Density Based Clustering for Downlink User Grouping in FDD Massive MIMO

Alessandro Grassi[1], Martin Kurras[2], Giuseppe Piro[1], Gennaro Boggia[1], Stephan Faehse[2], Lars Thiele[2]
[1]*Dept. of Electrical and Information Engineering, Politecnico di Bari*, Bari, Italy.
CNIT, Consorzio Nazionale Interuniversitario per le Telecomunicazioni
Email: {name.surname}@poliba.it
[2]*Wireless Communications and Networks, Fraunhofer Heinrich Hertz Institute*, Berlin, Germany.
Email: martin.kurras@hhi.fraunhofer.de

***Abstract*—Recently, the two-stage Joint Spatial Division and Multiplexing (JSDM) precoding scheme was introduced as a valuable solution for implementing, with a reduced complexity, Massive MIMO transmission techniques in emerging 5th Generation of mobile networks. Among its main features, JSDM requires to partition mobile users into groups, based on the channel covariance similarity. To this end, the Density-Based Clustering of Applications With Noise (DBSCAN) clustering algorithm has already proved to be reasonably suitable. Unfortunately, DBSCAN falls short in challenging situations, like a large homogeneous crowd of users. Here, in fact, all the users are put in the same group despite experiencing different channel covariances, which degrades the behavior of JSDM. Based on these premises, the contribution presented herein proposes a modified version of DBSCAN, which enforces a maximum size on detected clusters by limiting the difference in channel covariances between any pair of users in the same group. In line with the requirements of JSDM, the modified DBSCAN breaks down excessively large clusters into smaller ones, thus extending its applicability to a much wider range of scenarios (from small dispersed user groups to large compact ones). Computer simulations confirm the behavior predicted by the theoretical formulation: the proposed approach reaches a better user and system spectral efficiency, while showing low sensitivity against parameter tuning.**

***Index Terms*—Massive MIMO, 5G, JSDM, clustering, DBSCAN**

I. Introduction

The 5th Generation (5G) of mobile networks introduces new technologies, which promise features and flexibility that are unprecedented in current cellular networks and other wireless systems, enabling multi-Gbps peak speeds, large swarms of autonomous devices, real-time applications with extremely low latency, and many more [1]. The Massive MIMO communication technique is one of the key components foreseen in this context [2]. It adopts hundreds of antennas at the base station to concurrently serve more users than in current systems, while simultaneously reducing energy consumption and only requiring relatively simple linear processing. Originally, it uses Time Division Duplexing (TDD) operation to take advantage of the downlink-uplink duality. However, many spectrum allocations are intended for Frequency Division Duplexing (FDD) mode only. This poses additional difficulties, as the training overhead in the downlink direction increases proportionally with the number of transmitting antennas, thus reducing the spectral efficiency for actual data as more antennas are added [3]. Also, the size of the Channel State Information (CSI) feedback that needs to be sent in the uplink direction increases in a similar manner.

At the time of this writing, the Joint Spatial Division and Multiplexing (JSDM) technique, firstly proposed in [4], emerged as a possible method to mitigate these unfavorable trends. It is a two-stage precoding approach, where the first stage is common to multiple users and is only based on channel covariance matrices. These change slowly and can be reported less often than instantaneous CSI. They also need to be of low rank, which can happen in different circumstances [**?**]. Hence, the training overhead and instantaneous CSI required for the second stage are greatly reduced. The first stage of JSDM needs a way to identify groups of users with similar statistical CSI. This requirement is more cumbersome than it may appear, because it should be achieved over a large range of situations. In [5], this goal is approached with k-means partitioning, which iteratively assigns each user the the closest cluster, and with fixed partitioning of the angular domain, with the latter being simpler and providing better results. Instead, [6] proposes an agglomerative clustering method as an alternative to k-means. While it is faster for low numbers of users, it becomes comparable or even slower when hundreds of users are considered. The work in [7] compares k-means, k-means++, and Density-Based Clustering of Applications With Noise (DBSCAN) for the same purpose, in a scenario where the clusters already exist and need to be detected. The results show that k-means and k-means++ are not well-suited, mainly because they assign all the users to some cluster, even isolated ones. Instead, the DBSCAN approach came out as the most promising, it can exclude outliers and parameter selection is relatively easy. These findings are confirmed in [8], where DBSCAN is similarly compared to k-means and k-medoids methods.

However, DBSCAN presents important limitations in many scenarios. For example, when there is a large compact crowd of users with a mostly constant density, it would be detected as a single cluster. This poses serious problems for JSDM. From one side, it may not be possible to serve multiple groups, thus limiting the total throughput. From the other side,

users in a large group may have very different covariance matrices, which results in intra-group interference at the first-stage precoder.

In this work, we extend the baseline DBSCAN approach with a configurable limit on the cluster size. This way, large clusters are split into smaller parts, which are more suitable for JSDM precoding. The resulting algorithm retains the desirable properties of DBSCAN identified in prior work, and can be applied to a much broader set of scenarios. This is verified by numerical simulations, which show an increase of both user and system spectral efficiencies, and also highlight that the algorithm parameters do not need a very precise tuning.

This paper is organized as follows: Section II describes the general system model, while Section III describes the improved clustering algorithm, and Section IV presents the simulation results. Finally, Section V closes the paper.

II. System model

Throughout this paper, scalar quantities are denoted with italic letters (a, A), while vectors are written with lowercase bold letters ($\mathbf{a}$) and matrices are assigned uppercase bold letters ($\mathbf{A}$). Sets are represented by calligraphic letters ($\mathcal{A}$), and their elements are listed in square brackets ($[a_1, a_2, \ldots, a_n]$). $(\cdot)^{\mathrm{H}}$ represents the Hermitian matrix, while $\mathbb{E}[\cdot]$ is the expectation operator, and $\|\cdot\|_F$ is the Frobenius norm of a matrix.

In the considered scenario, the base station is equipped with N antennas and a Multiple Input Multiple Output (MIMO) downlink block-fading channel is taken into account. At the end user side, there are K single antenna receivers. Thus, the receive signal $\mathbf{y} \in \mathbb{C}^K$ is written as:

$$\mathbf{y} = \mathbf{H}^{\mathrm{H}}\mathbf{V}\mathbf{x} + \mathbf{n}, \tag{1}$$

where $\mathbf{H} \in \mathbb{C}^{N\times K}$ is the radio channel between the N antennas of the base station and the K antennas of the end users, $\mathbf{V} \in \mathbb{C}^{N\times T}$ is the precoding matrix for T independent data streams, $\mathbf{x} \in \mathbb{C}^T$ is the data vector, and $\mathbf{n} \in \mathbb{C}^K$ denotes the additive white Gaussian noise assuming $\mathbf{n} \propto \mathcal{CN}(1, \mathbf{I}_K)$. Since linear precoding is used, the number of data streams is bounded by $T \leq \min(N, K)$ [4]. By following the assumptions in [5], the transmit power is equally split to all data streams, such that $\mathbb{E}\left[\mathbf{x}\mathbf{x}^{\mathrm{H}}\right] = \frac{P}{T}\mathbf{I}_T$, where P is the total transmit power. The radio channel of the receiver $k \in \mathcal{K} = [1, \ldots, K]$ follows $\mathbf{h}_k \propto \mathcal{CN}(0, \mathbf{R}_k)$, where the covariance matrix $\mathbf{R}_k$ is positive semi-definite. $\mathbf{R}_k$ is a so called second-order statistic of the radio channel with a longer coherence time than direct CSI of $\mathbf{h}_k$. The singular value decomposition of $\mathbf{R}_k$ is given as:

$$\mathbf{R}_k = \mathbf{U}_k\mathbf{\Lambda}_k\mathbf{U}_k^{\mathrm{H}}, \tag{2}$$

where $\mathbf{U}_k \in \mathbb{C}^{N\times b_k}$ contains the b_k eigenvectors that correspond to the non-zero b_k eigenvalues in the diagonal matrix $\mathbf{\Lambda}_k$ of size $b_k \times b_k$. A subset of $\mathbf{U}_k$ is used in Section III to obtain the input metric for receiver clustering.

Following the two stage precoding in [4], the precoder $\mathbf{V} = \mathbf{B}\mathbf{P}$ is split into a first and second stage precoder. According to the clustering approach described in Section III, the K receivers are divided into G clusters of size K_g, such that $K = \sum_g^G K_g$, where subscript $g \in [1, \ldots, G]$ is the cluster index. In this work, the terms "group" and "cluster" are used interchangeably to denote the disjoint subsets of users produced by a clustering algorithm. Following the guidelines in [4] receivers within the same cluster should have similar covariance matrix eigenvalues whereas the eigenvalues of different clusters should be orthogonal. The first stage precoder $\mathbf{B} \in \mathbb{C}^{N\times b}$ is a function of second order statistics to coordinate inter-cluster interference. The second stage precoder $\mathbf{P} \in \mathbb{C}^{b\times T}$ is a function of direct CSI of the effective channel $\tilde{\mathbf{H}} = \mathbf{B}^{\mathrm{H}}\mathbf{H}$. To take into account the G receiver clusters, $\mathbf{H}_g = \left[\mathbf{h}_{g1}, \ldots, \mathbf{h}_{gK_g}\right]$ denotes the channel matrix for one group as the concatenation of the individual users' channels, and the first-stage precoder becomes $\mathbf{B} = [\mathbf{B}_1, \ldots, \mathbf{B}_G]$ where $\mathbf{B}_i$ is the i-th group's precoding matrix. Also, $\mathbf{P}$ can be written as $\mathbf{P} = \mathrm{diag}(\mathbf{P}_1, \ldots, \mathbf{P}_G)$, where $\mathbf{P}_i$ is the i-th group's diagonal power allocation matrix, and $\mathbf{x}_g = \left[x_{g1}, \ldots x_{gK_g}\right]$ is the matrix of all the user signals in group g.

With these definitions, the receive signal reported in Eq. (1) can be rewritten for the user k in cluster g as:

$$y_{gk} = \underbrace{\mathbf{h}_{gk}^{\mathrm{H}}\mathbf{B}_g\mathbf{p}_k}_{\hat{h}_{gk}}x_k + \underbrace{\sum_{k'\neq k}^{k'\in\mathcal{K}_g}\mathbf{h}_{gk'}^{\mathrm{H}}\mathbf{B}_g\mathbf{p}_{k'}x_{k'}}_{z_{gk}} + \underbrace{\sum_{g'\neq g}\mathbf{H}_g^{\mathrm{H}}\mathbf{B}_{g'}\mathbf{P}_{g'}\mathbf{x}_{g'}}_{\hat{z}_{gk}} + n_k, \tag{3}$$

where subscript gk denotes receiver k in cluster g, $\mathcal{K}_g = [1, \ldots, K_g]$ the set of receivers in cluster g and $\hat{h}_{gk}, z_{gk}, \hat{z}_{gk}$ comprise the signal, inner-cluster and inter-cluster interference, respectively. The advantage of this scheme is that in FDD, the dimension of $\tilde{\mathbf{H}}$ is $b \times K$, where $b = \sum_{g=1}^G b_g$, and the required feedback per receiver is reduced from $\mathbf{h}_{gk} \in \mathbb{C}^N$ to $\tilde{\mathbf{h}}_{gk} \in \mathbb{C}^{b_g}$ assuming that $b_g \ll N$.

Thus, the resulting Signal to Interference plus Noise Ratio (SINR), γ_{gk}, is given by:

$$\gamma_{gk} = \frac{\hat{h}_{gk}^{\mathrm{H}}\hat{h}_{gk}}{z_{gk}^{\mathrm{H}}z_{gk} + \hat{z}_{gk}^{\mathrm{H}}\hat{z}_{gk} + \mathbb{E}\left[n_k^{\mathrm{H}}n_k\right]} \tag{4}$$

and the corresponding capacity as normalized with the bandwidth assigned to receiver k in cluster g is obtained by:

$$C_{gk} = \log_2(1 + \gamma_{gk}). \tag{5}$$

III. Clustering

As stated in the previous section, JSDM uses the same first-stage precoder for all the users within one group. This implies that it should match all the covariance matrices of the users. If the precoder does not match the covariance of a given user, then the orthogonality between groups is violated: some of the power allocated to the target user is received by other groups, and the user itself is susceptible to interference from other groups. For a perfect matching, the users in one group should have the exact same covariance matrix, but this is not possible in practice. Instead, it is possible to find groups of users that have similar covariance matrices, so that the inter-group interference is small enough to have a limited impact.

This is an example of a clustering problem, which usually occurs in data science, and whose typical terminology derives from that field. Given a set of data points, potentially very large, it is necessary to partition it into groups. The points in each group should be as close to each other as possible, according to a given similarity measure, while groups should be as far apart as possible. In the context of Massive MIMO and JSDM, some clustering approaches have been investigated in [7], where the density-based approach DBSCAN is shown by simulations to be better than the parametric-based methods, i.e. k-means and k-means++. Specifically, it was assumed that the users were actually located in well-separated clusters, except for a few outliers, and the clustering step had to detect such clusters. In this work, we target a different user distribution where DBSCAN performs poorly, and propose an improved version that can cope with the new situation. The original DBSCAN algorithm is described in Section III-A, while the new challenging scenario and the improved DBSCAN algorithm are presented in Section III-B.

A. DBSCAN

In the DBSCAN clustering algorithm [9], a cluster is intuitively defined as a region of the feature space where there is a higher density of data points, compared to regions outside of the cluster. A more formal definition starts with the notion of *Eps-neighborhood*: given a distance function $\mathrm{dist}(p, q)$ and a distance Eps, the Eps-neighborhood $N_{Eps}(p)$ of a data point p is defined as the set of points no farther than Eps from p:

$$N_{Eps}(p) = \{q | \mathrm{dist}(p, q) \leq Eps\} \tag{6}$$

A simple approach could define a cluster as a set of points whose Eps-neighborhood contain at least a certain number $minPts$ of data points. However, this would be sub-optimal, because points on the edge of the cluster (*border points*) typically have a lower number of points in their Eps-neighborhood, compared to internal points (*core points*), even if the density of the cluster is constant. By using a lower value for $minPts$ would underestimate the density of the cluster, which can be problematic in the presence of noise. A better solution is to require core points to have at least $minPts$ points in their Eps-neighborhood, and border points to be *directly density-reachable* from one or more core points. Given a core point q, a point p is directly density-reachable from q if it is contained in the Eps-neighborhood of q.

The directly density-reachability property can only connect points in a cluster that are close to each other. The definition of an entire cluster requires two more concepts: *density-reachable* and *density-connected*. A point p is density-reachable from a point q if there exist a sequence of points $p_1 = q, p_2, \ldots, p_{n-1}, p_n = p$ which connects q to p, so that p_{i+1} is directly density-reachable from p_i. This allows going from any core point to any other point of the cluster, but not from border points to other points. The notion of density-connected points fills this last gap: two points p and q are said to be density-connected if they are both density-reachable from some point o. Now, a cluster can be unambiguously identified by any one of its points, either a core point or a border point, together with all of the points that are density-connected to it.

DBSCAN relies on such definition of cluster. Ideally, given the right values of both Eps and $minPts$, which may be different for each cluster, and a starting point for each cluster, it is straightforward to assign each other point to the right cluster. However, this would require some previous knowledge of the data that is usually not available. Instead, DBSCAN uses a single value for Eps and for $minPts$, which are assumed to be valid for the thinnest cluster and thus would also work for more dense clusters.

The authors of [9] describe a procedure to determine such parameters with minimal user interaction. The algorithm starts from the first point in the database, finds all the points that are density-connected to it, and marks them as belonging to the same cluster. Then it picks the next unclassified point and repeats the procedure. This is repeated until all points are either assigned to a cluster or marked as noise points, i.e. points that are not density-reachable from any other point.

The detection of noise points is particularly important in the context of JSDM: in fact, they represents users that have covariance matrices quite different from any other user, and thus can not be assigned to any group. As already observed in [7], the best strategy for dealing with these users is to schedule them in dedicated slots.

The pseudo-code for most parts of DBSCAN is given in [9], and is used in this work without modification. Here we only give the pseudo-code used for the *RegionQuery* function, that is not reported in the original work. This function returns the Eps-neighborhood ($Neps$ in the pseudo-code) of a given point p, and it is implemented by Algorithm 1. The improved version of this function will be described later. The distance function used here is the chordal distance between matrices, applied to the eigenspaces of the users' covariance matrices:

$$\mathrm{dist}(\mathbf{U_1}, \mathbf{U_2}) = \|\mathbf{U_1}\mathbf{U_1}^{\mathrm{H}} - \mathbf{U_2}\mathbf{U_2}^{\mathrm{H}}\|_{\mathrm{F}}^2 \tag{7}$$

Algorithm 1 RegionQuery for the original DBSCAN

```
Input: p, Eps
  for i := 0 to SetOfPoints.size do
    q := SetOfPoints.get(i)
    // Check that q is not assigned to other clusters already
    if q.ClusterId IN {NONE,NOISE,p.ClusterId} then
      if dist(p,q)<Eps then
        Neps.append(q)
      end if
    end if
  end for
  return Neps
```

B. Improved DBSCAN

DBSCAN is effective at detecting clusters when they are delimited by a clear decrease of data points density, i.e. when the original data is already spontaneously clustered. This is

assumed in [7], where the groups of users are separated by a certain distance. In this work, we address the totally opposite scenario where there are no clearly identifiable groups, but it is necessary to artificially create them.

Suppose that there is a dense crowd of mobile users spread over a large area, i.e. at an outdoor concert or similarly crowded events, with no significant fluctuations in density, and we use JSDM for downlink transmission. The area of interest should be large enough to span the coverage area of the closest base stations, which is likely if micro/pico cells are deployed. With DBSCAN, each of these base station would only detect a single large cluster of users in its coverage area, because their density stays essentially constant. However, this is detrimental for the use of JSDM for two reasons: (i) the number of groups that can be concurrently served is reduced to one, and (i) users sufficiently distant from each other will have different covariance matrices, so it is not possible to find a first-stage precoder that applies to the entire group.

A better strategy to handle this situation is to split the large cluster into smaller clusters that have a limited "size", measured through the chordal distance between the eigenspaces of the users. It should be tuned so that all the users of one group can be served with the same first-step precoding matrix. Then, the base station would select a set of groups that are suitable for co-scheduling, i.e. groups that are sufficiently separated in the angular domain. If such condition is not met, they would still experience significant mutual interference [5].

It would be cumbersome to have different clustering methods for many scenarios, as it would be necessary to detect such scenarios as well. Instead, we modified DBSCAN to work as desired in both these extreme cases. It should also cover all the intermediate cases, although we can't present these in this paper due to the limited space. The key idea is to modify the *RegionQuery* function so that when it builds the Eps-neighborhood of point p, it also takes into account the centroid of the cluster where p belongs. Define $D_{max} \in \mathbb{R}^+$ as the maximum acceptable diameter for a cluster. Therefore, if a point is further away then $D_{max}/2$ from the centroid, it is not included in the Eps-neighborhood of p. The extended *RegionQuery* function is reported in Algorithm 2. Given the eigenspaces $\mathrm{U}_1, \ldots, \mathrm{U}_\mathrm{N} \in \mathbb{C}^{M \times b}$ and the corresponding eigenvalues $D_1, \ldots, D_N \in \mathbb{C}^b$ of the N users assigned to a cluster, the eigenspace of its centroid is calculated as:

$$\mathrm{U_c} = \mathrm{eig}\left[\frac{1}{N}\sum_{i=1}^{N} \mathrm{U_i} \mathrm{diag}\left(D_i\right) \mathrm{U_i^H}\right] \tag{8}$$

where $\mathrm{diag}(x)$ is the diagonal matrix with the elements of x on its diagonal. The centroid is updated for each invocation of *RegionQuery*, because new points could have been added between subsequent invocation.

This updated formulation introduces new properties, while retaining others that already existed in DBSCAN. First of all, the detection of border points with respect to Eps and $minPts$ is untouched, so the limits of clusters that already exist are still recognized. Thus, in the scenario of [7] with small and well-separated clusters, the modified DBSCAN works exactly as the original one, provided that the new parameter D_{max} is sufficiently large. In fact, this is true in any case: one can always find an D_{max} so large that the new DBSCAN behaves as the original one, with $D_{max} \to \infty$ as the extreme case.

As for the new scenario with a single large cluster, the selection of D_{max} creates different cases. Note that as D_{max} decreases, the number of detected groups $G \in \mathbb{N}$ grows and their average user count $\widehat{G} \in \mathbb{R}^+$ decreases. The possible outcomes, differentiated for their effect on the JSDM precoding, are listed here in the order of decreasing D_{max}:

- For $D_{max} \to \infty$, the new DBSCAN behaves as the original DBSCAN, i.e. $G = 1$. The difference between user covariances can be large, and JSDM is not useful.
- For some value of D_{max}, the algorithm results in $G > 1$, but the clusters are still too large to meet the assumption of almost-identical covariance throughout one group. JSDM is still not usable at its full potential.
- For decreasing D_{max}, G gets larger and the groups get smaller with respect to the distance between users' eigenspaces. As long as $\widehat{G} \geq b$, JSDM can be used with high performance.
- For even smaller D_{max}, the groups get so small that $\widehat{G} < b$. JSDM can still be used, but less users can be served and the sum throughput diminishes.
- For $D_{max} \to 0$, all the users are marked as outliers ($\widehat{G} = 1$) and JSDM can't be used anymore.

Thus, the value chosen for D_{max} is critical for the outcome of the clustering step, and it should be selected according to the specific application and its requirements. For our use in the context of JSDM, we can observe the following: D_{max} is expressed with respect to the chordal distance between users' eigenspaces, which directly affects the performance of the two-step precoder. Thus, the optimal value for a given scenario is very likely to work in most other situations: when there are large groups, they would be split in smaller groups of reasonable size according to D_{max}, and when they are small, they would just be detected and used as they are. Ultimately, this modified version of DBSCAN would be able to handle both the extreme cases outlined here, intermediate situations, and scenarios with both small and large groups, without additional tuning.

IV. Numerical results

The improved version of DBSCAN has been evaluated through computer simulations, by assuming a large-crowd scenario as described in Section III. Specifically, mobile users are evenly distributed over a sub-region of a base station sector, as shown in Figure 1. The distance from the base station varies from 100 to 225 m, and the azimuth ranges from $-60°$ to $60°$. The angular spread varies with the distance, from $28°$ for the closest users to $13.5°$ for the farthest users.

Parameters from [7] are reused as much as possible to make comparison easier: the base station uses a Massive MIMO array with 256 antennas, with $\lambda/2$ spacing, center frequency of 2.5 GHz, and JSDM for precoding. For the first stage,

Algorithm 2 RegionQuery for the improved DBSCAN

Input: p, Eps, Dmax
// Calculate centroid of the cluster where p is assigned
c := SetOfPoints.centroid(p.ClusterId)
for i := 0 **to** SetOfPoints.size **do**
q := SetOfPoints.get(i)
// Check that q is not assigned to other clusters already
if q.ClusterId IN {NONE,NOISE,p.ClusterId} **then**
if dist(p,q)<Eps **and** dist(c,q)<Dmax/2 **then**
Neps.append(q)
end if
end if
end for
return Neps

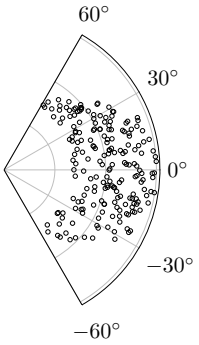

Fig. 1. Positions of the mobile users

the design parameter b of JSDM is set to 6, and the first-stage precoder, valid for all the users in the same group, is matched to the covariance matrix of one of the scheduled users, that is selected randomly [5]. As for the second-stage precoder, it uses Regularized Zero-Forcing (RZF) [10] and Per-Group Processing (PGP) [5]. For channel modeling, we use the Kronecker correlation model [11] together with the one-ring scattering model [12].

At the clustering step, the improved version of DBSCAN is employed, with parameters $Eps = 3.0$, $minPts = 3$, and D_{max} from 0.5 to 24. After clustering, an ideal selection is assumed for which clusters are actually served. Since the best results are obtained when the clusters are well-separated in the angular domain, we select the three clusters closest to azimuth directions $-60°$, $0°$, and $60°$. After the group selection, there is an intra-group scheduling to randomly select $b = 6$ users within each group, unless the group already has 6 or less users. A more realistic selection of the clusters is left for future study.

The receivers are equipped with a single antenna, and the total equivalent Signal to Noise Ratio (SNR) at the end user side is 50 dB. This means that if a single user is served and no interference is considered, its SNR is 50 dB. With multiple users, the equivalent SNR is split among them in equal parts, and then their individual intra-group interference and inter-group interference are also added. After all the precoding and scheduling steps, the final SINR experienced by each user is evaluated, and the Spectral Efficiency (SE) is derived using Shannon's capacity formula. The entire process is repeated for 30 independent channel realizations.

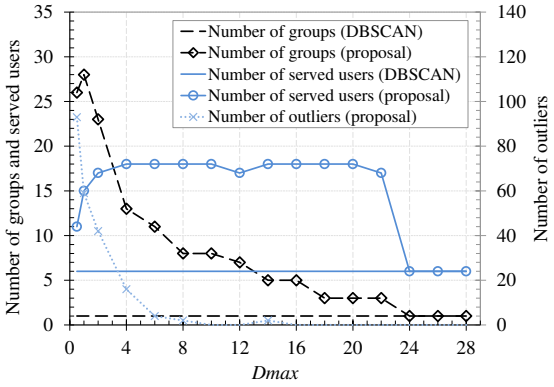

Fig. 2. Number of detected groups and served users

Figure 2 shows the number of groups detected during the clustering phase, as a function of D_{max}, together with the total number of users served in each case. For $D_{max} \geq 24$ there is only one group, which means that the algorithm is working like the classical DBSCAN, treating the entire population as a single density-connected cluster. Consequently, only one group can be scheduled and only $b = 6$ users can be served. For lower values of D_{max}, more and more sub-clusters are created, and in most cases it is possible to serve 3 groups with 6 users each, reaching 18 users in total. However, with $D_{max} < 4$, the number of served users starts decreasing, because groups get increasingly smaller, to the point of containing less than 6 users. As a consequence, the multiplexing gain is reduced. Also note that at the lowest considered value of $D_{max} = 0.5$, almost half of the users are marked as noise point and thus can't be served as part of a cluster.

Figure 3 shows the sum SE and the user SE, both as a function of D_{max}. They are proportional for most of the values, i.e. for the entire range where the number of served users is high and stable. However, there is a sharp contrast between the left part ($4 \leq D_{max} \leq 12$) and the right part ($12 < D_{max} \leq 22$), as the left part exhibits consistently higher values than the right one. The reason is that for $D_{max} > 12$, the groups become so large that some adjacent groups must be scheduled together, which creates high inter-group interference and thus lower SE values. Please note that the performance in the left part is quite stable, thus the optimal value for D_{max} does not need to be estimated with extreme accuracy.

At the extremes of the investigated region, the user SE increases but the sum SE decreases. For the sum SE, it is easily noted that the lower values are due to the lower number of served users. As for the user SE, the reason is different for the two cases. With very low values of D_{max}, the chordal distance between the user eigenspaces is reduced, so the first-stage precoder is better matched to all the users in the group, leading to higher SE. Instead, when $D_{max} \geq 24$, the user SE suddenly increases because there is only one group, and thus

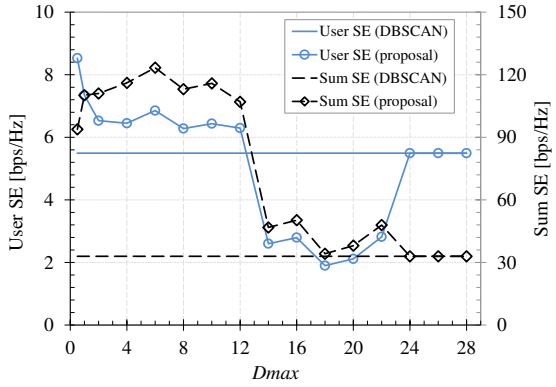

Fig. 3. Average user SE and sum SE

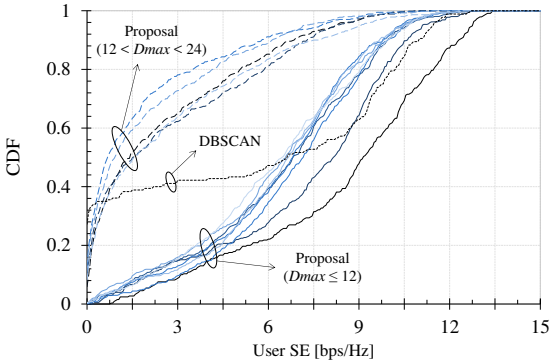

Fig. 4. Cumulative Density Function (CDF) of user SE [bps/hz]

the inter-group interference disappears. However, this comes at the cost of serving less users and is thus undesirable.

Figure 4 shows the CDF of the user spectral efficiency, for different values of D_{max}. Again, there is a sharp difference between $D_{max} > 12$ region (dashed lines) and the $D_{max} \leq 12$ region (solid lines). In the first case, many users have low throughput and only few users have high throughput, similarly to a maximum-throughout scheduler with low fairness. This is due not only to both inter-group interference between adjacent groups, but also intra-group imbalance: recall that the first-stage precoder is based only on one of the users, and in large groups other users may not be well matched to it. Conversely, in the second case, the CDFs look more like a round-robin scheduler with higher fairness. In this regime, inter-group interference is more limited because the scheduled groups are not adjacent, and intra-group imbalance is also reduced due to the lower group size. This confirms that the ideal operating region is $4 \leq D_{max} \leq 12$ in the adopted scenario. Regarding complexity, computing the additional distance from the cluster center can make RegionQuery twice as expensive. However, users that have been already classified in other groups are completely ignored by it, therefore creating smaller groups removes users from calculations earlier. In the optimal operating range for the considered scenario, this results in a net complexity gain compared to the original DBSCAN.

V. Conclusion

In this work we propose a modified version of the DBSCAN algorithm for downlink user grouping in Frequency Division Duplexing Massive MIMO. The proposed modification is an additional stopping criteria, controlled by a new system design parameter D_{max}, in order to ensure the similarity of the second order channel statistics of receivers within the same cluster, and thus enforce a maximum cluster size. We studied the impact of this new control parameter by numerical evaluation, and showed that in large dense user distributions the modified algorithm works as intended. In terms of sum and user spectral efficiency the proposed modified DBSCAN algorithm can outperform the original DBSCAN algorithm. Some possible future improvements on this topic include a more realistic selection of the clusters to be scheduled, and the selection of the clusterhead within each user cluster.

Acknowledgements

This work is supported by the FANTASTIC-5G project, which receives funding from the European Union's Horizon 2020 research and innovation programme under the grant agreement ICT-671660.

References

[1] NGMN Alliance, "5G White Paper," 2015.
[2] E. G. Larsson, O. Edfors, F. Tufvesson, and T. L. Marzetta, "Massive MIMO for next generation wireless systems," *IEEE Communications Magazine*, vol. 52, no. 2, pp. 186–195, 2014.
[3] E. Björnson, E. G. Larsson, and T. L. Marzetta, "Massive MIMO: Ten myths and one critical question," *IEEE Communications Magazine*, vol. 54, no. 2, pp. 114–123, 2016.
[4] A. Adhikary, J. Nam, J.-Y. Ahn, and G. Caire, "Joint spatial division and multiplexing - The large-scale array regime," *IEEE transactions on information theory*, vol. 59, no. 10, pp. 6441–6463, 2013.
[5] J. Nam, A. Adhikary, J.-Y. Ahn, and G. Caire, "Joint spatial division and multiplexing: Opportunistic beamforming, user grouping and simplified downlink scheduling," *IEEE Journal of Selected Topics in Signal Processing*, vol. 8, no. 5, pp. 876–890, 2014.
[6] X. Sun, X. Gao, G. Y. Li, and W. Han, "Agglomerative user clustering and downlink group scheduling for FDD massive MIMO systems," in *Proc. of IEEE International Conference on Communications (ICC)*. IEEE, 2017, pp. 1–6.
[7] M. Kurras, S. Fähse, and L. Thiele, "Density Based User Clustering for Wireless Massive Connectivity Enabling Internet of Things," in *Proc. of Globecom Workshops*. IEEE, 2015, pp. 1–6.
[8] Y. Sun, S. Lv, S. Liu, and Y. Zhang, "Density based user grouping for massive MIMO downlink in FDD system," in *Proc. of IEEE 9th International Conference on Communication Software and Networks (ICCSN)*, 2017.
[9] M. Ester, H.-P. Kriegel, J. Sander, X. Xu *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise." in *Proc. of Conference on Knowledge Discovery and Data Mining (KDD)*, vol. 96, no. 34, 1996, pp. 226–231.
[10] C. B. Peel, B. M. Hochwald, and A. L. Swindlehurst, "A vector-perturbation technique for near-capacity multiantenna multiuser communication-part I: channel inversion and regularization," *IEEE Transactions on Communications*, vol. 53, no. 1, pp. 195–202, 2005.
[11] A. Van Zelst, "A compact representation of spatial correlation in MIMO radio channels," in *submitted to International Conference on Communications (ICC)*, 2004.
[12] D.-S. Shiu, G. J. Foschini, M. J. Gans, and J. M. Kahn, "Fading correlation and its effect on the capacity of multielement antenna systems," *IEEE Transactions on communications*, vol. 48, no. 3, pp. 502–513, 2000.