# Decentralized Design of Fast Iterative Receivers for Massive MIMO with Spatial Non-Stationarities

Croisfelt, Victor; Abrao, Taufik; Amiri, Abolfazl; De Carvalho, Elisabeth; Popovski, Petar

*Published in:*
55th Asilomar Conference on Signals, Systems and Computers, ACSSC 2021

*DOI (link to publication from Publisher):*
10.1109/IEEECONF53345.2021.9723315

*Publication date:*
2021

*Document Version*
Accepted author manuscript, peer reviewed version

# Decentralized Design of Fast Iterative Receivers for Massive MIMO with Spatial Non-Stationarities

Victor Croisfelt, Taufik Abrão, Abolfazl Amiri, Elisabeth de Carvalho, and Petar Popovski

***Abstract*—Despite the extensive use of a centralized approach to design receivers for massive multiple-input multiple-output (M-MIMO) base stations, their actual implementation is a major challenge due to several bottlenecks imposed by the large number of antennas. One way to deal with this problem is by fully decentralizing the classical zero-forcing (ZF) receiver across multiple processing nodes based on the gradient descent (GD) method. In this paper, we first explicitly relate this decentralized receiver to a distributed version of the Kaczmarz algorithm and to the use of the successive interference cancellation (SIC) philosophy. In addition, we propose three methods to further accelerate the initial convergence of this decentralized receiver by exploring the connection with the Kaczmarz algorithm: 1) a new Bayesian distributed receiver, which iteratively estimates and eliminates noise; 2) a more practical method for choosing a relaxation parameter on an iteration basis; and 3) extension of it to tree-based architectures. The discussion also considers spatial non-stationarities that arise when the antenna arrays are extremely large (XL-MIMO). With our methods, we are able to improve the performance of the decentralized GD receiver for both spatially stationary and non-stationary channels, but mainly the non-stationary performance can still be improved. Future research directions are provided with the aim of further improving the effectiveness of receivers based on the so-called principle of successive residual cancellation (SRC).**

***Index Terms*—Massive MIMO (M-MIMO), extra-large scale MIMO (XL-MIMO), fully decentralized receivers, daisy-chain architecture, Kaczmarz algorithm, successive residual cancellation (SRC), Bayesian receiver, relaxation method.**

## I. Introduction

A key problem in the hardware implementation of massive multiple-input multiple-output (M-MIMO) base stations (BSs) is the centralization of baseband processing tasks on a *centralized processing unit* (CPU) [1], [2]. Such an approach is limited by a tremendous amount of information that must be transferred internally from the modules of the large number of antennas to the CPU. In addition, other physical hardware constraints also make centralization difficult, such as power management and limited pins on integrated circuits. A natural solution to this problem is to decentralize the processing tasks [2], implementing more nodes called *remote processing units* (RPUs) to process received signals as close as possible to the antennas. While this reduces the amount of information exchange, other challenges emerge. In fact, most classical MIMO receivers are designed assuming that all channel state information (CSI) is gathered on the CPU, such as in the zero-forcing (ZF) and regularized zero-forcing (RZF) schemes [3]–[5]. Due to the recent interest in producing systems that are scalable with an increasing number of antennas, the receiver design must be rethought considering that the CSI is only available locally on each RPU. In this work, we consider the problem of designing *fully* decentralized receivers in which the local CSI known in each RPU cannot be shared.

As the antenna arrays become extremely large (XL), a new channel effect also challenges the functionality of the receivers. The appearance of the so-called *spatially non-stationary channels* consists of the fact that only part of the antenna array is containing most of the users' equipment (UEs) energy due to the augmented array size [6]. In order to allow even more scalability of the spatial resolution of XL-MIMO systems, we are also interested in designing receivers which are aware of such spatial non-stationarities.

Many recent publications address the idea of decentralizing the baseband processing in M-MIMO BSs [7]. A partially decentralized receiver was considered in [8]. However, a clear drawback of a partially decentralized method is that it requires: a) a lot of information exchange between RPUs and b) orchestration of the process by the CPU. Furthermore, the method from [8] involves inverting the channel Gramian matrix in the CPU, with a third-order complexity increase in the number of UEs. These disadvantages motivated the search for fully decentralized algorithms. The method proposed in [9] is fully decentralized, but it suffers excessively from the exchange of consensus information between RPUs. The authors of [1] proposed an iterative approach to fully decentralize the classical ZFs design considering the gradient descent (GD) method. Their scheme is based on a hardware architecture with daisy-chained RPUs. Due to the sequential property of the architecture, the *GD receiver* from [1] must perform several iterations over the RPUs to obtain a performance close to that of the centralized ZF. Thus, it may have a considerable high processing delay.

In the context of XL-MIMO systems, the authors of [10] combine the variational message-passing (MP) and the belief-propagation (BP) frameworks to propose a fully decentralized scheme called the *MP-BF receiver*. Their method takes into account the spatial non-stationarities. Despite the good results, the method is more complex than the *GD receiver* from [1]. We believe that methods with such good performance and cheaper than the MP-BF receiver can be found for XL-MIMO.

**Contributions.** In this paper, we start by showing two alternative approaches to obtain the GD receiver from [1]. These methods are based on the decentralized version of the Kaczmarz algorithm that was recently proposed in [11] and the exploitation of the philosophy behind successive interference cancellation (SIC) [3], [4]. Because of the intensive exploitation of Kaczmarz-based acceleration techniques,we will refer

to the GD receiver of [1] as the *standard distributed Kaczmarz (SDK) receiver*, which is based on what is called the principle of *successive residual cancellation* (SRC). We then make use of the connection with the Kaczmarz algorithm to present three ways to further improve the initial convergence of the SDK receiver. First, we propose a new receiver that decentralizes the classical ZF receiver incorporating the Bayesian perspective. This scheme has the advantage of estimating and eliminating noise as iterations along the RPUs are performed. Second, we propose a more practical method to choose the relaxation parameter of the SDK receiver than the one proposed in [1]. Our method is based on a heuristic principle that tries to overcome the fact that the optimization of the relaxation parameter is non-trivial. Yet, it is more flexible and performs better than [1]. Third, we discuss how the SDK receiver can be applied over tree-based basedband processing architectures. The motivation is that tree-based architectures can exploit parallel computation and, as consequence, further reduce the delay of the daisy-chain architecture.

**Notations.** Non-boldface letters denote scalars. Upper and lower case boldface are used for matrices and vectors. Discrete and continuous sets are given by calligraphic $\mathcal{X}$ and blackboard bold $\mathbb{X}$. The $n$-th entry of $\mathbf{x}$ is $x_n$. Matrix concatenation is $[\mathbf{X}, \mathbf{Y}]$ for horizontal and $[\mathbf{X}; \mathbf{Y}]$ for vertical. Transpose and Hermitian transpose are $(\cdot)^{\mathsf{T}}$ and $(\cdot)^{\mathsf{H}}$. The $l_2$-norm is $\|\cdot\|_2^2$ and $\langle \cdot, \cdot \rangle$ denotes the inner product between any two vectors. The $M \times N$ matrix of zeros is $\mathbf{0}_{M\times N}$, whereas the identity matrix of size $N$ is $\mathbf{I}_N$. The standard basis vector $\mathbf{e}_n \in \mathbb{C}^{N\times 1}$ is the $n$-th column of $\mathbf{I}_N$. Circularly symmetric complex-Gaussian distribution is $\mathcal{CN}(\cdot, \cdot)$. The expected value is $\mathbb{E}[\cdot]$. The product of matrices $\prod_{n=1}^{N} \mathbf{X}_n$ is set to the identity matrix if the index is out of the bound.

## II. System Model

The system model concentrates on the uplink (UL) phase of an M-MIMO system wherein a BS equipped with $M$ antennas simultaneously serves $K$ single-antenna UEs, respectively indexed by the sets $\mathcal{M} = \{1, 2, \ldots, M\}$ and $\mathcal{K} = \{1, 2, \ldots, K\}$. The BS received signal $\mathbf{y} \in \mathbb{C}^{M\times 1}$ is

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{n} = \sum_{m=1}^{M} (\mathbf{h}_m^{\mathsf{T}}\mathbf{x})\mathbf{e}_m + \mathbf{n}, \tag{1}$$

where $\mathbf{H} \in \mathbb{C}^{M\times K} = [\mathbf{h}_1^{\mathsf{T}}, \mathbf{h}_2^{\mathsf{T}}, \ldots, \mathbf{h}_M^{\mathsf{T}}]^{\mathsf{T}}$ is the channel matrix, $\mathbf{x} \in \mathbb{C}^{K\times 1} \sim \mathcal{CN}(\mathbf{0}_{K\times 1}, p\mathbf{I}_K)$ is the data signal vector with UL transmit power $p$, $\mathbf{n} \in \mathbb{C}^{M\times 1} \sim \mathcal{CN}(\mathbf{0}_{M\times 1}, \sigma^2\mathbf{I}_M)$ is the receiver noise vector with noise power $\sigma^2$.

### A. Channel Model

We assume the block-fading channel model in which non-line-of-sight (NLoS) channels are considered to be frequency-flat and time-invariant [5]. The channel vector $\mathbf{h}_m \in \mathbb{C}^{K\times 1} \sim \mathcal{CN}(\mathbf{0}_{K\times 1}, \mathbf{D}_m)$ encompasses the wireless links between the $K$ UEs and antenna $m \in \mathcal{M}$. The channel vectors of different antennas are considered spatially uncorrelated, *i.e.*, $\mathbb{E}[\mathbf{h}_i\mathbf{h}_j^{\mathsf{H}}] = \mathbf{0}_{K\times K}$, $\forall i \neq j$, $i, j \in \mathcal{M}$. The covariance matrix $\mathbf{D}_m \in \{0, 1\}^{K\times K}$ is a diagonal, indicator matrix that models spatial non-stationarities [6], [12]. An antenna $m$ is said to be *visible* by UE $k$ if $[\mathbf{D}_m]_{k,k} = 1$ and is *non-visible* otherwise. We assume that the diagonal of $\mathbf{D}_m$ has at least $D_0$ nonzero entries for all $m \in \mathcal{M}$. The physical meaning of this assumption is that each antenna $m$ is contributing to the communication of at least $D_0 > 0$ UEs. Furthermore, let $D_0 < D \leq K$ be the trace of $\mathbf{D}_m$ that is equal for all antenna $m \in \mathcal{M}$. We refer to $D$ as the *number of effective UEs* served by each antenna. The spatially stationary case is $\mathbf{D}_m = \mathbf{I}_K$, $\forall m \in \mathcal{M}$, which characterizes the M-MIMO regime of compact antenna arrays. The matrix $\mathbf{D}_m \neq \mathbf{I}_K$ then introduces sparsity into $\mathbf{h}_m$ in the spatially non-stationary case. We are interested to evaluate how this sparsity affects the performance of the decentralized receivers discussed here.

## III. Background

In this section, we first present the basic structure of the decentralized baseband processing architecture with daisy-chained RPUs and the basic principles behind the design of fully decentralized linear receivers. Next, we review the GD receiver from [1] presenting two different ways to obtain this receiver that improve the understanding of the method.

### A. Daisy-Chain Baseband Processing Architecture

We adopt the fully decentralized architecture for baseband processing at the BS considered in [1]. For simplicity, the hardware architecture comprises $M$ RPUs, each of which manages the processing tasks concerning an antenna $m \in \mathcal{M}$.[1] The $m$-th RPU is serially connected to the $(m+1)$-th RPU, which gives rise to a daisy-chain connection layout. Only RPU 1 has a dedicated physical link to the CPU. Figure 1 gives a graphical representation of the architecture. The undirected graph consists of $M$ nodes representing the $M$ RPUs and antennas. Henceforth, we will use the words RPUs and nodes interchangeably. The flat-tree is rooted at node 1. The local knowledge available at the $m$-th node includes perfect CSI and the signal received by the $m$-th antenna, which are embraced by the tuple $(\mathbf{h}_m, y_m)$ in Fig. 1.

### B. Fully Distributed Linear Receivers

The design of a complete receiver comprehends two phases of signal processing that result respectively in a soft-estimate and a hard-estimate of $\mathbf{x}$. The soft-estimate $\hat{\mathbf{x}} \in \mathbb{C}^{K\times 1}$ can be obtained as [4], [5]:

$$\hat{\mathbf{x}} = \mathbf{V}^{\mathsf{H}}\mathbf{y} = \sum_{m=1}^{M} \mathbf{v}_m^* y_m = \sum_{m=1}^{M} \hat{\mathbf{x}}_m, \tag{2}$$

where $\mathbf{V} \in \mathbb{C}^{M\times K} = [\mathbf{v}_1^{\mathsf{T}}, \mathbf{v}_2^{\mathsf{T}}, \ldots, \mathbf{v}_M^{\mathsf{T}}]^{\mathsf{T}}$ is the linear receive combining matrix. The objective of the $m$-th RPU is to obtain a reliable local soft-estimate $\hat{\mathbf{x}}_m$ based on the observation $y_m$ and knowing the local CSI $\mathbf{h}_m$. To do so, the two constraints imposed by the decentralized architecture in Fig.

[1] A more practical assumption would be to group the processing of some antennas in an RPU, since there are some common tasks on different antennas.

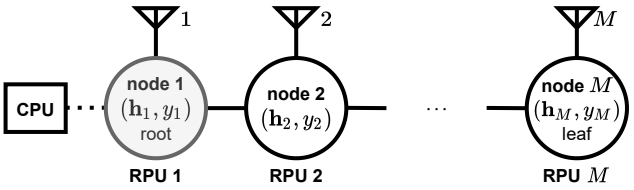



Fig. 1. Decentralized baseband processing architecture at the BS following a daisy-chain arrangement.

1 need to be considered [1]: (C.1) *Scalability.* Information exchange between RPUs must scale with $K$ instead of $M$. (C.2) *Autonomy.* Only the $m$-th RPU should be responsible for extracting relevant information from the local information $(\mathbf{h}_m, y_m)$. After the RPUs converge to a common, final soft-estimate $\hat{\mathbf{x}}$, the CPU receives and demodulates $\hat{\mathbf{x}}$ to produce the hard-estimate. Given the above constraints, we will focus on obtaining a fully decentralized method to obtain the soft-estimate $\hat{\mathbf{x}}$, which encompasses the problems of i) how to combine the information received from $M$ different antennas and 2) how to separate the signals from $K$ different UEs; both in a distributed way across the RPUs.

### C. Standard Distributed Kaczmarz (SDK)

The ZF combiner can be obtained by solving the following unconstrained least-squares (LS) problem [3]:

$$\arg\min_{\mathbf{x}\in\mathbb{C}^{K\times 1}} \|\mathbf{y} - \mathbf{H}\mathbf{x}\|_2^2. \tag{3}$$

We look at (3) as the overdetermined system of linear equations (SLE) $\mathbf{y} = \mathbf{H}\mathbf{x}$, which is inconsistent due to noise disturbance in $\mathbf{y}$. Furthermore, $\mathbf{H}$ is full-rank in the spatially stationary case and can be rank-deficient[2] in the spatially non-stationary case. In our setting, due to (C.1) and (C.2), the $m$-th node only knows the $m$-th equation $y_m = \mathbf{h}_m^{\mathsf{T}}\mathbf{x}$. A popular iterative algorithm that can solve $\mathbf{y} = \mathbf{H}\mathbf{x}$ is the Kaczmarz algorithm [13]. Indeed, the modified Kaczmarz method introduced in [11] considers exactly the case of interest where the equations are indexed by nodes of a tree. The method is divided into the *dispersion* and *pooling* stages. Its application gives rise to the *SDK receiver*.

**Dispersion stage.** The root node starts with an initial guess $\hat{\mathbf{x}}_0^{(t)}$, where $t \in \{1, 2, \ldots, T\}$ indexes *cycles* over the tree. This initial guess is improved by iterative steps that are defined by node $m \in \mathcal{M}$ receiving from its immediate predecessor $m-1$ an input estimate $\hat{\mathbf{x}}_{m-1}^{(t)}$. Node $m$ then applies the Kaczmarz update [13] to generate the following new estimate:

$$\begin{aligned} r_m^{(t)} &= y_m - \mathbf{h}_m^{\mathsf{T}}\hat{\mathbf{x}}_{m-1}^{(t)}, \\ \hat{\mathbf{x}}_m^{(t)} &= \hat{\mathbf{x}}_{m-1}^{(t)} + \lambda_m \mathbf{h}_m^{*} r_m^{(t)}, \end{aligned} \tag{4}$$

where $r_m^{(t)}$ is the *residual* and $\lambda_m = \lambda\|\mathbf{h}_m\|_2^{-2}$ with $\lambda$ being the *relaxation parameter*. The relaxation parameter can help in handling the inconsistency of the SLE [14] and is discussed in depth in Subsection IV-B. Fig. 2 shows a block diagram illustrating the above iterative step. The process continue recursively with node $m$ transferring $\hat{\mathbf{x}}_m^{(t)}$ to its immediate successor $m+1$. The dispersion stage finishes when the leaf node obtains its estimate $\hat{\mathbf{x}}_M^{(t)}$.

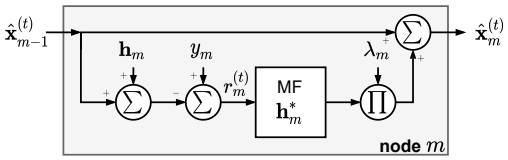



Fig. 2. Kaczmarz iterative step in Eq. (4) of the SDK receiver at node $m \in \mathcal{M}$. MF stands for matched filter.

**Pooling stage.** Since we are considering a flat-tree in Fig. 1, the pooling stage simply proceeds with the leaf node back-propagating its estimate $\hat{\mathbf{x}}_M^{(t)}$ through the tree without updating and until it reaches the root node. A new cycle $t+1$ over the tree can begin by defining $\hat{\mathbf{x}}_0^{(t+1)} = \hat{\mathbf{x}}_M^{(t)}$ at the root node; otherwise, $\hat{\mathbf{x}} = \hat{\mathbf{x}}_M^{(t)}$ is transferred from the root node to the CPU for demodulation. In [1], a dedicated physical connection between the root and leaf nodes was considered to reduce communication delay. However, we chose to consider the most general case of backpropagation because this naturally allows the generalization of the algorithm for more intricate tree's topologies, as we will discuss in Subsection IV-D. Fig. 3 offers an overview of the dispersion and pooling stages.

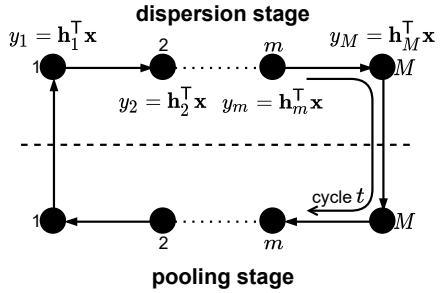



Fig. 3. Illustration of the dispersion and pooling stages. The pooling stage is depicted by mirroring the nodes. The algebraic equations of $\mathbf{y} = \mathbf{H}\mathbf{x}$ are indexed by the nodes of the flat-tree in Fig. 1. A cycle $t \in \{1, 2, \ldots, T\}$ can be seen as the loop defined by the sequential realization of dispersion and pooling stages.

*Remark 1 (Convergence of the SDK receiver).* For an inconsistent SLE, the SDK receiver converges to a weighted LS solution that depends on $\lambda$ in the case of a full-rank $\mathbf{H}$ (stationary case) and to a weighted minimum norm LS solution in the case of a rank-deficient $\mathbf{H}$ (non-stationary case) [11].

*Remark 2 (Complexity of the SDK receiver).* Each node $m \in \mathcal{M}$ only needs to store a $K$-dimensional complex vector $\hat{\mathbf{x}}_m^{(t)}$. For the $m$-th node, the Kaczmarz iterative step in (4) requires $(12K+2)T$ floating-point per second (FLOPs) measured in terms of real operations, where $T$ is the total number of cycles. The overall information exchange between any two nodes scales with $4KT$ FLOPs due to the pooling stage with backpropagation.

We note that the iterative step of the SDK receiver in Eq. (4) is essentially equivalent to the one introduced in Eq. (11) of [1], where the authors approximated the GD method to

[2]The sparsity in $\mathbf{H}$ can introduce the possibility that at least two channel vectors yields a linearly dependent set [12].

cover constraints (C.1) and (C.2). However, the connection to the distributed Kaczmarz algorithm is beneficial because of the convergence guarantees proved in [11]. Moreover, we will borrow ideas from the vast body of literature that has studied ways to improve the convergence of Kaczmarz-based approaches [11], [14]–[16] in Section IV.

The authors of [1] exploited the intrinsic recursion of the Kaczmarz update step in (4) to calculate the combining vector $\mathbf{v}_m$. For $T = 1$, the leaf estimate $\hat{\mathbf{x}}_M^{(1)}$ can be written as:

$$\hat{\mathbf{x}}_M^{(1)} = \prod_{m=1}^{M} (\mathbf{I}_K - \lambda_m \mathbf{h}_m \mathbf{h}_m^{\mathsf{H}}) \hat{\mathbf{x}}_0^{(1)} + \sum_{m=1}^{M} \prod_{i=m+1}^{M} (\mathbf{I}_K - \lambda_i \mathbf{h}_i \mathbf{h}_i^{\mathsf{H}}) \lambda_m \mathbf{h}_m^* y_m, \quad (5)$$

which encounters structural similarity with Eq. (2); assuming that the initial guess is $\hat{\mathbf{x}}_0^{(1)} = \mathbf{0}_{K\times 1}$, $\mathbf{v}_m = \prod_{i=m+1}^{M} (\mathbf{I}_K - \lambda_i \mathbf{h}_i \mathbf{h}_i^{\mathsf{H}}) \lambda_m \mathbf{h}_m$. Expressions like the above will be used to draw some conclusions throughout the paper and make it explicit the straightforwardly generalization of the receivers to obtain $\mathbf{v}_m$ in a distributed manner.

### D. Successive Residual Cancellation (SRC)

Another classical approach to obtain the soft-estimate $\hat{\mathbf{x}}$ is to apply the heuristic idea of the linear SIC receiver. The principle of SIC can be seen as a two step operation [3], [4]: (S.1) only the soft-estimate of a UE is estimated at a time and (S.2) its contribution is removed from the received signal. We now want to apply these steps to combine the local soft-estimates $\hat{\mathbf{x}}_m$ between the different nodes in the decentralized architecture of Fig. 1. To this end, an observation is in order. In the conventional multiuser detection problem [3], [4], the SIC suppresses inter-user interference. In the new decentralized processing problem, each node $m \in \mathcal{M}$ only knows $y_m$ and $\mathbf{h}_m$ and gets from these a local soft-estimate $\hat{\mathbf{x}}_m$, which is already in $K$-dimension. The nodes have then to agree on a final soft-estimate $\hat{\mathbf{x}}$ of $\mathbf{x}$. Therefore, the inter-user interference notion is translated into the residual differences between $\hat{\mathbf{x}}_m$'s from the different nodes.

We now apply the SIC steps over the fully decentralized architecture by changing the perspective from UEs to RPUs, giving rise to the following *successive residual cancellation* (SRC) receiver:[3]

$$\hat{\mathbf{x}}_m = \mathbf{w}_m^* y_m^{(m)} + \sum_{i=1}^{m-1} \hat{\mathbf{x}}_i, \qquad \text{from (S.1)},$$
$$\mathbf{y}^{(m)} = \mathbf{y}^{(m-1)} - (\mathbf{w}_m^{\mathsf{T}} \hat{\mathbf{x}}_{m-1}) \mathbf{e}_m,\ \mathbf{y}^{(1)} = \mathbf{y}, \qquad \text{from (S.2)}, \quad (6)$$

where $\mathbf{w}_m \in \mathbb{C}^{K\times 1}$ stands for a linear combiner and $\mathbf{y}^{(m)}$ denotes the residual, that is, the received signal $\mathbf{y}$ less the estimates $\hat{\mathbf{x}}_i$ from the previous $m-1$ iterates. From a straightforward matrix-algebra analysis over the recursion of $\mathbf{y}^{(m)}$, we have the following soft-decision estimate after $M$ SRC iterations:

$$\hat{\mathbf{x}}_M = \sum_{m=1}^{M} \prod_{i=m+1}^{M} (\mathbf{I}_K - \mathbf{w}_i \mathbf{w}_i^{\mathsf{H}}) \mathbf{w}_m^* y_m, \quad (7)$$

which is mathematically equivalent to the expression obtained for the SDK receiver in (5) when the normalized matched-filter (MF) is chosen $\mathbf{w}_m = \|\mathbf{h}_m\|_2^{-2} \mathbf{h}_m$, zero vector is the initial guess, and $\lambda = 1$. In fact, the SDK receiver can be seen as a generalization of the SRC due to the relaxation parameter $\lambda$. In [17], a similar connection was found between the Kaczmarz algorithm and the SIC receiver in the usual centralized setting.

## IV. Accelerating Initial Convergence

In this section, motivated by the connections made with the Kaczmarz algorithm and the SRC principle, we discuss different ways to improve the initial convergence of the SDK receiver. The interest in improving its initial behavior lies in the fact that we naturally want the algorithm to end as soon as possible, thus reducing the complexity and latency to obtain the soft-estimate $\hat{\mathbf{x}}$.

### A. Bayesian Distributed Kaczmarz (BDK)

One way to handle the inconsistency of $\mathbf{y} = \mathbf{H}\mathbf{x}$ is to adopt the Bayesian perspective for finding $\mathbf{x}$. Given a realization of the channel matrix $\mathbf{H}$, the idea is to incorporate prior[4] knowledge of the data signal $\mathbf{x}$ and receiver noise $\mathbf{n}$ from (1) to the estimation problem. We make the following assumptions: (A.1) either $\mathbf{x}$ and $\mathbf{n}$ follows complex circularly-symmetric central Gaussian distributions, which is justified by a worst-case scenario from the point of view of communication [5]; (A.2) the entries of either $\mathbf{x}$ and $\mathbf{n}$ are uncorrelated and have the same variance (UL transmit power $p$ and noise power $\sigma^2$, respectively), which is justified respectively by: each UE $k \in \mathcal{K}$ generates its data independently; each antenna $m \in \mathcal{M}$ sensor experiences noise independently. This gives rise to the following $l_2$-constrained LS problem that maximizes the *a posteriori* probability [15]:

$$\underset{\mathbf{x}\in\mathbb{C}^{K\times 1}}{\arg\min} \|\mathbf{y} - \mathbf{H}\mathbf{x}\|_2^2 + \xi \|\mathbf{x}\|_2^2, \quad (8)$$

where $\xi = \sigma^2/p$ is the inverse of the pre-processing signal-to-noise (SNR) ratio. Some readers familiar with the regularization method may recognize that the problem above can also be interpreted from the point of view of the Tikhonov regularization [18] and its solution is related to the RZF combiner [3], [5]. What we essentially have now from (8) is the following underdetermined SLE $\mathbf{y} = \mathbf{H}\mathbf{x} + \sqrt{\xi}\mathbf{u} = \mathbf{B}\mathbf{z}$, where $\mathbf{B} \in \mathbb{C}^{M\times(K+M)} = [\mathbf{H}, \sqrt{\xi}\mathbf{I}_M]$ and $\mathbf{z} \in \mathbb{C}^{(K+M)\times 1} = [\mathbf{x} \in \mathbb{C}^{K\times 1}; \mathbf{u} \in \mathbb{C}^{M\times 1}]$ [15]. Different from the previous SLE, $\mathbf{y} = \mathbf{B}\mathbf{z}$ is consistent due the augmented range brought by $\mathbf{u}$.

[3] The SRC principle goes back to the idea of solving the SLE by operating on one equation at a time [13].

[4] In the event that other practical information is known about $\mathbf{x}$ and $\mathbf{n}$, other processes known as tricks can be inserted to further improve convergence performance of SDK and BDK. For example, in the case that $\mathbf{x}$ is drawn from a known modulation scheme, the values of $\mathbf{x}$ can be confined to a certain region. See [15] for an overview of these tricks.

Moreover, $\mathbf{B}$ is full-rank in the spatially stationary case and can be rank-deficient in the spatially non-stationary case. Below, we describe the distributed Kaczmarz algorithm [11] applied to solve $\mathbf{y} = \mathbf{Bz}$ and omit repetitive definitions. We refer to the procedure obtained as the *Bayesian distributed Kaczmarz* (BDK) receiver.

**Dispersion stage.** Initial guesses $\hat{\mathbf{x}}_0^{(t)}$ and $\hat{\mathbf{u}}_0^{(t)}$ are available at the root node. The Kaczmarz iterative step at node $m$ is now:

$$\begin{aligned} r_m^{(t)} &= y_m - \mathbf{h}_m^{\mathsf{T}}\hat{\mathbf{x}}_{m-1}^{(t)} - \sqrt{\xi}\mathbf{e}_m^{\mathsf{T}}\hat{\mathbf{u}}_{m-1}^{(t)}, \\ \hat{\mathbf{x}}_m^{(t)} &= \hat{\mathbf{x}}_{m-1}^{(t)} + \lambda_m^{\star}\mathbf{h}_m^{*} r_m^{(t)}, \\ \hat{\mathbf{u}}_m^{(t)} &= \hat{\mathbf{u}}_{m-1}^{(t)} + \lambda_m^{\star}\sqrt{\xi}\mathbf{e}_m^{*} r_m^{(t)}, \end{aligned} \tag{9}$$

where $\lambda_m^{\star} = \lambda^{\star}(\|\mathbf{h}_m\|_2^2 + \xi)^{-1}$ is the relaxation parameter regarding node $m \in \mathcal{M}$. Note that the star in $\lambda^{\star}$ is to differentiate from $\lambda$ of the SDK receiver. Since the problem being solved is now consistent, we will set $\lambda^{\star} = 1$.[5] Fig. 4 illustrates the new Kaczmarz iterative step. The dispersion stage ends when the leaf node gets $\hat{\mathbf{x}}_M^{(t)}$ and $\hat{\mathbf{u}}_M^{(t)}$.

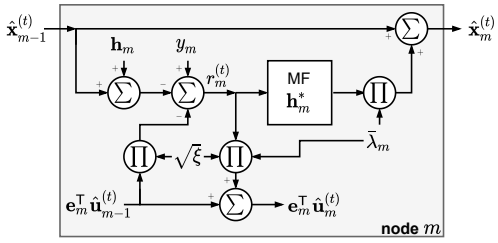



Fig. 4. Kaczmarz iterative step in Eq. (9) of the BDK receiver at node $m \in \mathcal{M}$.

**Pooling stage.** At first sight, the BDK receiver with iterative step defined by (9) seems to make little sense. The algorithm description suggests that $\hat{\mathbf{u}}_m^{(t)}$ should be exchanged between nodes, whose vector is in $\mathbb{C}^{M\times 1}$; therefore, violating (C.1). However, a careful look at the iterative step shows that the $m$-th component of $\hat{\mathbf{u}}_m^{(t)}$ is only altered by the $m$-th node because of $\mathbf{e}_m$. As a consequence, $\hat{\mathbf{u}}_M^{(t)}$ does not need to be backpropagated to the root node. To gain further intuition about this, note that $\sqrt{\xi}\mathbf{e}_m^{\mathsf{T}}\hat{\mathbf{u}}_{m-1}$ is trivially an estimate of $n_m$, which is the receiver noise relative to the $m$-th antenna in (1). Remember that $n_m$ is considered independent of the noise experienced by other antennas. Hence, the pooling stage is the same to that of the SDK algorithm, as shown is Fig. 3.

*Remark 3 (Convergence of the BDK receiver).* For a consistent SLE, the BDK receiver converges to the minimum norm LS solution in either stationary and non-stationary cases [11].

*Remark 4 (Complexity of the BDK receiver).* Each node $m \in \mathcal{M}$ requires fixed storage for a $K$-dimensional complex vector $\hat{\mathbf{x}}_m^{(t)}$ and a complex scalar representing the $m$-th entry of $\hat{\mathbf{u}}_m^{(t)}$. For the $m$-th node, the Kaczmarz iterative step in (9) requires $(12K+6)T$ FLOPs. The overall information exchange between any two nodes scales with $4KT$ FLOPs.

From Remarks 2 and 4, it is easy to see that the computational requirements of the SDK and BDK receivers are essentially the same thanks to the benign property involving the distributed knowledge of $\hat{\mathbf{u}}_M^{(t)}$ in the nodes. The remarks also comprise an upper bound of complexity, since they consider $\mathbf{h}_m$ dense. Further computational gains are obtained when considering $\mathbf{h}_m$ sparse [15].

*1) Understanding BDK receiver:* There are two key differences between the SDK and BDK receivers. The first is the incorporation of prior knowledge $\xi$ in $\lambda_m^{\star} = \lambda^{\star}(\|\mathbf{h}_m\|_2^2 + \xi)^{-1}$, $\forall m \in \mathcal{M}$, which helps to combat the noise propagation through the residuals. The second is a change on the way that the estimates are combined through the nodes due to the addition of the noise estimation. For intuition about this, we use part of the recursion of the Kaczmarz iterative step in (9) to generate the following expressions for $\hat{\mathbf{x}}_M^{(1)}$ and $\hat{\mathbf{u}}_M^{(1)}$:

$$\begin{aligned} \hat{\mathbf{x}}_M^{(1)} &= \prod_{m=1}^{M}(\mathbf{I}_K - \lambda_m^{\star}\mathbf{h}_m\mathbf{h}_m^{\mathsf{H}})\hat{\mathbf{x}}_0^{(1)} \\ &\quad + \sum_{m=1}^{M}\prod_{i=m+1}^{M}(\mathbf{I}_K - \lambda_i^{\star}\mathbf{h}_i\mathbf{h}_i^{\mathsf{H}})\lambda_m^{\star}\mathbf{h}_m^{*}(y_m - \sqrt{\xi}\mathbf{e}_m^{\mathsf{T}}\hat{\mathbf{u}}_{m-1}^{(t)}), \\ \hat{\mathbf{u}}_M^{(1)} &= \prod_{m=1}^{M}(\mathbf{I}_M - \lambda_m^{\star}\xi\mathbf{e}_m\mathbf{e}_m^{\mathsf{H}})\hat{\mathbf{u}}_0^{(1)} \\ &\quad + \sum_{m=1}^{M}\prod_{i=m+1}^{M}(\mathbf{I}_M - \lambda_i^{\star}\xi\mathbf{e}_i\mathbf{e}_i^{\mathsf{H}})\lambda_m^{\star}\sqrt{\xi}\mathbf{e}_m^{*}(y_m - \mathbf{h}_m^{\mathsf{T}}\hat{\mathbf{x}}_{m-1}^{(t)}). \end{aligned}$$

Comparing these with Eq. (5), it is possible to note that the estimates now rely on "effective" received signals. For example, to estimate a new $\hat{\mathbf{x}}_m^{(t)}$, the effective signals of the form $y_i - \sqrt{\xi}\mathbf{e}_i^{\mathsf{T}}\hat{\mathbf{u}}_{i-1}^{(t)}$ for $i \in \mathcal{M}, i \le m$ will be considered, which eliminates the current noise estimate $\sqrt{\xi}\mathbf{e}_i^{\mathsf{T}}\hat{\mathbf{u}}_{i-1}^{(t)}$ from the true received received signal $y_i$.

### *B. Choosing the Relaxation Parameter for the SDK Receiver*

The relaxation parameter can significantly improve the performance of the Kaczmarz algorithm in practice. However, it is very difficult to find a suitable relaxation parameter, since its optimization is non-trivial [15]. The authors in Eq. (22) of [1] proposed the choice $\lambda^{\dagger} = \frac{1}{2}\frac{K}{M}\log(4 \cdot M \cdot \mathrm{SNR})$ for the SDK receiver. Despite the good results, this value comes from the approximation of a closed-form expression of the signal-to-interference-plus-noise ratio (SINR) for $\hat{\mathbf{x}}_M^{(1)}$ expressed in Eq. (5). Therefore, this approach is very inflexible in the sense that we need to approximate the SINR for different cycles $t$ and new approximations are necessary for different channel models. In this part, we propose a more practical and effective way to define the relaxation parameter resulting in better, although sub-optimal yet, performance than the approach proposed in [1]. Our method is a heuristic based on the combination of the following three observations:[6]

(O.1) *Connection to GD.* In [18], a simple and reasonable mathematical framework to select the step-size of the GD

[5] Future work can discuss a more suitable choice.

[6] There are some approaches that choose the relaxation parameters based on the normalization of the eigenvalues of $\mathbf{H}^{\mathsf{H}}\mathbf{H}$ and improvement of its condition number [19]. However, due to the decentralized constraints (C.1) and (C.2), we do not have access to $\mathbf{H}^{\mathsf{H}}\mathbf{H}$ let alone its inverse.

method was given. The step-size of the GD mathematically resembles $\lambda$ of the Kaczmarz iterative step in (4).

(O.2) *Noise Aggregation.* In [20], it was shown that the propagation of the noise error embedded in the residual is proportional to $\sqrt{m}$, where $m \in \mathcal{M}$ represents the current Kaczmarz iterative step applied by node $m$. This intuitively makes sense in the *case of dense* $\mathbf{h}_m$, since $\hat{\mathbf{x}}_{m-1}^{(t)}$ suffers from the aggregation of $m-1$ different and independent noise sources of the type $n_i \sim \mathcal{CN}(0, \sigma^2)$, $i < m$, $\forall i \in \mathcal{M}$. This suggests the definition of a $\lambda$ that is a function of $m$ and decreases according to $\sqrt{m}$. Since there is less useful information in future iterations, we can expand this idea to also provide a $\lambda$ that decreases with the number of cycles in the form $\sqrt{t}$.

(O.3) *Underrelaxation.* It has been found in [14], [15] that the use of $0 < \lambda < 1$ improves the performance of Kaczmarz-based methods when solving inconsistent SLEs.

Our proposal is to chose $\lambda$ based on the mathematical framework provided by (O.1) and incorporate (O.2) and (O.3) to the solution. First, we obtain the following result.

**Theorem 1.** *Let* $\mathbf{x}$ *be the solution of* (3) *and* $\boldsymbol{\varrho}_m^{(t)} = \mathbf{h}_m^* r_m^{(t)}$. *The Kaczmarz iterative step in* (4) *can then be rewritten as*

$$\hat{\mathbf{x}}_m^{(t)} = \hat{\mathbf{x}}_{m-1}^{(t)} + \lambda_m \boldsymbol{\varrho}_m^{(t)}. \tag{10}$$

*If* $t = 1$ *and* $\hat{\mathbf{x}}_0^{(1)} = \mathbf{0}_{K\times 1}$, *the expected sum of the squared errors of the form* $\|\hat{\mathbf{x}}_m^{(1)} - \mathbf{x}\|_2^2$ *conditioned on the set* $\{\mathbf{h}_m\}$, $\forall m \in \mathcal{M}$ *that represents perfect CSI knowledge can be bounded by the following quantity:*

$$\mathbb{E}\left[\sum_{m=1}^{M} \Re(\langle \hat{\mathbf{x}}_{m-1}^{(1)} - \mathbf{x}, \boldsymbol{\varrho}_m^{(1)} \rangle) | \{\mathbf{h}_m\}\right] \geq -\frac{1}{2\lambda_1}\mathbb{E}[\|\mathbf{x}\|_2^2 | \{\mathbf{h}_m\}] - \sum_{m=1}^{M} \frac{\lambda_m}{2}\mathbb{E}[\|\boldsymbol{\varrho}_m^{(1)}\|_2^2 | \{\mathbf{h}_m\}], \tag{11}$$

*where expectation is taken with respect to* $\mathbf{x}$ *and* $\mathbf{n}$. *In particular, if we specify that* $\mathbb{E}[\|\mathbf{x}\|_2^2 | \{\mathbf{h}_m\}] \geq B^2$, $\mathbb{E}[\|\boldsymbol{\varrho}_m^{(1)}\|_2^2 | \{\mathbf{h}_m\}] \geq \rho^2$, *for every* $B, \rho > 0$ *for all* $m \in \mathcal{M}$, *and if we define* $\lambda_m = \sqrt{\frac{B^2}{\rho^2 m}}\|\mathbf{h}_m\|_2^{-2}$, *we have that*

$$\mathbb{E}\left[\sum_{m=1}^{M} \Re(\langle \hat{\mathbf{x}}_{m-1}^{(1)} - \mathbf{x}, \boldsymbol{\varrho}_m^{(1)} \rangle) | \{\mathbf{h}_m\}\right] \geq -\frac{\|\mathbf{h}_1\|_2^2 B\rho}{2} - \frac{B\rho}{2}\sum_{m=1}^{M} \frac{\|\mathbf{h}_m\|_2^{-2}}{\sqrt{m}}. \tag{12}$$

*Given the distributions of* $\mathbf{x}$ *and* $\mathbf{n}$ *in* (1), *loose choices are* $B^2 = Kp$ *and* $\rho^2 = \sigma^2$ *for the SDK and dense* $\mathbf{h}_m$, $\forall m \in \mathcal{M}$.

*Proof.* The proof is given in Appendix A. □

Theorem 1 relates the choice of the relaxation parameter to a quantity of interest that measures how the expectation of the sum of squared errors of the type $\|\hat{\mathbf{x}}_m^{(1)} - \mathbf{x}\|_2^2$ progresses through the nodes. In (12), we already have considered part of (O.2) by incorporating the factor $\sqrt{m}$ into the parameter choice. Plugging (O.2), (O.3), and the result of the application of (O.1) in Theorem 1 in (12), we can heuristically set

$$\lambda_m = \underbrace{\min\left[\sqrt{\frac{K \cdot \mathrm{SNR}}{t \cdot m}}, 1\right]}_{=\lambda_{m,t}} \|\mathbf{h}_m\|_2^{-2}, \tag{13}$$

where $\mathrm{SNR} = p/\sigma^2$. The proposed relaxation parameter $\lambda_{m,t}$ is a function of, according to (O.2), $m \in \mathcal{M}$ and $t \in \{1, 2, \ldots, T\}$. The minimum function ensures that under-relaxation is applied in each node $m$ due to (O.3).

### C. Ordering and Randomization

In the context of SIC receivers, it is well known that the order the UEs are detected can affect the overall reliability of the successive detector due to error propagation [3], [4]. A good strategy is then to detect the UEs in the order of decreasing post-processing SNRs. In the decentralized setting, this observation also applies for the SRC-based receivers: the order in which local estimates $\hat{\mathbf{x}}_m$ are obtained at each node $m \in \mathcal{M}$ can increase the risk of error propagation. In the context of the Kaczmarz algorithm, this problem is also known for the fact that an efficient order of the equations of the SLE being solved can significantly improve the initial behavior of the algorithm in practice [15]. Since it is often difficult to obtain the optimum order, a suboptimal approach consists in randomizing the choice of the equations based on their energies [16]. In our setting, however, the nodes are physically connected sequentially, which prevents us from ordering how the local estimates are obtained without adding a moderate level of coordination by the CPU, while aggregating an overhead to the process. Motivated by [16], we have partially randomized the order by selecting the root node of the flat-tree in Fig. 1 based on probabilities proportional to $\|\mathbf{h}_m\|_2^2$. Although small improvements could be seen in high SNR, the results obtained were considered not very promising. Further study is needed to comprehend how the wired restriction affects the error propagation in SRC-based receivers. In the XL-MIMO case, this discussion is even more important, since the received energy is unequal across the nodes.

### D. Tree-Based Baseband Processing Architectures

To illustrate the generalization of the SDK receiver to tree-based processing architectures, we rely on the concept of sub-arrays introduced in [6]. A sub-array embraces part of the $M$ antenna elements that forms the whole antenna array deployed at the BS in which the channel can be considered spatially stationary. Fig. 5 illustrates a decentralized processing architecture with a sub-array $s_i$ being represented by a *node common bus*, where $i \in \{1, 2, \ldots, S\}$ and $S$ is the total number of subarrays. The sub-arrays are connected to the *sub-array common bus*, which can be seen as the root $r$ of a tree. Each leaf node $l_m$, $m \in \mathcal{M}$ represents one of the $M$ RPUs and has as information one equation $y_m = \mathbf{h}_m^{\mathsf{T}}\mathbf{x}$ from the SLE $\mathbf{y} = \mathbf{H}\mathbf{x}$. There are now weights of the form $g(n, v) = g(v, n)$, with $g(n, v) > 0$, associated to the links between $n$ and $v$.

Two changes are needed to define the SDK algorithm described previously for the structure in Fig. 5. First, in the dispersion stage, the initial guess $\hat{\mathbf{x}}_r^{(t)}$ is also spread over the structure with the detail that the same information is accessible in the buses. Second, in the pooling stage, the leaf nodes $l_m$'s also need to backpropagate their $\hat{\mathbf{x}}_m^{(t)}$'s to the root node $r$. But now, their local estimates are summed up together at the node common bus $s_i$ considering the weights $g(s_i, l_m)$ – the sum of the weights is equal to 1. This is followed by a weighted final sum between the sub-array nodes $s_i$ so as to send the final information to the root node $r$. The benefit of tree-based architectures is that they allow more processing to be performed in parallel than the full sequential mode provided by daisy-chaining RPUs in Fig. 1. This comes with the expectation of further reducing the delay to output the soft-estimate $\hat{\mathbf{x}}$. Future work can better characterize the operation and the trade-offs of more complex tree structures, comparing them with the daisy-chain of Fig. 1.

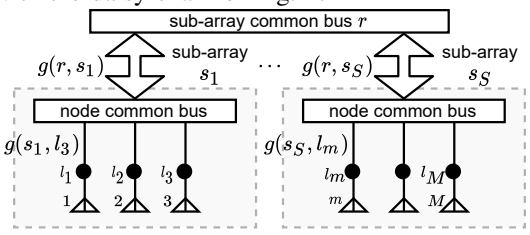


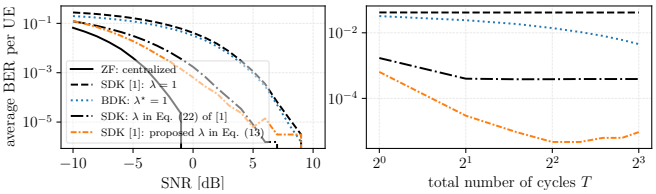

Fig. 5. Decentralized baseband processing architecture at the BS including the concept of sub-arrays.

## V. Numerical Results

We now present some simulation results showing the efficiency of the BDK receiver and our method to choose the relaxation parameter for the SDK receiver.[7] We use the bit-error rate (BER) to measure performance and drawn $\mathbf{x}$ according to a normalized 16-QAM (quadrature amplitude modulation) constellation with natural binary-coded ordering.

### A. Stationary Case

The stationary case comprises of all antennas serving all UEs, that is, $\mathbf{D}_m = \mathbf{I}_K$, $\forall m \in \mathcal{M}$. Fig. 6a shows the average BER per UE as a function of the SNR in dB. The BDK receiver performs slightly better than the SDK receiver, especially at low SNR where noise awareness is critical. However, the convergence rate is far from good and an appropriate choice of the relaxation parameter $\lambda$ seems to be very important to accelerate the convergence of the SDK receiver. Our heuristic method to select $\lambda$ defined in (13) is better than the one proposed in [1], just being worse for high SNR values. This is expected since in the case that the SNR is high, the expression in (13) typically evaluates to $\lambda = 1$ because of the minimum function.[8] However, the advantage of our method can be better seen in Fig. 6b that shows the average BER per UE as a function of the total number of cycles $T$. The figure uncovers the main drawback of the previous method of choosing $\lambda$ proposed in [1]: after $T = 1$ cycles the SINR expression changes, which in turn changes the choice of the optimal $\lambda^\dagger$ [1] value that maximizes the SINR. Hence, by increasing $T$, the constant relaxation parameter $\lambda^\dagger$ chosen before is no longer optimal, scaling future estimates erroneously. Meanwhile, our method provides improved performance until $T = 4$ iterations; for $T > 4$ the error propagation dominates, which is known as semi-convergence of the Kaczmarz algorithm [20].




(a) *vs* SNR in dB with $T = 1$. (b) *vs* $T$ with SNR = 0 dB.

Fig. 6. Average BER per UE with $M = 128$ antennas, $K = 16$ UEs.

### B. Non-Stationary Case

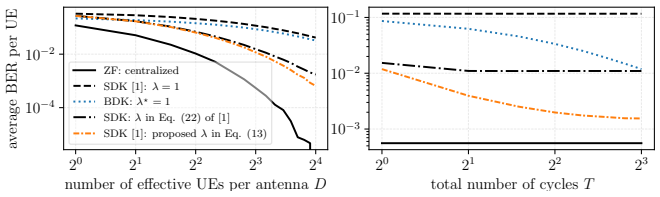
To generate the diagonals of the matrix $\mathbf{D}_m$, we adopted a simple model defined by the following step: for each antenna $m \in \mathcal{M}$, we successively generate a random sequence of length $K$ in which each random variable follows a Bernoulli distribution with equal probability until the sum of the random sequence is equal to the number of effective UEs $D$. The average BER per UE as a function of $D$ is shown in Fig. 7a. For very small values of $D$, we note that the SRC-based methods perform bad due to the strong sparsity condition imposed by the spatial non-stationarities that mathematically hinders the solution of the SLE. For larger values of $D$, the SDK receiver with a suitable choice of $\lambda$ starts to perform better in general and improves as $D$ comes closer to $K = 16$ UEs. This happens because the rank-deficiency of the channel matrix $\mathbf{H}$ is reduced by increasing $D$; when $D = K$, $\mathbf{H}$ is full-rank [12] and ZF performs better even with increasing interference. In Fig. 7b, we can note that our method of choosing $\lambda$ works better than the previous method under sparse $\mathbf{h}_m$. However, the curve converges to a floor because the method generates a scaled final estimate $\boldsymbol{\alpha}\hat{\mathbf{x}}$, where $\boldsymbol{\alpha} \in \mathbb{C}^{K\times 1} \neq [1, 1, \ldots, 1]$ embraces two facts introduced by the sparse $\mathbf{h}_m$: a) different from the stationary case, the $m$-th estimate $\hat{\mathbf{x}}_m$ does not necessarily suffer from $m - 1$ independent noise sources, going against (O.2); b) the Kaczmarz iterative step non-coherently combines the estimates from one iteration to another, converging to a *weighted* minimum norm LS solution (see Remark 1).

## VI. Conclusions

In this paper, we have shown two new approaches to derive the GD receiver proposed by [1] based on the distributed Kaczmarz algorithm and the principles of SRC. This part of the work was motivated by trying to better understand the GD receiver from [1] and to draw new connections to help improve its initial convergence. Based on the connections

[7] See: github.com/victorcroisfelt/dec-fast-iterative-receivers.

[8] One way to deal with this problem is to further limit the range $0 < \lambda < 1/2$, which was observed to work better in high SNR. However, convergence is naturally improved in the case of high SNR after a few cycles because the noise error does not affect the iterations much.




(a) *vs* $D$ with SNR = 0 dB, $T = 1$. (b) *vs* $T$ with SNR = 0 dB, $D = 8$.

Fig. 7. Average BER per UE with $M = 128$ antennas, $K = 16$ UEs.

made, we have proposed a new Bayesian distributed receiver and a new method to choose the relaxation parameter $\lambda$. Efforts were dedicated to improve the functionality of fully decentralized receivers under both spatially stationary and non-stationary channels. Future work may evaluate if our methods are applicable to cell-free M-MIMO networks, where RPUs would represent access points.

## References


[1] J. Rodríguez Sánchez, F. Rusek, O. Edfors, M. Sarajlić, and L. Liu, "Decentralized massive MIMO processing exploring daisy-chain architecture and recursive algorithms," *IEEE Transactions on Signal Processing*, vol. 68, pp. 687–700, 2020.

[2] C. Shepard, H. Yu, N. Anand, E. Li, T. Marzetta, R. Yang, and L. Zhong, "Argos: Practical many-antenna base stations," in *Proceedings of the 18th Annual International Conference on Mobile Computing and Networking*, ser. Mobicom '12. New York, NY, USA: Association for Computing Machinery, 2012, p. 53–64.

[3] S. Verdú, *Multiuser Detection*, 1st ed. USA: Cambridge University Press, 1998.

[4] T. Brown, E. De Carvalho, and P. Kyritsi, *Practical Guide to the MIMO Radio Channel with MATLAB Examples*, 1st ed. Wiley, 2012.

[5] E. Björnson, J. Hoydis, and L. Sanguinetti, "Massive MIMO networks: Spectral, energy, and hardware efficiency," *Foundations and Trends® in Signal Processing*, vol. 11, no. 3-4, pp. 154–655, 2017.

[6] E. D. Carvalho, A. Ali, A. Amiri, M. Angjelichinoski, and R. W. Heath, "Non-stationarities in extra-large-scale massive MIMO," *IEEE Wireless Communications*, vol. 27, no. 4, pp. 74–80, 2020.

[7] K. Li, J. McNaney, C. Tarver, O. Castañeda, C. Jeon, J. R. Cavallaro, and C. Studer, "Design trade-offs for decentralized baseband processing in massive MU-MIMO systems," in *2019 53rd Asilomar Conference on Signals, Systems, and Computers*, 2019, pp. 906–912.

[8] C. Jeon, K. Li, J. R. Cavallaro, and C. Studer, "Decentralized equalization with feedforward architectures for massive MU-MIMO," *IEEE Transactions on Signal Processing*, vol. 67, no. 17, pp. 4418–4432, 2019.

[9] K. Li, R. R. Sharan, Y. Chen, T. Goldstein, J. R. Cavallaro, and C. Studer, "Decentralized baseband processing for massive MU-MIMO systems," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 7, no. 4, pp. 491–507, 2017.

[10] A. Amiri, C. N. Manch'on, and E. de Carvalho, "Uncoordinated and decentralized processing in extra-large MIMO arrays," 2021.

[11] C. Hegde, F. Keinert, and E. S. Weber, "A Kaczmarz algorithm for solving tree based distributed systems of equations," 2019.

[12] A. Ali, E. D. Carvalho, and R. W. Heath, "Linear receivers in non-stationary massive MIMO channels with visibility regions," *IEEE Wireless Communications Letters*, vol. 8, no. 3, pp. 885–888, 2019.

[13] S. Kaczmarz, "Angenäherte auflösung von systemen linearer gleichungen," *Bulletin International de l'Académie Polonaise des Sciences et des Lettres. Classe des Sciences Mathématiques et Naturelles. Série A, Sciences Mathématiques*, vol. 35, pp. 355–357, 1937.

[14] Y. Censor, P. P. B. Eggermont, and D. Gordon, "Strong underrelaxation in Kaczmarz's method for inconsistent systems," *Numerische Mathematik*, vol. 41, no. 1, pp. 83–92, Feb 1983.

[15] G. T. Herman, *Fundamentals of Computerized Tomography*, ser. Advances in Pattern Recognition. London: Springer London, 2009.

[16] T. Strohmer and R. Vershynin, "Comments on the randomized Kaczmarz method," *Journal of Fourier Analysis and Applications*, vol. 15, no. 4, pp. 437–440, 2009.

[17] A. Bentrcia, "Kaczmarz successive interference cancellation: A matrix algebraic approach," *Digital Signal Processing*, vol. 89, pp. 60–69, 2019.

[18] S. Shalev-Shwartz and S. Ben-David, *Understanding Machine Learning - From Theory to Algorithms*. Cambridge University Press, 2014.

[19] M. Hanke and W. Niethammer, "On the acceleration of Kaczmarz's method for inconsistent linear systems," *Linear Algebra and its Applications*, vol. 130, pp. 83–98, 1990.

[20] T. Elfving, P. C. Hansen, and T. Nikazad, "Semi-convergence properties of Kaczmarz's method," *Inverse Problems*, vol. 30, no. 5, p. 055007, May 2014.


## Appendix
## Proof of Theorem 1

Let $\mathbf{x}$ be the solution of (3) and $\boldsymbol{\varrho}_m^{(t)} = \mathbf{h}_m^* r_m^{(t)}$. We then re-define the Kaczmarz iterative step in (4) as $\hat{\mathbf{x}}_m^{(t)} = \hat{\mathbf{x}}_{m-1}^{(t)} + \lambda_m \boldsymbol{\varrho}_m^{(t)}$. By defining the error at the Kaczmarz iterative step applied to the $m$-th node at the first cycle[9] ($t = 1$) results:

$$\begin{aligned}
\hat{\mathbf{x}}_m^{(1)} - \mathbf{x} &= (\hat{\mathbf{x}}_{m-1}^{(1)} - \mathbf{x}) + \lambda_m \boldsymbol{\varrho}_m^{(1)} \\
\|\hat{\mathbf{x}}_m^{(1)} - \mathbf{x}\|_2^2 &= \|\hat{\mathbf{x}}_{m-1}^{(1)} - \mathbf{x}\|_2^2 + 2\lambda_m \Re(\langle \hat{\mathbf{x}}_{m-1}^{(1)} - \mathbf{x}, \boldsymbol{\varrho}_m^{(1)} \rangle) \\
&\quad + \lambda_m^2 \|\boldsymbol{\varrho}_m^{(1)}\|_2^2 \\
\Re(\langle \hat{\mathbf{x}}_{m-1}^{(1)} - \mathbf{x}, \boldsymbol{\varrho}_m^{(1)} \rangle) &= \frac{1}{2\lambda_m}(\|\hat{\mathbf{x}}_m^{(1)} - \mathbf{x}\|_2^2 - \|\hat{\mathbf{x}}_{m-1}^{(1)} - \mathbf{x}\|_2^2) \\
&\quad - \frac{\lambda_m}{2}\|\boldsymbol{\varrho}_m^{(1)}\|_2^2.
\end{aligned}$$

Summing the above equality over $m \in \mathcal{M}$ yields in

$$\begin{aligned}
\sum_{m=1}^{M} \Re(\langle \hat{\mathbf{x}}_{m-1}^{(1)} - \mathbf{x}, \boldsymbol{\varrho}_m^{(1)} \rangle) &= \sum_{m=1}^{M} \frac{1}{2\lambda_m}(\|\hat{\mathbf{x}}_m^{(1)} - \mathbf{x}\|_2^2 \\
&\quad - \|\hat{\mathbf{x}}_{m-1}^{(1)} - \mathbf{x}\|_2^2) - \sum_{m=1}^{M} \frac{\lambda_m}{2}\|\boldsymbol{\varrho}_m^{(1)}\|_2^2 \\
&\overset{(a)}{=} \frac{1}{2\lambda_M}\|\hat{\mathbf{x}}_M^{(1)} - \mathbf{x}\|_2^2 - \frac{1}{2\lambda_1}\|\hat{\mathbf{x}}_0^{(1)} - \mathbf{x}\|_2^2 \\
&\quad - \sum_{m=1}^{M} \frac{\lambda_m}{2}\|\boldsymbol{\varrho}_m^{(1)}\|_2^2 \\
&\overset{(b)}{\geq} -\frac{1}{2\lambda_1}\|\mathbf{x}\|_2^2 - \sum_{m=1}^{M} \frac{\lambda_m}{2}\|\boldsymbol{\varrho}_m^{(1)}\|_2^2,
\end{aligned}$$

where in (a) we use the fact that the first sum on the right-hand side is a telescopic sum, and in (b) we have lower bounded the left-hand side and assumed that $\hat{\mathbf{x}}_0^{(1)} = \mathbf{0}_{K\times 1}$. Eq. (11) then follows by taking the expectation of the above expression w.r.t. $\mathbf{x}$ and $\mathbf{n}$ conditioned on the perfect CSI knowledge denoted by the set $\{\mathbf{h}_m\}$, $\forall m \in \mathcal{M}$. Then, lower bounding $\mathbb{E}[\|\mathbf{x}\|_2^2|\{\mathbf{h}_m\}]$ by $B^2$, $\mathbb{E}[\|\boldsymbol{\varrho}_m^{(t)}\|_2^2|\{\mathbf{h}_m\}]$ by $\rho^2$ for every $m \in \mathcal{M}$, and defining $\lambda_m$ to incorporate $B$, $\rho$, $\sqrt{m}$, $\sqrt{t}$ in addition to $\|\mathbf{h}_m\|_2^2$, which comes from the definition of the Kaczmarz iterative step in (4), leads to (12). It is straightforward to obtain the loose lower bound $B^2 = Kp$ from $\mathbf{x} \sim \mathcal{CN}(\mathbf{0}, p\mathbf{I}_K)$. For $\rho$, we consider the worst-case that the residual in $\boldsymbol{\varrho}_m^{(t)}$ is pure noise and has the form $n_m \sim \mathcal{CN}(0, \sigma^2)$, then $\rho^2 = \sigma^2$; the channel is normalized out to set $B^2$ and $\rho^2$ on the same power scale.

[9] It is straightforward to generalize Theorem 1 for other number of cycles, one just needs to use the solution from the previous cycle as the initialization of the next $\hat{\mathbf{x}}_0^{(t+1)} = \hat{\mathbf{x}}_M^{(t)}$.