open radio access network, AI-RAN, and 6G. He was honored with the 2008 Philip Yeo Prize for Outstanding Achievement in Research, the 2012 IEEE William R. Bennett Prize, the 2015 SUTD Outstanding Education Awards – Excellence in Research, the 2016 IEEE Signal Processing Society Young Author Best Paper Award, the 2017 CTTC Early Achievement Award, the 2017 IEEE ComSoc AP Outstanding Paper Award, the 2020 IEEE Communications Society Young Author Best Paper Award, the 2020 IEEE Stephen O. Rice Prize, the 2020 Nokia Visiting Professor, the 2022 IEEE Signal Processing Society Best Paper Award, the 2024 IIT Bombay International Award For Excellence in Research in Engineering and Technology, the IEEE Communications Society WTC Recognition Award 2024, and the Public Administration Medal (Bronze). He is a WWRF Fellow, an AIAA Fellow, and a Fellow of the Academy of Engineering Singapore.



H. Vincent Poor (Life Fellow, IEEE) received the Ph.D. degree in EECS from Princeton University in 1977. From 1977 to 1990, he was on the faculty of the University of Illinois at Urbana–Champaign. Since 1990, he has been on the faculty at Princeton University, where he is currently the Michael Henry Strater University Professor. From 2006 to 2016, he was the Dean of the School of Engineering and Applied Science, Princeton University, and he has also held visiting appointments at several other universities, including, most recently at Berkeley and Caltech. His research interests include information theory, stochastic analysis and machine learning, and their applications in wireless networks, energy systems, and related fields. Among his publications in these areas is the book *Machine Learning and Wireless Communications* (Cambridge University Press, 2022). He is a member of the National Academy of Engineering and the National Academy of Sciences and is a Foreign Member of the Royal Society and other national and international academies. He received the IEEE Alexander Graham Bell Medal in 2017.

Authorized licensed use limited to: UNIVERSITY OF SOUTHAMPTON. Downloaded on January 09,2026 at 17:30:07 UTC from IEEE Xplore. Restrictions apply.