Distributed Learning for Uplink Cell-Free Massive MIMO Networks

Rui Wang, *Student Member, IEEE*, Weijie Dai, and Yi Jiang, *Member, IEEE*

***Abstract*— Cell-free massive multiple-input multiple-output (MIMO) can resolve the inter-cell interference issue in cellular networks through cooperative beamforming of the distributed access points (APs). This paper focuses on an uplink cell-free massive MIMO network and investigates novel methods to train the central processing unit (CPU), the APs, and the users in the network. To reduce the communication burden posed on the fronthaul, each AP applies receive beamforming to compress the vector signals into scalar ones before passing them to the CPU for centralized processing. By drawing analogies between an uplink cell-free network and a quasi-neural network and borrowing the idea of backpropagation algorithm, we propose a novel scheme named the distributed learning for uplink cell-free massive MIMO beamforming (DLCB), which can achieve the multi-AP cooperation without explicit estimation of their channel state information (CSI). The DLCB has low computational complexity and is applicable to various objective functions, such as the minimum mean squared error criterion and the maximum sum rate criterion. Extensive simulations verify that the proposed scheme achieves superior performance over the state-of-the-art methods.**

***Index Terms*— Cell-free massive MIMO network, backpropagation algorithm, quasi-neural network, distributed learning.**

I. Introduction

CELL-FREE massive multiple-input multiple-output (MIMO) is one of the promising technologies for future generation of wireless networks [1], [2]. As shown in Fig. 1, the geographically distributed access points (APs) being connected to the central processing units (CPU) can cooperate to form a virtual massive MIMO system [3], which can achieve both the high spectral efficiency of massive MIMO and the macro-diversity gain of joint and coherent processing from distributed APs [1]. Hence it can provide much higher quality-of-service for the user equipments (UEs) than the conventional small-cell technology [4].

To fully reap the theoretical benefits of cell-free massive MIMO, considerable research efforts have been devoted to numerous aspects of cell-free massive MIMO in recent years, including the network scalability [5], [6], [7], the user-centric deployment [8], [9], [10], the cooperative precoding and combining [11], [12], [13], [14], [15], [16], [17], [18], the power allocation [14], [15], [19], [20], the pilot assignment [21], and the analysis of hardware impairments [22], [23].

Manuscript received 7 July 2022; revised 17 December 2022, 18 April 2023, and 18 June 2023; accepted 18 June 2023. Date of publication 22 June 2023; date of current version 18 September 2023. This work was supported in part by National Natural Science Foundation of China Grant No. 61771005. The associate editor coordinating the review of this article and approving it for publication was L. Sanguinetti. *(Corresponding author: Yi Jiang.)*

The authors are with the Key Laboratory for Information Science of Electromagnetic Waves (MoE), Department of Communication Science and Engineering, School of Information Science and Technologies, Fudan University, Shanghai 200438, China (e-mail: ruiwang18@fudan.edu.cn; weijiedai22@m.fudan.edu.cn; yijiang@fudan.edu.cn).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCOMM.2023.3288578.

Digital Object Identifier 10.1109/TCOMM.2023.3288578

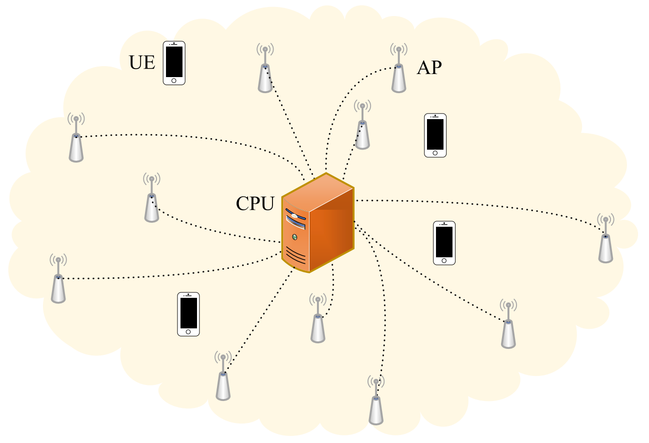

Fig. 1. Illustration of a cell-free massive MIMO network with geographically distributed APs connected to the CPU via the fronthaul links.

This paper investigates the cooperative transmission and reception in the uplink scenario, where the cost of the APs' cooperation is the main obstacle to real implementation. The optimal performance of an uplink cell-free massive MIMO network can be achieved by fully centralized minimum mean square error (MMSE) method [18] (referred to as "Level 4" method therein). But the centralized method is impractical because it needs channel state information (CSI) between all the APs and all the UEs and uses the inversions of a collective channel matrix with high dimension. To avoid the estimation and sharing of global CSI, simplistic maximum ratio (MR) combining is proposed in [24], [25], and [26]. But its performance is far inferior to the other linear methods that also require no CSI exchanges among the APs, such as local ZF combining [26] and local MMSE combining (referred to as "Level 2" in [18]). These local combiners, albeit advantageous over MR combining, are considerably inferior to the centralized methods [18], [27]. In [18], a large-scale fading decoding method (the so-called "Level 3") is also proposed, which can outperform "Level 2" by using the channel statistics available at the CPU. While the local designs in [18] only consider cell-free massive MIMO networks with single-antenna UEs, the authors of [28] consider the scenario of multi-antenna UEs and propose to coordinate distributed APs through over-the-air (OTA) interactions without explicit channel estimation.

Most existing papers assume that the fronthaul links from the APs to the CPU have ample channel capacity [4], [22]. But the fronthual links are throughput-limited and incur communication delays in practice [24]. Hence it may be unrealistic to assume that the CPU can collect all data and/or CSI from all the APs in real-time [29]. The authors in [24] and [25] consider such constraints and analyze the effect of the fronthaul quantization. But they only consider single-antenna UEs and use MR combining. "Distributed-OTA" algorithm of [28] actually requires no signal transmission over the fronthaul during the process of optimizing the AP's receiver. But outside of the optimization process, it still requires all the APs to forward high-dimensional vector data streams to the CPU [28], which consumes excessive bandwidth of the fronthaul links.

To significantly reduce the required throughput of the fronthaul link, in this paper we let all the APs first apply receiver beamforming to reduce their received *vector* signal to a *scalar* one before passing it to the CPU. Then to optimize the beamforming weights of the APs without sharing their CSI and the received samples, we resort to the concept of "quasi-neural network", which was originally proposed in our previous work on interference-resilient relay communications [30], [31]. That is, by drawing striking analogies between a cell-free massive MIMO network and an artificial neural network (ANN), we can model a cell-free network as a "quasi-neural network". Owing to the UE-AP-CPU layered structure of cell-free massive MIMO networks, the backpropagation (BP) algorithm [32] can be used to optimize the transmit and receive beamforming weights of cell-free massive MIMO networks. The topological feature of the quasi-neural network is that adjacent layers are connected, but the nodes in the same layer are not. Therefore, when the BP algorithm is executed for optimizing cell-free massive MIMO networks, the APs need not to exchange data among themselves, nor to share their CSI. The proposed scheme, termed as Distributed Learning for uplink Cell-free massive MIMO Beamforming (DLCB), can accommodate various optimization objectives, including the MMSE criterion and the maximum sum rate (MSR). It has low computational complexity and can significantly outperform the state-of-the-art methods, including "Distributed-OTA" in [28] and "Level 3" in [18] as verified by the simulations.

The rest of the paper is organized as follows. Section II introduces the system model of uplink cell-free massive MIMO, formulates the optimization problem, and models the network as a quasi-neural network by proposing the novel operations of the UEs and the APs. Section III proposes the DLCB scheme, designs a frame to support it, and analyzes the number of multiplications conducted by the distributed nodes and the number of OTA and fronthaul signaling per round of training. Section IV presents the scheme for the special scenario of cell-free massive MIMO networks with single-antenna UEs in comparison with the algorithms in [18]. Section V presents simulation results to verify the superior performance of the proposed algorithms over the state-of-the-art. Section VI gives the conclusion.

The following notations are used throughout this paper. $(\cdot)^*$, $(\cdot)^T$, and $(\cdot)^H$ stand for conjugate, transpose, and conjugate transpose, respectively. $\mathbb{R}$ is the set of real numbers. $\mathbb{C}^{M\times N}$ is the set of $M \times N$ complex matrices. $\mathbf{A}_{(m,:)}$ is the m-th row of the $M \times N$ matrix $\mathbf{A}$. $\mathcal{R}e\{\cdot\}$ represents the real part. $\sigma \circ \mathbf{V}$ stands for a composite function of $\sigma(\cdot)$ and $\mathbf{V}$. $\mathrm{diag}(\mathbf{a})$ is a diagonal matrix with vector $\mathbf{a}$ being its diagonal. $\mathrm{blkdiag}(\mathbf{u}_1, \cdots, \mathbf{u}_K)$ is a block diagonal matrix with vector $\mathbf{u}_1, \cdots, \mathbf{u}_K$ being its diagonal.

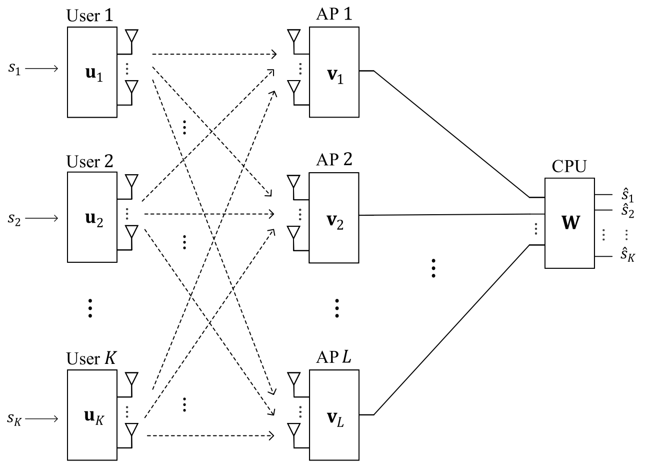

Fig. 2. System structure of an uplink cell-free massive MIMO network with K N-antenna UEs, L M-antenna APs and one CPU.

II. System Model and Problem Formulation

A. System Model

Cell-free massive MIMO shown in Fig. 1 has a layered structure as illustrated in Fig. 2, where K UEs with N antennas communicate with L APs with M antennas and the APs are connected to a CPU via the fronthaul links.

Every UE transmits a single data stream by applying the beamforming vector $\mathbf{u}_k \in \mathbb{C}^N$ to the data $s_k(i)$, i.e.,

$$\mathbf{x}_k(i) = \mathbf{u}_k s_k(i), \quad k = 1, \cdots, K, \tag{1}$$

where i is the time slot index and is omitted in the sequel for notation simplicity, and $\|\mathbf{u}_k\|^2 \leq 1$ is the power constraint per UE.

Let $\mathbf{H}_{l,k} \in \mathbb{C}^{M\times N}$ denote the uplink channel between the k-th UE and the l-th AP, $\mathbf{H}_l \triangleq [\mathbf{H}_{l,1}, \cdots, \mathbf{H}_{l,K}]$ denote the aggregated uplink channel seen by the l-th AP, and $\mathbf{x} \triangleq [\mathbf{x}_1^T, \mathbf{x}_2^T, \cdots, \mathbf{x}_K^T]^T \in \mathbb{C}^{KN}$ denote the aggregated transmit signal of all UEs. Then the received signal of the l-th AP is

$$\mathbf{y}_l = \mathbf{H}_l \mathbf{x} + \mathbf{z}_l, \quad l = 1, \cdots, L, \tag{2}$$

where $\mathbf{z}_l \sim \mathcal{CN}(0, \sigma^2\mathbf{I})$ is the additive complex-valued Gaussian noise. To reduce the required bandwidth of the AP-to-CPU fronthaul, the APs apply beamforming weights $\mathbf{v}_l \in \mathbb{C}^M$ to compress the vector signals into scalars

$$r_l = \mathbf{v}_l^H \mathbf{y}_l, \quad l = 1, \cdots, L, \tag{3}$$

before sending them to the CPU. The CPU collects all these signals into $\mathbf{r} \triangleq [r_1, r_2, \cdots, r_L]^T \in \mathbb{C}^L$ and applies the combining matrix $\mathbf{W} \in \mathbb{C}^{L\times K}$ $(L \geq K)$ to obtain

$$\hat{\mathbf{s}} = \mathbf{W}^H \mathbf{r} \in \mathbb{C}^K \tag{4}$$

as estimation of the transmitted signals of the K UEs.

TABLE I
DIFFERENCES BETWEEN A QUASI-NEURAL NETWORK AND AN ANN

	Quasi-neural network	ANN
connection weights	can be complex-valued (can be fixed and unknown)	typically real-valued (adjustable and known)
activation function	a function with clear physical meaning	a nonlinear function without clear physical meaning
data transmission	affected by noise/interference	error-free
goal of the training	to obtain a modeling tool	to obtain a classifier or a mapping

This paper focuses on optimizing the weights of the UEs, the APs, and the CPU according to some objective function f subject to the power constraint per UE. That is,

$$\begin{aligned} \underset{\{\mathbf{u}_k\}_{k=1}^K,\{\mathbf{v}_l\}_{l=1}^L,\mathbf{W}}{\text{minimize}} \quad & f(\mathbf{u},\mathbf{v},\mathbf{W};\mathbf{H}) \\ \text{s.t. } \|\mathbf{u}_k\|^2 \le 1, \quad & \text{for all } k=1,\cdots,K, \end{aligned} \tag{5}$$

where f can be an arbitrary cost function of interest, such as the mean squared error (MSE) and the negative of the sum rate.

The main challenge in solving (5) is that the UEs, the APs, and the CPU need to optimize their respective weights in a distributed manner without knowing $\mathbf{H}$. The key is to reformulate (5) into an unconstrained problem by relating a cell-free massive MIMO network to a *quasi-neural network* as explained below.

B. A Quasi-Neural Network Representation

To meet the power constraint that $\|\mathbf{u}_k\|^2 \le 1$, we denote

$$\mathbf{u}_k \triangleq \frac{\mathbf{p}_k}{\|\mathbf{p}_k\|} e^{-\theta_k^2}, \quad k=1,\cdots,K, \tag{6}$$

where $\mathbf{p}_k \in \mathbb{C}^N$ and $\theta_k \in \mathbb{R}$ control the direction and the amplitude of the weight, respectively.

Now that the power constrains in (5) is automatically met, we rewrite (5) as an unconstrained problem

$$\underset{\{\mathbf{p}_k,\theta_k\}_{k=1}^K,\{\mathbf{v}_l\}_{l=1}^L,\mathbf{W}}{\text{minimize}} \quad f(\mathbf{p},\theta,\mathbf{v},\mathbf{W};\mathbf{H}). \tag{7}$$

Based on (6) and Fig. 2, we draw a layered diagram of an uplink cell-free massive MIMO network as shown in Fig. 3, which can be represented by the so-called quasi-neural network [31]. As a concept originally proposed in our previous work [31], the quasi-neural network [31] is topologically similar to an ANN owing to its layered structure: multi-layer nodes, connection weights, and nonlinear functions. But it differs from an ANN in many aspects as summarized in Table I. Most important, the quasi-neural network is a model-based tool that has clear physical meanings, for example, its nonlinear function may be the power constraint and its fixed weights may be the physical channel but the activation function and the connection weights of the classic ANN generally has no clear physical meanings.

Indeed, we can draw the analogies between an uplink cell-free massive MIMO network and a quasi-neural network in the following aspects:

i) As illustrated by the shaded band in Fig. 3, the cell-free network has a layered structure consisting of the data streams, and the outputs of the UEs, the APs, and the CPU.

ii) As illustrated by the rectangles in Fig. 3, the cell-free network has complex-valued weights $\{\frac{\mathbf{p}_k}{\|\mathbf{p}_k\|}\}_{k=1}^K$, $\{\mathbf{H}_l \circ \mathbf{v}_l\}_{l=1}^L$, and $\mathbf{W}$, in which the channel coefficients $\{\mathbf{H}_l\}_{l=1}^L$ are fixed and unknown.

iii) The power constraint regulated by the multiplying term $e^{-\theta_k^2}$ is analogous to the nonlinear function of the quasi-neural network.

iv) The network has inherent randomness due to channel noise.

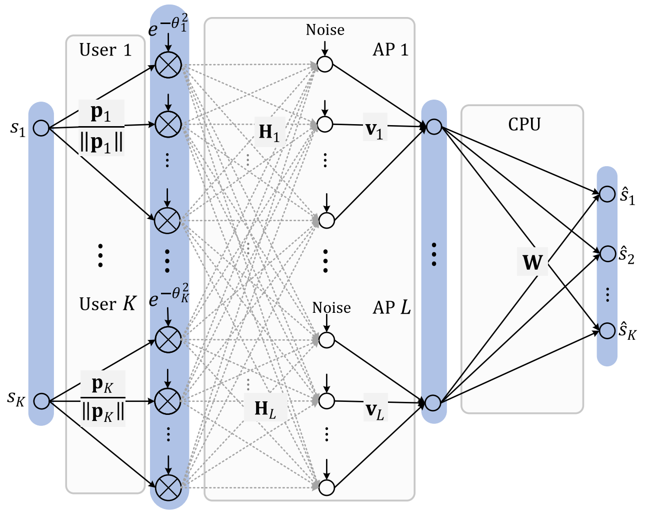

Fig. 3. A quasi-neural network representation of an uplink cell-free massive MIMO network with K UEs, L APs and one CPU.

Inspired by these analogies, we can use the quasi-neural network to represent a cell-free network and thus borrow the idea of the BP algorithm [31] to achieve distributed optimization of a cell-free massive MIMO network, which requires no explicit CSI, consumes reduced fronthaul bandwidth and enjoys low computational complexity, as shown next.

III. THE DISTRIBUTED LEARNING-BASED CELL-FREE BEAMFORMING (DLCB) SCHEME

In this section, we first derive the DLCB algorithm and then present a frame design to support the algorithm.

A. The DLCB Algorithm

Just like a conventional neural network that is trained based on some training "data", cell-free massive MIMO networks can be optimized based on pilot sequences. First, consider optimizing the weight of the CPU by minimizing the MSE, i.e., $\mathbb{E}\left[\|\mathbf{W}^H\mathbf{r}-\mathbf{s}\|^2\right]$. We use the Monte Carlo trials to approximate the expectations as

$$\underset{\mathbf{W}}{\text{minimize}} \ \frac{1}{\tau}\sum_{i=1}^{\tau}\|\mathbf{W}^H\mathbf{r}(i)-\mathbf{s}(i)\|^2, \tag{8}$$

where $\mathbf{s}(i) \in \mathbb{C}^{K\times 1}, i = 1,\ldots,\tau$ is the pilot sequences of the K users and τ is the pilot length. Equating the first derivative of the function in (8) with respect to the weight $\mathbf{W}$ to zero, we can obtain the optimal combiner

$$\mathbf{W}_{\text{opt}} = \left[\sum_{i=1}^{\tau}\mathbf{r}(i)\mathbf{r}(i)^H\right]^{-1}\left[\sum_{i=1}^{\tau}\mathbf{r}(i)\mathbf{s}(i)^H\right], \tag{9}$$

which uses no explicit CSI but the received signals $\mathbf{r}$ and the pilot sequences $\mathbf{s}$. As the length of the pilots increases to infinity, the Monte Carlo approximation equals the value of the expectation and thus $\mathbf{W}_{\text{opt}} \to \mathbf{W}_{\text{MMSE}} = \left(\mathbb{E}[\mathbf{r}\mathbf{r}^H]\right)^{-1}\mathbb{E}[\mathbf{r}\mathbf{s}^H]$.

Given $\mathbf{W}_{\text{opt}}$, the weights of the UEs and the APs can be updated according to

$$\underset{\{\mathbf{p}_k,\theta_k\}_{k=1}^K,\{\mathbf{v}_l\}_{l=1}^L}{\text{minimize}} \quad f(\mathbf{p},\theta,\mathbf{v};\mathbf{H},\mathbf{W}=\mathbf{W}_{\text{opt}}). \tag{10}$$

To this end, we attempt to derive the gradients of f with respect to the weights of the nodes in the network based on a single sample of the pilot $s_k(i)$. Inspired by the BP algorithm, we use the chain rule of derivatives to establish the following proposition.

Proposition 1: Let $\nabla_{\mathbf{v}_l} f = \frac{\partial f}{\partial \mathbf{v}_l^*} \in \mathbb{C}^M$ denote the complex gradient operator, in which the gradient is a vector with the m-th element defined as $[\nabla_{\mathbf{v}_l} f]_m = \nabla_{[\mathbf{v}_l]_m} f = \frac{\partial f}{\partial [\mathbf{v}_l^*]_m}$. The gradient of the objective function f with respect to the weight of the l-th AP is

$$\nabla_{\mathbf{v}_l} f = \mathbf{y}_l\left(\frac{\partial f}{\partial r_l^*}\right)^* \in \mathbb{C}^M, \tag{11}$$

where $\frac{\partial f}{\partial r_l^*}$ is the l-th element of

$$\nabla_{\mathbf{r}} f = \mathbf{W}\cdot\nabla_{\hat{\mathbf{s}}} f \in \mathbb{C}^L. \tag{12}$$

The gradient of f with respect to the weight of the k-th UE is

$$\nabla_{\mathbf{u}_k} f = s_k^* \cdot \nabla_{\mathbf{x}_k} f \in \mathbb{C}^N, \tag{13}$$

where $\nabla_{\mathbf{x}_k} f \in \mathbb{C}^N$ is the k-th block of

$$\nabla_{\mathbf{x}} f \triangleq \begin{bmatrix}\nabla_{\mathbf{x}_1} f\\ \vdots \\ \nabla_{\mathbf{x}_K} f\end{bmatrix} = \sum_{l=1}^{L}\mathbf{H}_l^H\mathbf{v}_l\frac{\partial f}{\partial r_l^*} \in \mathbb{C}^{KN}. \tag{14}$$

Let $\nabla_{\mathbf{a}_k}\mathbf{b}_k = \frac{\partial \mathbf{b}_k}{\partial \mathbf{a}_k^*} \in \mathbb{C}^{M'\times N'}$ denote the complex gradient operator, in which the gradient is a matrix with the $[m,n]$-th element defined as $[\nabla_{\mathbf{a}_k}\mathbf{b}_k]_{mn} = \nabla_{[\mathbf{a}_k]_m}[\mathbf{b}_k]_n = \frac{\partial[\mathbf{b}_k]_n}{\partial[\mathbf{a}_k]_m}$ for $\forall m \in \{1,2,\cdots,M\}, \forall n \in \{1,2,\cdots,N'\}$. According to (6),

$$\nabla_{\mathbf{p}_k} f = \nabla_{\mathbf{p}_k}\mathbf{u}_k^*\cdot\nabla_{\mathbf{u}_k} f + \nabla_{\mathbf{p}_k}\mathbf{u}_k\cdot(\nabla_{\mathbf{u}_k} f)^*, \tag{15}$$

where

$$\nabla_{\mathbf{p}_k}\mathbf{u}_k^* = \frac{e^{-\theta_k^2}}{\|\mathbf{p}_k\|}\left(\mathbf{I}_N - \frac{\mathbf{p}_k\mathbf{p}_k^H}{2\|\mathbf{p}_k\|^2}\right), \tag{16}$$

and

$$\nabla_{\mathbf{p}_k}\mathbf{u}_k = -\frac{e^{-\theta_k^2}}{2\|\mathbf{p}_k\|^3}\mathbf{p}_k\mathbf{p}_k^T; \tag{17}$$

the derivative with respect to the parameter θ_k is

$$\frac{\partial f}{\partial \theta_k} = 2\mathcal{R}e\left\{\frac{\partial \mathbf{u}_k^H}{\partial \theta_k}\cdot\nabla_{\mathbf{u}_k} f\right\}, \tag{18}$$

where

$$\frac{\partial \mathbf{u}_k^H}{\partial \theta_k} = -2\theta_k e^{-\theta_k^2}\frac{\mathbf{p}_k^H}{\|\mathbf{p}_k\|}. \tag{19}$$

Proof: The proof is relegated to Appendix. ∎

The result of $\nabla_{\hat{\mathbf{s}}} f$ in Proposition 1 depends on the cost function. For the MMSE criterion, the cost function is

$$f = \frac{1}{\tau}\sum_{k=1}^{K}\sum_{i=1}^{\tau}|\hat{s}_k(i)-s_k(i)|^2, \tag{20}$$

and the derivative is

$$\frac{\partial f}{\partial \hat{s}_k^*(i)} = \frac{\hat{s}_k(i)-s_k(i)}{\tau}. \tag{21}$$

For the MSR criterion, according to the relationship that Rate $= -\log_2$MSE [33], the cost function is

$$f = \sum_{k=1}^{K}\log_2\left[\frac{1}{\tau}\sum_{i=1}^{\tau}|\hat{s}_k(i)-s_k(i)|^2\right], \tag{22}$$

for which

$$\frac{\partial f}{\partial \hat{s}_k^*(i)} = \frac{\hat{s}_k(i)-s_k(i)}{\sum_{i=1}^{\tau}|\hat{s}_k(i)-s_k(i)|^2}. \tag{23}$$

Based on Proposition 1, all the APs and the UEs can update their processing weights based on the derivatives (11), (15) and (18) in a distributed manner. Indeed, combining (11) and (12) yields

$$\nabla_{\mathbf{v}_l} f = \mathbf{y}_l\left[\mathbf{W}_{(l,:)}\cdot\nabla_{\hat{\mathbf{s}}} f\right]^*, \tag{24}$$

where $\mathbf{y}_l$ is locally available to the l-th AP, and $\mathbf{W}_{(l,:)}$ and $\nabla_{\hat{\mathbf{s}}} f$ are available to the CPU. Given that the CPU unicast the scalar signal $\left[\mathbf{W}_{(l,:)}\cdot\nabla_{\hat{\mathbf{s}}} f\right]^*$ to the l-th AP through the fronthaul link, the AP can obtain the derivative (24) without any explicit channel information.

In a way similar to the AP's obtaining $\nabla_{\mathbf{v}_l} f$, the k-th UE can obtain the derivatives in (15) and (18). Note that $\nabla_{\mathbf{p}_k}\mathbf{u}_k^*$, $\nabla_{\mathbf{p}_k}\mathbf{u}_k$, and $\frac{\partial \mathbf{u}_k^H}{\partial \theta_k}$ are locally available to the k-th UE as can be seen from (16), (17), and (19). Combining (13) and (14) yields

$$\nabla_{\mathbf{u}_k} f = s_k^*\left(\sum_{l=1}^{L}\mathbf{H}_{l,k}^H\mathbf{v}_l\frac{\partial f}{\partial r_l^*}\right). \tag{25}$$

Given that the l-th AP broadcast the beamformed derivative $\left(\mathbf{v}_l\frac{\partial f}{\partial r_l^*}\right)^*$ to the UE through the reverse channel, the k-th UE will receive the combined signal $\sum_{l=1}^{L}\mathbf{H}_{l,k}^T\left(\mathbf{V}\frac{\partial f}{\partial r_l^*}\right)^*$ owing to the channel reciprocity; thus, the UE can obtain the derivative (25) and the derivatives (15) and (18) without knowing explicit CSI.

Now we see that the distributed APs and UEs can obtain the derivatives of their respective weights through two phases: the

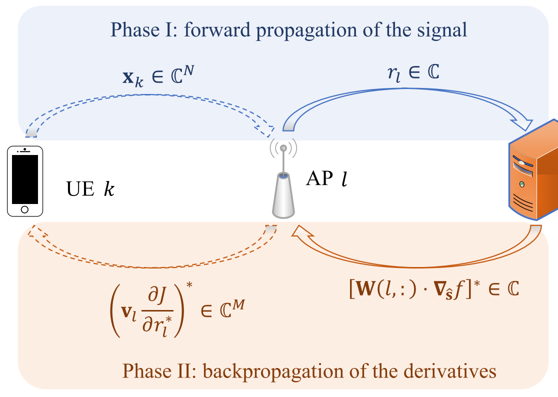
Phase I: forward propagation of the signal
$\mathbf{x}_k \in \mathbb{C}^N$
$r_l \in \mathbb{C}$
UE k
AP l
CPU
$\left(\mathbf{v}_l \frac{\partial J}{\partial r_l^*}\right)^* \in \mathbb{C}^M$
$[\mathbf{W}(l,:) \cdot \nabla_{\hat{\mathbf{s}}} f]^* \in \mathbb{C}$
Phase II: backpropagation of the derivatives

Fig. 4. Illustration of two phases in the DLCB algorithm: the forward propagation of the signals and the backpropagation of the derivatives.

forward propagation of the signals, and the backpropagation of the derivatives as shown in Fig. 4. In Phase I, the k-th UE transmits the beamformed signal $\mathbf{x}_k = \mathbf{u}_k s_k = \frac{\mathbf{p}_k}{\|\mathbf{p}_k\|} e^{-\theta_k^2} s_k$, the l-th AP compresses the received vector signal and forwards $r_l = \mathbf{v}_l^H \mathbf{y}_l$, and then the CPU applies an equalizer to the received signal to obtain $\hat{\mathbf{s}} = \mathbf{W}^H \mathbf{r}$. In Phase II, the CPU uses the received signal $\mathbf{r}$ and the pilot $\mathbf{s}$ to update its own weight $\mathbf{W}$ according to (9), and then transmits the beamformed derivative $\left[\mathbf{W}_{(l,:)} \cdot \nabla_{\hat{\mathbf{s}}} f\right]^*$ to the AP; the AP obtains $\nabla_{\mathbf{v}_l} f$ as shown in (24) and transmits the beamformed derivative $\left(\mathbf{v}_l \frac{\partial f}{\partial r_l^*}\right)^*$ to the UE so that the UE obtains $(\nabla_{\mathbf{u}_k} f)^*$ as shown in (25), and hence $(\nabla_{\mathbf{p}_k} f)^*$ and $\left(\frac{\partial f}{\partial \theta_k}\right)^*$ as shown in (15) and (18), respectively. Through one round of the forward propagation of the signals and the backpropagation of the derivatives, the nodes can update the derivatives and the weights $\mathbf{p}_k$, θ_k, $\mathbf{v}_l$, and $\mathbf{W}$ without knowing explicit CSI [cf. (18), (15), (11), and (9)].

The derivatives given in Proposition 1 are based on a one-sample pilot $\mathbf{s}(i)$, whereas the cost functions in (20) and (22) include a set of τ-length pilots. Accordingly, we have τ results for the derivative of f with respect to each weight and can average them to mitigate the channel noise:

$$\bar{\mathbf{d}}_\Theta = \frac{1}{\tau} \sum_{i=1}^{\tau} \frac{\partial f}{\partial \Theta^*}(i), \quad \Theta \in \left\{\{\mathbf{p}_k, \theta_k\}_{k=1}^K, \{\mathbf{v}_l\}_{l=1}^L\right\}. \tag{26}$$

Furthermore, given multiple sets of pilot sequences, we can use the exponentially weighted moving average (EWMA) [34] to iteratively update the derivatives as

$$\mathbf{d}_\Theta(t) = \lambda \mathbf{d}_\Theta(t-1) + (1-\lambda)\bar{\mathbf{d}}_\Theta(t), \tag{27}$$

where $t \in \{1, 2, \ldots T\}$ and $\lambda \in [0, 1)$ is an adjustable coefficient that determines the memory length [35]. In-depth mathematical analyses of EWMA can be founded in [34], [35], [36], [37], and [38], while extensions and improvements of EWMA are discussed in [38], [39], and [40]. The processing coefficients of the UEs and the APs are then updated by[1]

$$\Theta(t) = \Theta(t-1) - \alpha \mathbf{d}_\Theta(t), \tag{28}$$

[1]In general, the update for a complex variable $\mathbf{x}$ to minimize $f(\mathbf{x})$ is $\mathbf{x} \leftarrow \mathbf{x} - \alpha \frac{\partial f}{\partial \mathbf{x}^*}$ not $\mathbf{x} \leftarrow \mathbf{x} - \alpha \frac{\partial f}{\partial \mathbf{x}}$, since $\frac{\partial f}{\partial \mathbf{x}^*} = \frac{1}{2}\left(\frac{\partial f}{\partial \mathrm{Re}\{\mathbf{x}\}} + j \frac{\partial f}{\partial \mathrm{Im}\{\mathbf{x}\}}\right)$, while $\frac{\partial f}{\partial \mathbf{x}} = \frac{1}{2}\left(\frac{\partial f}{\partial \mathrm{Re}\{\mathbf{x}\}} - j \frac{\partial f}{\partial \mathrm{Im}\{\mathbf{x}\}}\right)$.

Algorithm 1 The DLCB Scheme

```
Initialization: Randomly initialize $\mathbf{W}(0)$, $\mathbf{v}_l(0)$, $\mathbf{p}_k(0)$, $\theta_k(0)$; $\alpha$, $\lambda$; $\mathbf{d}_{\mathbf{W}}(0) = 0$, $\mathbf{d}_{\mathbf{v}_l}(0) = 0$, $\mathbf{d}_{\mathbf{p}_k}(0) = 0$, $\mathbf{d}_{\theta_k}(0) = 0$.
Input: $T$ sets of $\tau$-length pilot sequences $\{s_k(i), k = 1, \cdots, K, i = 1, \cdots, \tau\}_{t=1}^T$.
Output: $\mathbf{W}$, $\{\mathbf{v}_l\}_{l=1}^L$, $\{\mathbf{p}_k, \theta_k\}_{k=1}^K$.
1: Synchronize the pilot sequences.
2: for $t = 1, 2, \ldots, T$ do
3:     Each UE applies the beamforming weight $\mathbf{u}_k(t-1) = \frac{\mathbf{p}_k(t-1)}{\|\mathbf{p}_k(t-1)\|} e^{-[\theta_k(t-1)]^2}$ to the pilot sequence $\{s_k(i), i = 1, \ldots, \tau\}_t$ by (1) and then transmits it to the AP.
4:     The APs receive the signals $\mathbf{y}_l(i), l = 1, \cdots, L, i = 1, \ldots, \tau$ in (2), apply the weights $\mathbf{v}_l(t-1)$ to obtain $r_l(i) = \mathbf{v}_l(t-1)^H \mathbf{y}_l(i)$ and forward them to the CPU.
5:     The CPU combines signal $\mathbf{r}$ with the weight $\mathbf{W}$ to obtain $\hat{\mathbf{s}}(i) = \mathbf{W}(t-1)^H \mathbf{r}(i), i = 1, \cdots, \tau$.
6:     The CPU calculates $\nabla_{\hat{\mathbf{s}}} f(i), i = 1, \cdots, \tau$ by (21) or (23), then passes the beamformed signal $\mathbf{W}_{(l,:)}(t-1) \cdot [\nabla_{\hat{\mathbf{s}}} f(i)]^*, i = 1, \cdots, \tau$ to the AP, and updates $\mathbf{W}(t)$ according to (9).
7:     The APs use the feedback signal from the CPU to compute $\nabla_{\mathbf{v}_l} f(i), i = 1, \cdots, \tau$ by (24), $\bar{\mathbf{d}}_{\mathbf{v}_l}(t)$ by (26), and $\mathbf{d}_{\mathbf{v}_l}(t)$ by (27), broadcast the beamformed signal $\left[\mathbf{v}_l(t-1) \frac{\partial f}{\partial r_l^*(i)}\right]^*$, $i = 1, \ldots, \tau$ to the UE, and updates $\mathbf{v}_l(t)$ according to (28).
8:     The UEs receive $\sum_{l=1}^L \mathbf{H}_{l,k}^T \left[\mathbf{v}_l(t-1) \frac{\partial f}{\partial r_l^*(i)}\right]^*$, $i = 1, \cdots, \tau$, obtain $\nabla_{\mathbf{u}_k} f(i), i = 1, \cdots, \tau$ by (25), compute $\nabla_{\mathbf{p}_k} f(i), i = 1, \cdots, \tau$ by (15), $\bar{\mathbf{d}}_{\mathbf{p}_k}(t)$ by (26), $\mathbf{d}_{\mathbf{p}_k}(t)$ by (27), and updates $\mathbf{p}_k(t)$ according to (28). Likewise, the UEs compute $\frac{\partial f}{\partial \theta_k^*}(i), i = 1, \cdots, \tau$ by (18), $\bar{\mathbf{d}}_{\theta_k}(t)$ by (26), $\mathbf{d}_{\theta_k}(t)$ by (27), and update $\theta_k(t)$ according to (28).
9: end for
```

where $\alpha \in (0, 1)$ is the learning rate, also known as the step size, which is chosen to strike a balance between convergence speed and accuracy [13]. Further insights on the choice and convergence of α can be found in [29] and [41].

The DLCB scheme is summarized in Algorithm 1, where Step $3-5$ correspond to the signal forward propagation in Fig. 4, and Step $6-8$ correspond to the backpropagation in Fig. 4. The forward-backward procedure is iterated T times in the for-loop.

In addition, the proposed DLCB algorithm, a first-order optimization algorithm, is applicable to a wide variety of objective functions, which should generally be restricted to be Lipschitz continuous or to have Lipschitz continuous derivatives [32, Section 4.3]. For different objective functions, only the derivative $\frac{\partial f}{\partial \hat{s}_k^*(i)}$ differs, while all the other forward propagation and backpropagation remain the same. So the DLCB scheme is applicable to various objective functions.

Remark 2: We can generalize the proposed DLCB scheme to the scenario where the AP generates Q-dimension vectors. Specifically, the weights of the l-th AP and the CPU are $\mathbf{V}_l \in \mathbb{C}^{M \times Q}$ *and* $\mathbf{W} \in \mathbb{C}^{QL \times K}$*, respectively. And* $L \geq K$ *is relaxed to* $QL \geq K$*. Compared with the results for one-output APs,*

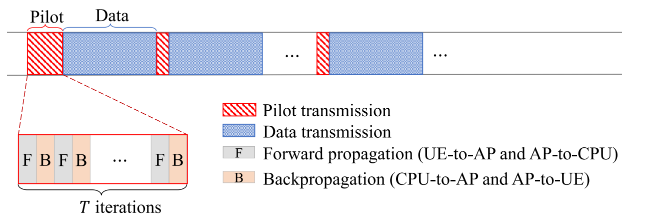
Pilot
Data
...
F B F B ... F B
T iterations
Pilot transmission
Data transmission
Forward propagation (UE-to-AP and AP-to-CPU)
Backpropagation (CPU-to-AP and AP-to-UE)

Fig. 5. Simplified frame structure for the DLCB algorithm with forward propagation and backpropagation.

the differences are

$$\nabla_{\mathbf{V}_l} f = \mathbf{y}_l \left(\frac{\partial f}{\partial \mathbf{r}_l^*} \right)^H \in \mathbb{C}^{M \times Q}, \tag{29}$$

where $\nabla_{\mathbf{r}_l} f$ *is the* l*-th block of* $\nabla_{\mathbf{r}} f = \mathbf{W} \cdot \nabla_{\hat{\mathbf{s}}} f \in \mathbb{C}^{QL \times 1}$ *and*

$$\nabla_{\mathbf{x}} f = \sum_{l=1}^{L} \mathbf{H}_l^H \mathbf{V}_l \cdot \nabla_{\mathbf{r}_l} f \in \mathbb{C}^{KN}. \tag{30}$$

B. A Frame Design to Support the DLCB Algorithm and Its Implement in 5G 3GPP NR

As shown in Fig. 4, the DLCB algorithm works for TDD systems with channel reciprocity and transfers information in both forward and backward directions. To support the T rounds of OTA training of the DLCB algorithm, we design the frame structure as shown in Fig. 5. After the initial T rounds of the OTA training in Fig. 5, the UEs can transmit their payload data. Since a wireless channel is typically time-varying, cell-free networks need to be retrained every once in a while. But the subsequent training can be significantly less frequent since the previously obtained weights can be used as a "warm start"; thus, as shown in Fig. 5, the training sessions after the initial one can be significantly sparser and briefer as verified by the simulations.

Indeed, the OTA forward and backward signaling can be integrated into the 5G 3GPP NR frame/slot structure as illustrated in [13] and [17]. In a similar vein, we show in the next how the DLCB scheme can also be implemented into the 5G-NR frame/slot structure.

A 5G-NR frame consists of 10 subframes, each of which spans 8 slots and can be divided into pilot transmission phases and data transmission phases [13, Fig. 1. (b)]. While each slot contains 14 orthogonal frequency division multiplexing (OFDM) symbols, a minislot structure consisting of two OFDM symbols can accommodate for either uplink or downlink signaling [13], [17].

We integrate the frame of our DLCB scheme in Fig. 5 into the 5G-NR frame/slot structure as shown in Fig. 6. The training sessions occurs during the first slot and then the UEs transmit their payload data during the subsequent slots. The training slot contains up to seven minislots, of which every two consecutive minislots accommodate for one round of uplink-and-downlink training. Therefore, one training slot can support up to $14/4 = 3.5$ training iterations.

Remark 3: We can further reduce the overhead of the OTA signaling in the backpropagation by letting the l*-th AP broadcast one (instead of* τ*) signal to the UEs for each set of* τ*-length pilots. As shown in (26), for each*

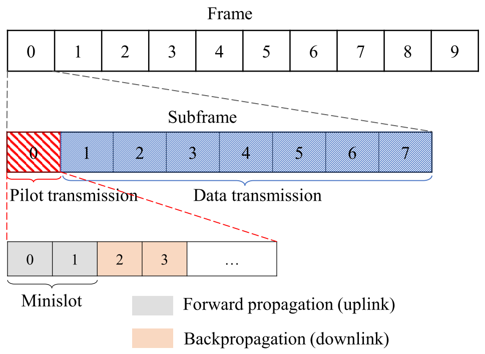

Fig. 6. Illustration of integrating the frame structure of the DLCB scheme into the 5G-NR frame/slot structure.

set of τ*-length pilots, each node needs to obtain* τ *derivatives, i.e.,* $\frac{\partial f}{\partial \Theta^*}(i), i = 1, \cdots, \tau$*. To this end, our proposed Algorithm 1 uses the most straightforward method: after the forward propagation of* τ *pilots, the CPU and each AP transmit* τ *beamformed derivatives, i.e.,* $\left[\mathbf{W}_{(l,:)}(i) \cdot \nabla_{\hat{\mathbf{s}}} f(i)\right]^*$ *and* $\left[\mathbf{v}_l(i) \frac{\partial f}{\partial r_l^*}(i)\right]^*$*,* $i = 1, \cdots, \tau$ *in the backpropagation, respectively (cf. Step 6 and Step 7 in Algorithm 1). Then each AP and each UE can obtain their own* τ *derivatives, i.e.,* $\nabla_{\mathbf{v}_l} f(i)$ *and* $\nabla_{\mathbf{u}_k} f(i)$*, for* $i = 1, \cdots, \tau$*, respectively (cf. (24) and (25)), and thus obtain the averages* $\overline{\mathbf{d}}_{\mathbf{v}}$*,* $\overline{\mathbf{d}}_{\mathbf{p}_k}$*, and* $\overline{\mathbf{d}}_{\theta_k}$ *in (26) [cf. (15) and (18)]. But now we propose another implementation to reduce the OTA overhead. Specifically, on the UE side, substituting (15) into (26) is*

$$\begin{aligned} \overline{\mathbf{d}}_{\mathbf{p}_k} &= \frac{1}{\tau} \sum_{i=1}^{\tau} \frac{\partial f}{\partial \mathbf{p}_k^*}(i) \\ &= \frac{1}{\tau} \sum_{i=1}^{\tau} \nabla_{\mathbf{p}_k} \mathbf{u}_k^*(i) \cdot \nabla_{\mathbf{u}_k} f(i) + \nabla_{\mathbf{p}_k} \mathbf{u}_k(i) \cdot [\nabla_{\mathbf{u}_k} f(i)]^* \\ &\overset{(a)}{=} \nabla_{\mathbf{p}_k} \mathbf{u}_k^* \cdot \overline{\mathbf{d}}_{\mathbf{u}_k} + \nabla_{\mathbf{p}_k} \mathbf{u}_k \cdot \left(\overline{\mathbf{d}}_{\mathbf{u}_k}\right)^*, \end{aligned} \tag{31}$$

where $\overline{\mathbf{d}}_{\mathbf{u}_k} \triangleq \frac{1}{\tau} \sum_{i=1}^{\tau} \nabla_{\mathbf{u}_k} f(i)$*;* $\overset{(a)}{=}$ *holds because* $\nabla_{\mathbf{p}_k} \mathbf{u}_k^*$ *and* $\nabla_{\mathbf{p}_k} \mathbf{u}_k$ *only depend on the UE's own weight [cf. (16) and (17)]; and according to (13)*

$$\overline{\mathbf{d}}_{\mathbf{u}_k} = \sum_{l=1}^{L} \mathbf{H}_{l,k}^H \mathbf{v}_l \left[\frac{1}{\tau} \sum_{i=1}^{\tau} s_k^*(i) \frac{\partial f}{\partial r_l^*}(i) \right], \tag{32}$$

where $s_k^*(i), i = 1, \cdots, \tau$ *are the known pilots and* $\frac{\partial f}{\partial r_l^*}(i), i = 1, \cdots, \tau$ *are the received signals of the AP in the backpropagation. Hence, the* l*-th AP can locally calculate* $\mathbf{v}_l \left[\frac{1}{\tau} \sum_{i=1}^{\tau} s_k^*(i) \frac{\partial f}{\partial r_l^*}(i)\right]$ *and broadcast it to the UEs. Then the UE will receive the signal* $\sum_{l=1}^{L} \mathbf{H}_{l,k}^T \mathbf{v}_l^* \left[\frac{1}{\tau} \sum_{i=1}^{\tau} s_k^*(i) \frac{\partial f}{\partial r_l^*}(i)\right]^*$ *owing to the channel reciprocity; thus, the UE can obtain* $\frac{1}{\tau} \sum_{i=1}^{\tau} \nabla_{\mathbf{u}_k} f(i)$ *in (32) and* $\overline{\mathbf{d}}_{\mathbf{p}_k}$ *in (31). Similarly, the UE can obtain* $\overline{\mathbf{d}}_{\theta_k}$ *(cf. (18), (19), and (26)). In doing so, for each set of pilots, the AP only needs to transmit one (not* τ*) signal to the UEs, which significantly reduces the OTA overhead of the algorithm.*

Remark 4: The OTA training between the UEs and the APs consumes time-frequency resource of the wireless network. To expediate the convergence and hence to reduce the required

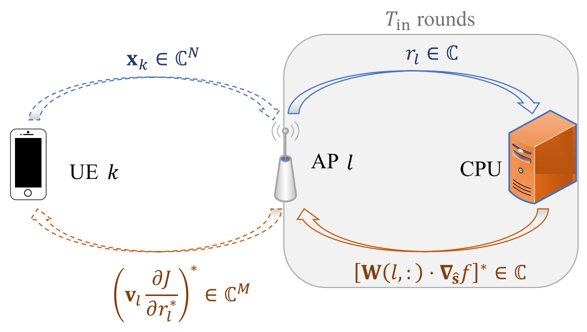
T_{in} rounds
$\mathbf{x}_k \in \mathbb{C}^N$
$r_l \in \mathbb{C}$
UE k
AP l
CPU
$[\mathbf{W}(l,:) \cdot \nabla_{\hat{\mathbf{s}}} f]^* \in \mathbb{C}$
$\left(\mathbf{v}_l \frac{\partial J}{\partial r_l^*}\right)^* \in \mathbb{C}^M$

Fig. 7. Illustration of T_{in} rounds of information exchanges via the fronthaul for each OTA interaction.

number of OTA interactions, we may use another implementation for each OTA training: the APs and the CPU can update their weights multiple times via $T_{\text{in}} \geq 2$ rounds of signaling exchanges over the cabled fronthaul as illustrated by Fig. 7. The underlying assumption of the aforementioned implementation is that the signaling exchange is less costly over the cabled fronthaul than via OTA. For real implementation, of course, T_{in} can be optimized according to the true costs of the signaling exchange over the fronthaul and OTA.

C. The Computational Complexity and Signaling Overhead

In the following, we analyze the computational complexity and the signaling overhead required by the DLCB scheme.

As the DLCB is a distributed algorithm, its computational burden is apportioned among the nodes. On the AP side, according to (11), each AP first needs $\mathcal{O}(M)$ multiplications to compute update direction of weight $\mathbf{y}_l(\frac{\partial f}{\partial r_l^*})^*$, where $(\frac{\partial f}{\partial r_l^*})^*$ is the received feedback signal from the CPU. On the UE side, substituting (13), (16), and (17) into (15) is

$$\nabla_{\mathbf{p}_k} f = \frac{e^{-\theta_k^2} s_k^*}{\|\mathbf{p}_k\|} \nabla_{\mathbf{x}_k} f - \frac{\mathcal{R}e\left\{e^{-\theta_k^2} s_k^* \mathbf{p}_k^H \nabla_{\mathbf{x}_k} f\right\}}{\|\mathbf{p}_k\|^3} \mathbf{p}_k \in \mathbb{C}^N, \tag{33}$$

and substituting (13) and (19) into (18) is

$$\frac{\partial f}{\partial \theta_k} = \frac{-4\theta_k}{\|\mathbf{p}_k\|} \mathcal{R}e\left\{e^{-\theta_k^2} s_k^* \mathbf{p}_k^H \nabla_{\mathbf{x}_k} f\right\}, \tag{34}$$

where the real part has been computed in (33). Note that $\nabla_{\mathbf{x}_k} f$ is the received feedback signal, in which the summations and matrix-vector multiplications are automatically achieved via the OTA propagation between the APs and the UEs. Hence, the computational complexity is $\mathcal{O}(N)$ for a UE to update its beamforming weight. In contrast, in each iteration of the Distributed-OTA algorithm from [28], each UE first uses the received signals and the pilots to compute the $N \times N$ matrix inverse to obtain its weight in the downlink transmission; and then each AP uses all the received signals and the pilots to compute the $M \times M$ matrix inverse to obtain its weight in the uplink transmission. Hence, the computational complexity is $\mathcal{O}(N^3)$ and $\mathcal{O}(M^3)$ for a UE and an AP to update the weight in the Distributed-OTA algorithm from [28], respectively. Therefore, the DLCB has much less computational complexity.

During the pilot transmission, the proposed Algorithm 1 and 2 need each AP to pass a scalar to the CPU in each iteration, whereas the algorithm from [28] does not need that and thus requires less training overhead on fronthaul. But in the data transmission, our algorithms and the algorithm from [28] need no iterations. The DLCB scheme still needs each AP to pass one scalar signal to the CPU over the fronthaul, but the Distributed-OTA algorithm in [28] requires each AP to transmit K scalar signals to the CPU; thus the fronthaul load required by our algorithm is only $\frac{1}{K}$ of that required by the algorithm from [28]. As in most communication scenarios, the time duration of data transmission is much longer than that of pilot transmission and $K \gg 1$ in a typical cell-free network, the overall communication burden upon the fronthaul by the DLCB is significantly less than the algorithm from [28]. For instance, if the ratio of the time duration allocated for pilot transmission and data transmission is $1 : 10$, for the proposed Algorithm 1 and 2 and the algorithm from [28], the load upon the fronthaul for pilot transmission is 1, T_{in}, and 0, respectively; and the fronthaul load for data transmission is 10, 10, and $10K$, respectively. Hence, the total cost on the fronthaul is 11, $10 + T_{\text{in}}$, and $10K$, respectively. Even for small numbers of UEs, for instance, $K = 4$ (but K is typically larger than 4 in practice), the fronthaul cost required by our algorithms is only about $1/4$ of that required by the Distributed-OTA algorithm from [28].

About the OTA training overhead, for our algorithms, one iteration consists of one uplink transmission and one downlink transmission, while that of [28] takes two uplink transmissions and one downlink transmission. Therefore, our algorithms require less OTA training than the algorithm of [28].

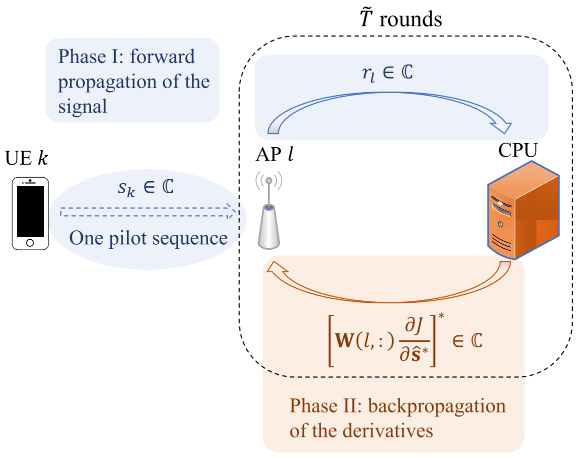
$\tilde{T}$ rounds
Phase I: forward propagation of the signal
$r_l \in \mathbb{C}$
UE k
AP l
CPU
$s_k \in \mathbb{C}$
One pilot sequence
$\left[\mathbf{W}(l,:)\frac{\partial J}{\partial \hat{\mathbf{s}}^*}\right]^* \in \mathbb{C}$
Phase II: backpropagation of the derivatives

Fig. 8. Illustration of two phases in the DLCB algorithm for cell-free networks with single-antenna UEs.

IV. The Special Scenario of Single-Antenna UEs

In this section, we consider a special and practically relevant case that each UE has only one transmit antenna. In this scenario, we can simplify the DLCB scheme to waive the backpropagation via OTA signaling between the APs and the UEs, but only via the fronthaul signaling between the APs and the CPU as shown in Fig. 8, which significantly reduce the complexity of the algorithm.

In the forward propagation, the single-antenna UE transmits the pilot signal directly, i.e., $x_k = s_k, k = 1 \cdots, K$. The AP receives $\mathbf{y}_l$ and forwards $r_l = \mathbf{v}_l^H \mathbf{y}_l$ to the CPU before the CPU yields the estimate $\hat{\mathbf{s}} = \mathbf{W}^H \mathbf{r}$. In the backpropagation, the CPU calculates its weights $\mathbf{W}$ by (9), and then transmits

Algorithm 2 The DLCB Algorithm for the Case of Single-Antenna UEs

```
Initialization: Randomly initialize $\mathbf{W}(0)$, $\mathbf{v}_l(0)$; $\alpha$, $\lambda$; $\mathbf{d}_{\mathbf{W}}(0) = 0$, $\mathbf{d}_{\mathbf{v}_l}(0) = 0$.
Input: One $\tau$-length pilot sequence $\{s_k(i), k = 1, \cdots, K, i = 1, \cdots, \tau\}$.
Output: $\mathbf{W}$, $\{\mathbf{v}_l\}_{l=1}^{L}$.
1: Synchronize the pilot sequences.
2: Each UE transmits the pilot sequence $\{s_k(i), i = 1, \ldots, \tau\}$ to the AP.
3: The AP receives the signals $\mathbf{y}_l(i), l = 1, \cdots, L, i = 1, \ldots, \tau$ in (2).
4: for $t = 1, 2, \ldots, \widetilde{T}$ do
5:     The AP applies the weight $\mathbf{v}_l(t-1)$ to obtain $r_l(i) = \mathbf{v}_l(t-1)^H \mathbf{y}(i), i = 1, \cdots, \tau$ by (3).
6:     The CPU combines signal $\mathbf{r} = [r_1, \cdots, r_L]^T$ with the weight $\mathbf{W}$ to obtain $\hat{\mathbf{s}}(i) = \mathbf{W}(t-1)^H \mathbf{r}(i), i = 1, \cdots, \tau$.
7:     The CPU passes the beamformed signal $\mathbf{W}_{(l,:)}(t-1) [\nabla_{\hat{\mathbf{s}}} f(i)]^*, i = 1, \cdots, \tau$ to the AP, and updates $\mathbf{W}(t)$ according to (9).
8:     The AP uses the feedback signal from the CPU to compute $\nabla_{\mathbf{v}_l} f(t)$ by (24) and $\mathbf{d}_{\mathbf{v}_l}(t)$ by (27), and updates $\mathbf{v}_l(t)$ according to (28).
9: end for
```

the beamformed derivative $\left[\mathbf{W}_{(l,:)} \cdot \nabla_{\hat{\mathbf{s}}} f\right]^*$ to the AP. Then the AP obtains $\nabla_{\mathbf{v}_l} f$ as shown in (24). Since the UE uses no weight, the APs need not to transmit the beamformed derivative to the UEs. Consequently, it is sufficient to use only one set of pilot sequences to train the APs and the CPU to update their weights via the fronthaul signaling. The DLCB algorithm for the case of single-antenna UEs is summarized in Algorithm 2.

In this special scenario, no OTA signaling is required other than the transmission of one τ-length pilot sequence. The UEs need not to calculate their beamforming weights; the APs need not to calculate the backpropagation. Despite the striking simplicity, the DLCB scheme can outperform the state-of-the-art "Level 3" method proposed in [18], as we will see soon.

V. Numerical Simulations

This section presents numerical examples to verify the effectiveness of the proposed DLCB algorithm. In the examples except for the last two, the channels between the APs and the UEs are assumed to be frequency-flat Rayleigh fading and remain static. In the first three examples, the channels $\mathbf{H}_k$'s are simulated without the path loss. The input SNRs of the different UEs are assumed to be the same value $\rho = \frac{1}{\sigma^2}$. The length of pilot sequence $\tau = 64$. The hyper-parameter and learning rate used in (27) and (28) are $\lambda = 0.9$ and $\alpha = 0.3$, respectively. Other parameters are given on the top of the figures.

According to the signal transmission process [cf. (1)-(4)], we can obtain the ideal sum rate of an uplink cell-free network with perfect CSI as

$$R = \sum_{k=1}^{K} R_k = \sum_{k=1}^{K} \log_2(1 + \gamma_k), \tag{35}$$

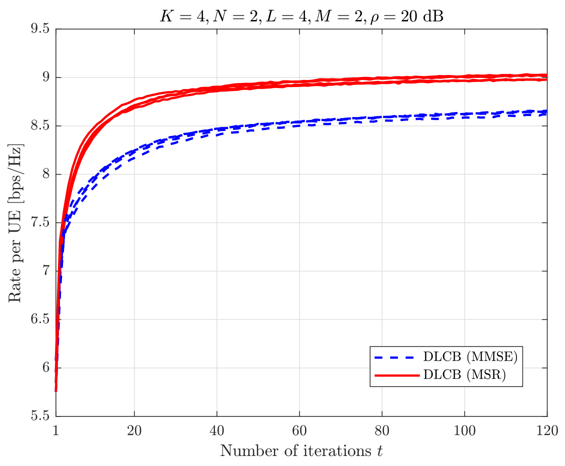

Fig. 9. Convergence of per-UE rate for the DLCB scheme using the MMSE and the MSR criteria.

with the output SINR of the kth UE being

$$\gamma_k = \frac{|\mathbf{w}_k^H \mathbf{G}_k \mathbf{u}_k|^2}{\|\mathbf{w}_k^H \mathbf{G} \mathbf{D}_u\|^2 + \sigma^2 \mathbf{w}^H \mathbf{D}_v \mathbf{w}}, \tag{36}$$

where

$$\mathbf{G} \triangleq \begin{bmatrix} \mathbf{v}_1^H \mathbf{H}_1 \\ \vdots \\ \mathbf{v}_L^H \mathbf{H}_L \end{bmatrix} \in \mathbb{C}^{L \times KN} \text{ with } \mathbf{G}_k \triangleq \begin{bmatrix} \mathbf{v}_1^H \mathbf{H}_{1,k} \\ \vdots \\ \mathbf{v}_L^H \mathbf{H}_{L,k} \end{bmatrix} \in \mathbb{C}^{L \times N}$$

being its k-th column block, $\mathbf{D}_u \triangleq \text{blkdiag}(\mathbf{u}_1, \cdots, \mathbf{u}_K) \in \mathbb{C}^{KN \times K}$ and $\mathbf{D}_v \triangleq \text{diag}(\|\mathbf{v}_1\|^2, \cdots, \|\mathbf{v}_L\|^2) \in \mathbb{C}^{L \times L}$. Note that expression (36) is only used as a metric to evaluate the performance of our proposed algorithms and is not needed in the optimization procedure. Based on the expression (35) and (36), in the simulations we can calculate the rate per UE $\{R_k, k = 1, \cdots, K\}$ or the sum rate R to evaluate the performance of the system based on the average of 100 Monte Carlo trials.

In the first example, we simulate the MMSE criterion in (20) and the MSR criterion in (22) to see the convergence of the DLCB algorithm. As shown in Fig. 9, the DLCB algorithm converges as the number of iterations increases. Here one iteration represents one round of forward and backward training sessions (cf. Fig. 4). The DLCB algorithm using the MSR criterion, labeled as DLCB (MSR), appears to outperform that using the MMSE criterion. Indeed, for the MMSE criterion, the derivative in (21) suffers from the so-called gradient vanishing issue as the iterations proceeds. In contrast, the DLCB using the MSR criterion is largely free from this issue owing to the denominator in (23).

In the second example, we simulate another implementation of the DLCB scheme as described in Remark 4. As shown in Fig. 10, having $T_{\text{in}} \geq 2$ rounds of the AP-to-CPU iterations over the fronthaul can expedite the convergence of the DLCB scheme, leading to fewer OTA iterations. In particular, only 15 OTA iterations are needed for $T_{\text{in}} = 2$ to achieve the performance of 30 iterations, and the OTA overhead is reduced by nearly 75% for $T_{\text{in}} = 5$.

In the subsequent performance comparisons, we simulate the same setting used in [18], where the APs are deployed on a square grid with the distance between neighboring APs

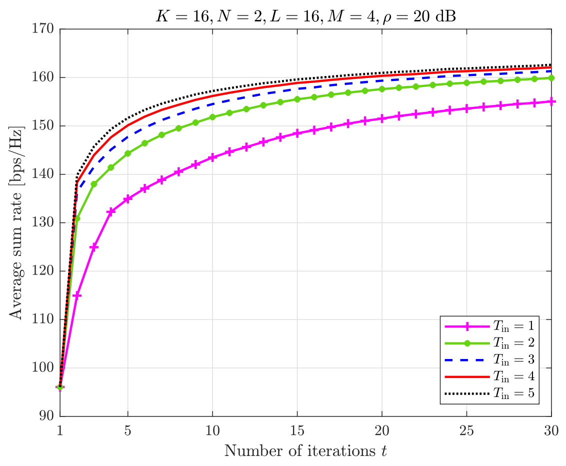

Fig. 10. Convergence of average sum rate under different number of the AP-to-CPU iterations.

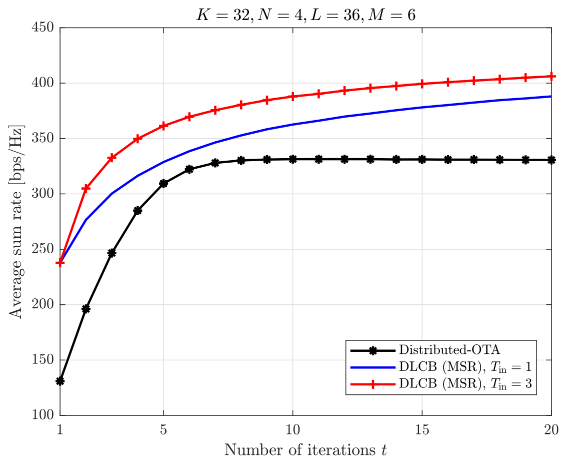

Fig. 11. Comparison of average sum rate between the DLCB algorithm and the Distributed-OTA algorithm proposed in [28] as the number of iterations.

of 50 m and minimum distance between UEs of 10 m; the UEs transmit with power 20 dBm, and the noise power is -96 dBm; and the path loss model is β_{kl} [dB] $= -30.5 - 3.67\log_{10}\left(\frac{d_{kl}}{1\,\text{m}}\right) + F_{kl}$,where d_{kl} is the distance between the k-th UE and the l-th AP and $F_{kl} \sim \mathcal{N}(0, 4^2)$ is the shadow fading.

We compare the convergence rate of the DLCB scheme with that of the Distributed-OTA algorithm proposed in [28], where $\rho_{\text{BS}} = 30$ dBm, $\sigma^2_{\text{UE}} = -96$ dBm. As shown in Fig. 11, our algorithms take no more than five iterations to outperform the final convergence value of the Distributed-OTA algorithm from [28] and always outperform the algorithm from [28] in any number of iterations. That is, the DLCB algorithm can significantly outperform the Distributed-OTA algorithm in both the sum rate and the convergence speed, even though the latter requires uplink signaling twice per OTA iteration.

Then we compare the impact of the signaling overhead on the DLCB scheme with that on the Distributed-OTA algorithm from [28] under the 5G-NR frame/slot structure. As illustrated in Sec. III-B, the training session occurs in the first slot and one training slot contains up to 3.5 training iterations. Considering the switching time of uplink and downlink signaling, we reserve two OFDM symbols to separate the uplink and downlink training time slots as illustrated in [13]. Hence, one

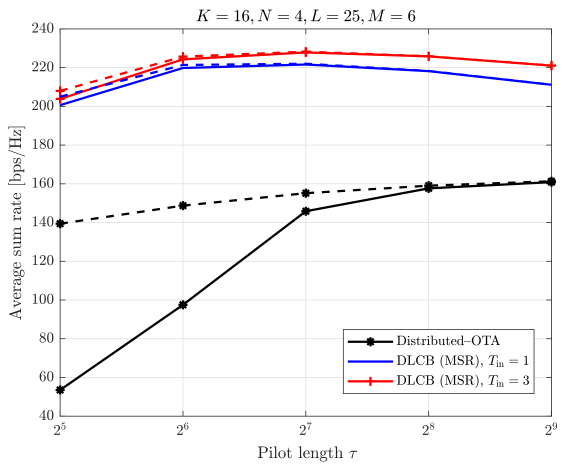

Fig. 12. Comparison of the DLCB scheme and the Distributed-OTA algorithm proposed in [28] as the length of orthogonal and non-orthogonal pilots.

training slot for the DLCB scheme contains $(14-2)/4 = 3$ training iterations, each of which occupies $14/3 \approx 4.67$ OFDM symbols. Furthermore, we consider that the scheduling blocks consist of N_f frames, during which the channels remain fixed, as discussed in [13]. Accordingly, the effective sum rate of the DLCB scheme after t iterations is

$$R^{(t)}_{\text{eff}} \triangleq \left(1 - \frac{14/6t}{10 \times 8 \times 14 \times N_f}\right) R^{(t)}, \tag{37}$$

where $R^{(t)}$ is the sum rate after t iterations [cf. (35)-(36)]. Therefore, the overhead ratio of the DLCB scheme, i.e., $\frac{14/3t}{1120N_f}$, is the same as that of the algorithm from [28]. Incorporating Fig. 11, we can deduce that the average effective sum rate of the DLCB scheme still outperforms that of the algorithm from [28].

We also consider a pilot-contaminated scenario by assuming non-orthogonal random pilots, which can be relevant in a large network. As shown in Fig. 12, the solid and dotted curves represent the scenario with ideal orthogonal and non-orthogonal random pilots, respectively. Fig. 12 shows that the proposed DLCB scheme is more robust against pilot contamination than the Distributed-OTA algorithm proposed in [28], whereas the Distributed-OTA algorithm needs pilot-aided CSI acquisition for optimizing the MMSE precoding and combining vectors at both the UEs and the APs. In addition, when the pilot length is reduced from 128 to 64, the performance of our algorithm remains almost unchanged. When the length is reduced to 32, the performance of our algorithm decreases by less than 9%. Note that the sum rate can slightly decrease if the pilot length increases over $\tau = 2^7$. This is because a lengthy pilot can expand the denominator in (23) and hence reduce the convergence speed.

We also simulate the special scenario of the single-antenna UEs. For comparison, we include the "Level 3" method and the fully centralized "Level 4" method proposed in [18], which only considers the single-antenna UEs. As shown in Fig. 13, our proposed DLCB algorithm converges fast and can outperform the "Level 3" method after very few iterations. Fig. 14 shows the CDF of the per-UE rate achieved by the DLCB algorithm, which is better than the Level 3 method in [18], even though no explicit CSI is required by our scheme and each AP only needs to pass the scalar sequences to the

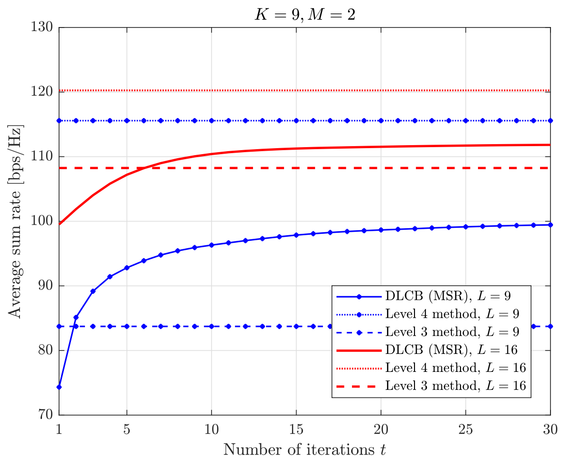

Fig. 13. Comparison of the DLCB algorithm and the Level 4 and Level 3 methods proposed in [18] as the number of iterations.

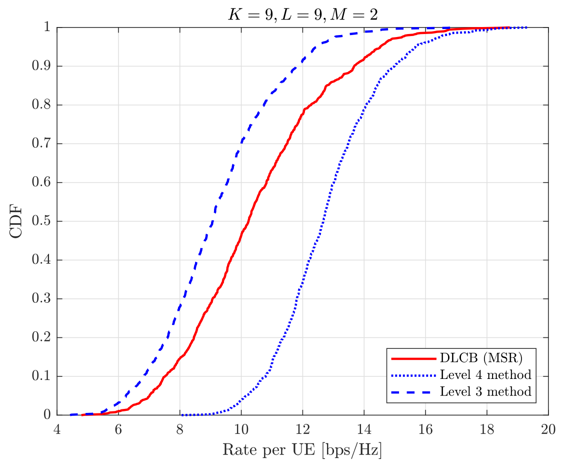

Fig. 14. CDF of the per-UE rate for the DLCB algorithm and the Level 4 and Level 3 methods proposed in [18].

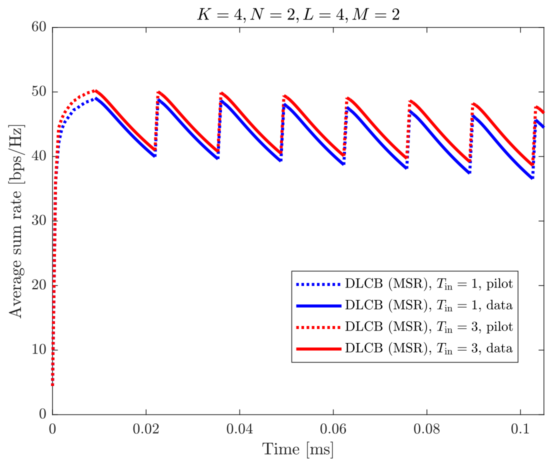

Fig. 15. Average sum rate of the DLCB algorithm in a time-varying channel.

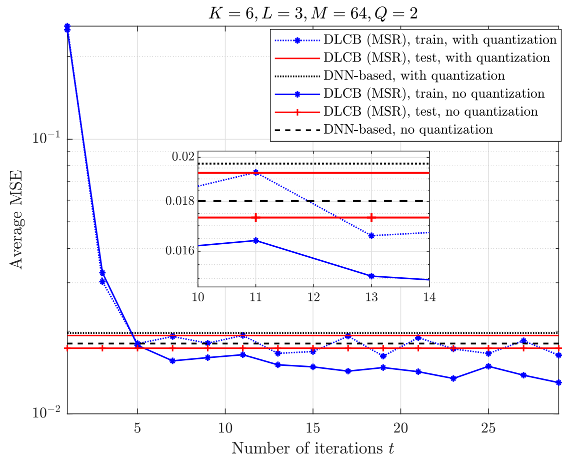

Fig. 16. Average MSE achieved by the DLCB algorithm and the DNN using local CSI (non-progressive) algorithm proposed in [42] as the number of iterations.

CPU over the fronthaul. In contrast, the 'Level 3" method in [18] requires each AP to estimate the channel and pass K-dimensional vector sequences to the CPU. In addition, the methods in [18] only consider the single-antenna UEs and assume much statistic information is available at the CPU [18, Table I]; but our algorithm has no such limitations. Fig. 13 shows as the number of APs L increases, the gap between our DLCB scheme and the benchmark decreases.

We then simulate a more practical scenario of the time-varying channel. Specifically, we set that the carrier frequency $f_c = 30$ GHz, the bandwidth $BW = 100$ MHz, i.e., the Nyquist sampling duration $T_s = \frac{1}{BW} = 0.01$ μs, and the channel mobility speed $v = 36$ km/h, i.e., the Doppler frequency spread $f_d = \frac{v}{c} f_c = \frac{36/3.6}{3\times10^8} \times 30 \times 10^9 \approx 1$ kHz. As shown in Fig. 15, in the initialization stage the DLCB scheme needs 14 sets of pilots for training that last for 9.1 μs, i.e., $14\times64\times0.01+14\times0.01 = 9.1$ μs [cf. Remark 3] before transmitting 20 sets of data; based on the first training, the subsequent training is warm-started, and hence only needs one iteration for only 0.65 μs followed by another 20 sets of the data payload. Although the performance of data transmission slowly degenerates owing to the time-varying channel, it only needs one iteration of training to compensate for the rate loss that occurred in the preceding data transmission. To sum up, our DLCB algorithms take only a moderate portion of resources to adapt to the changes of the channels in cell-free networks. Hence, our algorithms are applicable to static channels, block-fading channels, and time-varying channels as well.

The last example simulates the scenario of the Q-output APs [cf. Remark 2]. We include the "the DNN using local CSI (non-progressive) algorithm" in [42], which only considers the real-valued transmission and the single-antenna UEs. Hence, for comparison we use the real-valued weights and simulate the scenario of [42], that is, $\mathbf{s} \sim \mathcal{N}(\mathbf{0},\mathbf{I})$, $\mathbf{H}_l \sim \mathcal{N}(\mathbf{0},\mathbf{I})$, $N=1, K=6, L=3, M=64$, $\rho = 0$ dB, and 10^4 channel realizations. The algorithm of [42] considers the quantization operation at each AP [cf. (14)-(18) in [42]], which can be directly incorporated into the DLCB scheme, so we also simulate the DLCB scheme with the quantization operation for a fair comparison. The algorithm of [42] uses 10^5 training data, whereas our algorithm still uses $\tau = 64$. As shown in Fig. 16, the DLCB algorithm can outperform the algorithm of [42] (labeled as "DNN-based"), even though the latter requires the APs' local CSI and much more training data.

VI. Conclusion

In this paper, we transform the problem of cell-free massive MIMO optimization to an unconstrained nonlinear one and relate an uplink cell-free network to a quasi-neural network. In doing so, we propose a distributed learning scheme inspired by backpropagation algorithm, which can optimize the weights of the UEs, the APs, and the CPU based on a set of pilot sequences assuming no explicit CSI. The optimization is conducted via limited OTA signaling and fronthaul signaling, while requiring no information exchanges between the APs. To support the scheme, we also present a frame design and provide the complexity and signaling analysis. It is shown that the scheme needs less fronthaul signaling and fewer complex-valued multiplications than the state-of-the-art methods; thus, it is scalable. It is also applicable to various objective functions, such as the MMSE criterion and the MSR criterion. The extensive simulations verify the effectiveness of the proposed scheme.

Appendix
Proof to Proposition 1

We first reproduce the following lemma from [43].

Lemma 5 [43, Theorem 3.3]: Let $f : \mathbb{C}^{N\times Q}\times\mathbb{C}^{N\times Q}\rightarrow\mathbb{R}$. Then the following holds:

$$\nabla_{\mathbf{Z}^*} f = (\nabla_{\mathbf{Z}} f)^* \tag{38}$$

According to the chain rule of derivative and Lemma 5,

$$\nabla_{\mathbf{v}_l} f = \nabla_{\mathbf{v}_l} r_l \cdot \frac{\partial f}{\partial r_l} = \nabla_{\mathbf{v}_l} r_l \cdot \left(\frac{\partial f}{\partial r_l^*}\right)^*. \tag{39}$$

Since it follows from (3) that $\nabla_{\mathbf{v}_l} r_l = \mathbf{y}_l$, (39) can be reformulated as (11).

According to the chain rule of derivative, we can prove (13) as

$$\nabla_{\mathbf{u}_k} f = \nabla_{\mathbf{u}_k} \mathbf{x}_k^* \cdot \nabla_{\mathbf{x}_k} f \overset{(a)}{=} s_k^* \cdot \nabla_{\mathbf{x}_k} f, \tag{40}$$

where $\overset{(a)}{=}$ holds because $\nabla_{\mathbf{u}_k}\mathbf{x}_k^* = s_k^*\mathbf{I}_N$ [cf. (1)].

As for $\nabla_{\mathbf{x}_k} f$ in (13), it is the k-th block of $\nabla_{\mathbf{x}} f$ in (14). We can use the chain rule of derivative to obtain (14) as

$$\nabla_{\mathbf{x}} f = \sum_{l=1}^{L} \nabla_{\mathbf{x}} \mathbf{y}_l^* \cdot \nabla_{\mathbf{y}_l} f \overset{(b)}{=} \sum_{l=1}^{L} \mathbf{H}_l^H \nabla_{\mathbf{y}_l} f \overset{(c)}{=} \sum_{l=1}^{L} \mathbf{H}_l^H \mathbf{v}_l \frac{\partial f}{\partial r_l^*} \tag{41}$$

where $\overset{(b)}{=}$ holds because $\nabla_{\mathbf{x}}\mathbf{y}_l^* = \mathbf{H}_l^H$ [cf. (2)], and $\overset{(c)}{=}$ holds because $\nabla_{\mathbf{y}_l} f = \nabla_{\mathbf{y}_l} r_l^* \cdot \frac{\partial f}{\partial r_l^*} = \mathbf{v}_l \frac{\partial f}{\partial r_l^*}$, where $\nabla_{\mathbf{y}_l} r_l^* = \mathbf{v}_l$ [cf. (3)].

According to our proposed beamforming weight per UE ($\mathbf{u}_k = \frac{\mathbf{p}_k}{\|\mathbf{p}_k\|} e^{-\theta_k^2}, \mathbf{p}_k \in \mathbb{C}^N, \theta_k \in \mathbb{R}$) and the chain rule of derivative, we can obtain

$$\nabla_{\mathbf{p}_k} f = \nabla_{\mathbf{p}_k} \mathbf{u}_k^* \cdot \nabla_{\mathbf{u}_k} f + \nabla_{\mathbf{p}_k} \mathbf{u}_k \cdot \nabla_{\mathbf{u}_k^*} f,$$

and

$$\begin{aligned}\frac{\partial f}{\partial \theta_k} &= \frac{\partial \mathbf{u}_k^H}{\partial \theta_k} \nabla_{\mathbf{u}_k} f + \frac{\partial \mathbf{u}_k^T}{\partial \theta_k} \nabla_{\mathbf{u}_k^*} f \\ &= \frac{\partial \mathbf{u}_k^H}{\partial \theta_k} \nabla_{\mathbf{u}_k} f + \left(\frac{\partial \mathbf{u}_k^H}{\partial \theta_k^*}\right)^* (\nabla_{\mathbf{u}_k} f)^* \\ &= 2\mathcal{R}e\left\{\frac{\partial \mathbf{u}_k^H}{\partial \theta_k} \nabla_{\mathbf{u}_k} f\right\},\end{aligned}$$

i.e., (15) and (18), where $\nabla_{\mathbf{u}_k^*} f = (\nabla_{\mathbf{u}_k} f)^*$ [cf. Lemma 5], and $\nabla_{\mathbf{u}_k} f$ has been proven [cf. 13].

As for $\nabla_{\mathbf{p}_k} \mathbf{u}_k^*$, $\nabla_{\mathbf{p}_k} \mathbf{u}_k$ in (15) and $\frac{\partial \mathbf{u}_k^H}{\partial \theta_k}$ in (18), we can use $\mathbf{u}_k = \frac{\mathbf{p}_k}{\|\mathbf{p}_k\|} e^{-\theta_k^2}$ to obtain

$$\begin{aligned}\nabla_{\mathbf{p}_k} \mathbf{u}_k^* &= \frac{\partial \mathbf{p}_k^H (\mathbf{p}_k^H \mathbf{p}_k)^{-\frac{1}{2}} e^{-\theta_k^2}}{\partial \mathbf{p}_k^*} \\ &= \mathbf{I}_N \frac{1}{\|\mathbf{p}_k\|} e^{-\theta_k^2} - \mathbf{p}_k \mathbf{p}_k^H \frac{1}{2\|\mathbf{p}_k\|^3} e^{-\theta_k^2} \\ &= \frac{e^{-\theta_k^2}}{\|\mathbf{p}_k\|}\left(\mathbf{I}_N - \frac{\mathbf{p}_k \mathbf{p}_k^H}{2\|\mathbf{p}_k\|^2}\right), \end{aligned} \tag{42}$$

$$\nabla_{\mathbf{p}_k} \mathbf{u}_k = \frac{\partial \mathbf{p}_k^T (\mathbf{p}_k^H \mathbf{p}_k)^{-\frac{1}{2}} e^{-\theta_k^2}}{\partial \mathbf{p}_k^*} = -\frac{e^{-\theta_k^2}}{2\|\mathbf{p}_k\|^3} \mathbf{p}_k \mathbf{p}_k^T, \tag{43}$$

and

$$\frac{\partial \mathbf{u}_k^H}{\partial \theta_k} = \frac{\partial e^{-\theta_k^2}}{\partial \theta_k} \frac{\mathbf{p}_k^H}{\|\mathbf{p}_k\|} = -2\theta_k e^{-\theta_k^2} \frac{\mathbf{p}_k^H}{\|\mathbf{p}_k\|}, \tag{44}$$

i.e., (16), (17), and (19).

Inserting (13), (16) and (17) into (15) yields $\frac{\partial f}{\partial \mathbf{p}_k^*}$. Inserting (13) and (19) into (18) leads to $\frac{\partial f}{\partial \theta_k^*}$,

References

[1] M. Matthaiou, O. Yurduseven, H. Q. Ngo, D. Morales-Jimenez, S. L. Cotton, and V. F. Fusco, "The road to 6G: Ten physical layer challenges for communications engineers," *IEEE Commun. Mag.*, vol. 59, no. 1, pp. 64–69, Jan. 2021.

[2] Ö. T. Demir, E. Björnson, and L. Sanguinetti, "Foundations of user-centric cell-free massive MIMO," *Found. Trends Signal Process.*, vol. 14, nos. 3–4, pp. 162–472, 2021.

[3] G. Interdonato, E. Björnson, H. Quoc Ngo, P. Frenger, and E. G. Larsson, "Ubiquitous cell-free massive MIMO communications," *EURASIP J. Wireless Commun. Netw.*, vol. 2019, no. 1, pp. 1–13, Dec. 2019.

[4] H. Q. Ngo, A. Ashikhmin, H. Yang, E. G. Larsson, and T. L. Marzetta, "Cell-free massive MIMO versus small cells," *IEEE Trans. Wireless Commun.*, vol. 16, no. 3, pp. 1834–1850, Mar. 2017.

[5] E. Björnson and L. Sanguinetti, "A new look at cell-free massive MIMO: Making it practical with dynamic cooperation," in *Proc. IEEE 30th Annu. Int. Symp. Pers., Indoor Mobile Radio Commun. (PIMRC)*, Sep. 2019, pp. 1–6.

[6] G. Interdonato, P. Frenger, and E. G. Larsson, "Scalability aspects of cell-free massive MIMO," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2019, pp. 1–6.

[7] E. Björnson and L. Sanguinetti, "Scalable cell-free massive MIMO systems," *IEEE Trans. Commun.*, vol. 68, no. 7, pp. 4247–4261, Jul. 2020.

[8] S. Buzzi and C. D'Andrea, "Cell-free massive MIMO: User-centric approach," *IEEE Wireless Commun. Lett.*, vol. 6, no. 6, pp. 706–709, Dec. 2017.

[9] S. Buzzi, C. D'Andrea, A. Zappone, and C. D'Elia, "User-centric 5G cellular networks: Resource allocation and comparison with the cell-free massive MIMO approach," *IEEE Trans. Wireless Commun.*, vol. 19, no. 2, pp. 1250–1264, Feb. 2020.

[10] M. Alonzo, S. Buzzi, A. Zappone, and C. D'Elia, "Energy-efficient power control in cell-free and user-centric massive MIMO at millimeter wave," *IEEE Trans. Green Commun. Netw.*, vol. 3, no. 3, pp. 651–663, Sep. 2019.

[11] G. Interdonato, M. Karlsson, E. Björnson, and E. G. Larsson, "Local partial zero-forcing precoding for cell-free massive MIMO," *IEEE Trans. Wireless Commun.*, vol. 19, no. 7, pp. 4758–4774, Jul. 2020.

[12] B. Gouda, I. Atzeni, and A. Tölli, "Distributed precoding design for cell-free massive MIMO systems," in *Proc. IEEE 21st Int. Workshop Signal Process. Adv. Wireless Commun. (SPAWC)*, May 2020, pp. 1–5.

[13] I. Atzeni, B. Gouda, and A. Tölli, "Distributed precoding design via over-the-air signaling for cell-free massive MIMO," *IEEE Trans. Wireless Commun.*, vol. 20, no. 2, pp. 1201–1216, Feb. 2021.

[14] E. Nayebi, A. Ashikhmin, T. L. Marzetta, H. Yang, and B. D. Rao, "Precoding and power optimization in cell-free massive MIMO systems," *IEEE Trans. Wireless Commun.*, vol. 16, no. 7, pp. 4445–4459, Jul. 2017.

[15] L. Du, L. Li, H. Q. Ngo, T. C. Mai, and M. Matthaiou, "Cell-free massive MIMO: Joint maximum-ratio and zero-forcing precoder with power control," *IEEE Trans. Commun.*, vol. 69, no. 6, pp. 3741–3756, Jun. 2021.

[16] A. Ashikhmin, L. Li, and T. L. Marzetta, "Interference reduction in multi-cell massive MIMO systems with large-scale fading precoding," *IEEE Trans. Inf. Theory*, vol. 64, no. 9, pp. 6340–6361, Sep. 2018.

[17] A. Tolli et al., "Distributed coordinated transmission with forward-backward training for 5G radio access," *IEEE Commun. Mag.*, vol. 57, no. 1, pp. 58–64, Jan. 2019.

[18] E. Björnson and L. Sanguinetti, "Making cell-free massive MIMO competitive with MMSE processing and centralized implementation," *IEEE Trans. Wireless Commun.*, vol. 19, no. 1, pp. 77–90, Jan. 2020.

[19] H. Q. Ngo, L. Tran, T. Q. Duong, M. Matthaiou, and E. G. Larsson, "Energy efficiency optimization for cell-free massive MIMO," in *Proc. IEEE 18th Int. Workshop Signal Process. Adv. Wireless Commun. (SPAWC)*, Jul. 2017, pp. 1–5.

[20] G. Interdonato, H. Q. Ngo, P. Frenger, and E. G. Larsson, "Downlink training in cell-free massive MIMO: A blessing in disguise," *IEEE Trans. Wireless Commun.*, vol. 18, no. 11, pp. 5153–5169, Nov. 2019.

[21] H. Liu, J. Zhang, S. Jin, and B. Ai, "Graph coloring based pilot assignment for cell-free massive MIMO systems," *IEEE Trans. Veh. Technol.*, vol. 69, no. 8, pp. 9180–9184, Aug. 2020.

[22] J. Zhang, Y. Wei, E. Björnson, Y. Han, and S. Jin, "Performance analysis and power control of cell-free massive MIMO systems with hardware impairments," *IEEE Access*, vol. 6, pp. 55302–55314, 2018.

[23] X. Hu, C. Zhong, X. Chen, W. Xu, H. Lin, and Z. Zhang, "Cell-free massive MIMO systems with low resolution ADCs," *IEEE Trans. Commun.*, vol. 67, no. 10, pp. 6844–6857, Oct. 2019.

[24] H. Masoumi and M. J. Emadi, "Performance analysis of cell-free massive MIMO system with limited fronthaul capacity and hardware impairments," *IEEE Trans. Wireless Commun.*, vol. 19, no. 2, pp. 1038–1053, Feb. 2020.

[25] M. Bashar, K. Cumanan, A. G. Burr, H. Q. Ngo, E. G. Larsson, and P. Xiao, "Energy efficiency of the cell-free massive MIMO uplink with optimal uniform quantization," *IEEE Trans. Green Commun. Netw.*, vol. 3, no. 4, pp. 971–987, Dec. 2019.

[26] T. H. Nguyen, T. K. Nguyen, H. D. Han, and V. D. Nguyen, "Optimal power control and load balancing for uplink cell-free multi-user massive MIMO," *IEEE Access*, vol. 6, pp. 14462–14473, 2018.

[27] P. Liu, K. Luo, D. Chen, and T. Jiang, "Spectral efficiency analysis of cell-free massive MIMO systems with zero-forcing detector," *IEEE Trans. Wireless Commun.*, vol. 19, no. 2, pp. 795–807, Feb. 2020.

[28] I. Atzeni, B. Gouda, and A. Tölli, "Distributed joint receiver design for uplink cell-free massive MIMO," in *Proc. IEEE Int. Conf. Commun. Workshops (ICC Workshops)*, Jun. 2020, pp. 1–6.

[29] J. Kaleva, A. Tölli, M. Juntti, R. A. Berry, and M. L. Honig, "Decentralized joint precoding with pilot-aided beamformer estimation," *IEEE Trans. Signal Process.*, vol. 66, no. 9, pp. 2330–2341, May 2018.

[30] R. Wang, Y. Jiang, and W. Zhang, "A distributed MIMO relay scheme inspired by backpropagation algorithm," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2021, pp. 1–6.

[31] R. Wang, Y. Jiang, and W. Zhang, "Distributed learning for MIMO relay networks," *IEEE J. Sel. Topics Signal Process.*, vol. 16, no. 3, pp. 343–357, Apr. 2022.

[32] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016. [Online]. Available: http://www.deeplearningbook.org

[33] D. Guo, S. Shamai, and S. Verdu, "Mutual information and minimum mean-square error in Gaussian channels," *IEEE Trans. Inf. Theory*, vol. 51, no. 4, pp. 1261–1282, Apr. 2005.

[34] S. W. Roberts, "Control chart tests based on geometric moving averages," *Technometrics*, vol. 42, no. 1, pp. 97–101, Feb. 2000.

[35] H. J. Stuart, "The exponentially weighted moving average," *J. Quality Technol.*, vol. 18, no. 4, pp. 203–210, 1986.

[36] S. V. Crowder, "Design of exponentially weighted moving average schemes," *J. Quality Technol.*, vol. 21, no. 3, pp. 155–162, Jul. 1989.

[37] L. A. Jones, C. W. Champ, and S. E. Rigdon, "The performance of exponentially weighted moving average charts with estimated parameters," *Technometrics*, vol. 43, no. 2, pp. 156–167, May 2001.

[38] J. M. Lucas and M. S. Saccucci, "Exponentially weighted moving average control schemes: Properties and enhancements," *Technometrics*, vol. 32, no. 1, pp. 1–12, Feb. 1990.

[39] S. Sukparungsee, Y. Areepong, and R. Taboran, "Exponentially weighted moving average—Moving average charts for monitoring the process mean," *PLoS ONE*, vol. 15, no. 2, Feb. 2020, Art. no. e0228208.

[40] J. Yu, S. B. Kim, J. Bai, and S. W. Han, "Comparative study on exponentially weighted moving average approaches for the self-starting forecasting," *Appl. Sci.*, vol. 10, no. 20, p. 7351, Oct. 2020.

[41] G. Scutari, F. Facchinei, P. Song, D. P. Palomar, and J. Pang, "Decomposition by partial linearization: Parallel optimization of multi-agent systems," *IEEE Trans. Signal Process.*, vol. 62, no. 3, pp. 641–656, Feb. 2014.

[42] F. Sohrabi, T. Jiang, and W. Yu, "Learning progressive distributed compression strategies from local channel state information," *IEEE J. Sel. Topics Signal Process.*, vol. 16, no. 3, pp. 573–584, Apr. 2022.

[43] A. Hjørungnes, *Complex-Valued Matrix Derivatives: With Applications in Signal Processing and Communications*. Cambridge, U.K.: Cambridge Univ. Press, 2011.

Rui Wang (Student Member, IEEE) received the B.S. degree in communication engineering from Northeastern University, China, in 2018. She is currently pursuing the Ph.D. degree with the Department of Communication Science and Engineering, Fudan University. Her research interests include MIMO wireless communication, relay networks, cell-free networks, and signal processing for wireless communication.

Weijie Dai received the B.S. degree in electronic and information engineering from the University of Electronic Science and Technology of China (UESTC), Chengdu, China. He is currently pursuing the master's degree with the Department of Communication Science and Engineering, Fudan University. His research interests include MIMO wireless communication, cell-free networks, and related signal processing technology.

Yi Jiang (Member, IEEE) received the B.S. degree in electrical engineering and information science from the University of Science and Technology of China (USTC), Hefei, China, in 2001, and the M.S. and Ph.D. degrees in electrical engineering from the University of Florida, Gainesville, FL, USA, in 2003 and 2005, respectively. In 2005, he was a Research Consultant with Information Science Technologies Inc., Fort Collins, CO, USA. From 2005 to 2007, he was a Post-Doctoral Researcher with the University of Colorado Boulder, Boulder, CO. He moved to San Diego, CA, USA, in 2007, and worked for multiple companies, including NextWave Wireless, from May 2007 to July 2008; Qualcomm Corporate Research and Development, from August 2008 to May 2012; IAA Incorporated, from June 2012 to January 2013; and Silvus Technologies, from February 2013 to July 2016. He was an Adjunct Researcher with the Electrical Engineering Department, University of California at Los Angeles, Los Angeles, CA, from October 2014 to July 2016. He joined Fudan University, China, in August 2016, where he is currently a Professor of the Department of Communication Science and Engineering. His research interests include array signal processing and mobile ad hoc networks.