Research Article

Feedback Overhead in the FDD Downlink Massive MIMO System

Papis Ndiaye and Moussa Diallo

Department of Computer Science, Polytechnic Institute (ESP), Cheikh Anta Diop University, Dakar, Senegal

Correspondence should be addressed to Papis Ndiaye; idrissa.ndiaye@esp.sn

Received 3 April 2022; Revised 2 June 2022; Accepted 12 July 2022; Published 27 July 2022

Academic Editor: Liu Liu

Copyright © 2022 Papis Ndiaye and Moussa Diallo. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

One of the fundamental problems in massive multiple input multiple output (MMIMO) systems based on the frequency division duplexing (FDD) mode is the acquisition of the channel state information at the base station (BS). Indeed, in these systems, the reciprocity of the uplink (UL) and downlink (DL) channels is not applicable. Thus, the users are supposed, after having estimated their subchannels, to feedback them to the BS. Since the size of the information to be returned is proportional to the number of base station antennas that is very large, then we witness an excess of the number to be returned for each user. It is therefore essential to find a way to reduce this quantity, which affects the overall performance of the system. In this article, we propose two new MMIMO-OFDM-FDD-DL schemas based on the Givens rotation quantization used in the 802.11ac standard to reduce the amount of information to be returned for each user. The first is a fusion between the Givens rotation (GR) and the arithmetic coding (AC) technique, and the second is based on the combination of the GR method and the universal compression technique, Lempel Ziv Welch (LZW). The proposed schemes named GR & AC and GR & LZW (with an improvement of this proposal based on a modified version of LZW (LZWM) named GR & LZWM) for the first and second, respectively, are compared and both can achieve a compression ratio of up to 95% (or even 98%) and 83%, considerably exceeding recent advanced techniques.

1. Introduction

MMIMO systems can significantly increase the capacity and energy efficiency of communication systems by using an excessive number of antennas (more than 100 at the BS). It has been shown in [1] that increasing the number of antennas at the BS is always beneficial even with a very noisy channel estimate, because in this case, the BS can recover information even with a low signal-to-noise ratio (SNR). This strongly justifies the use of a very large number of transmit antennas compared to current cellular systems [2]. This is the reason this technology has been named as the most promising for 5G and 6G communications.

Note that in the MMIMO technology, obtaining the channel state information (CSI) for transmission (BS) and reception (mobile station) is essential for beamforming, signal detection, and optimization of resources allocation. Moreover, the manner of obtaining this CSI differs generally with respect to the mode of duplexing used. There are two nowadays: the time domain duplexing (TDD) mode and the frequency domain duplexing (FDD) mode.

In the TDD mode, the CSI can be easily obtained at BS. Indeed, in this mode, each user sends their pilot symbol to the BS and the BS estimates all subchannels. Since in the TDD mode there is reciprocity of the downlink and uplink channels, we only need one to deduce the other. Therefore, after estimating the channels in the uplink, the BS knows those in the downlink in an implicit way. Unlike the TDD mode, in the FDD mode, the downlink and uplink channels cannot be considered reciprocal: they are different. In this case, it is up to the BS to send the pilot symbols in a broadcast, and the estimation of the downlink subchannels will be done at the level of each user to obtain the CSI. After that, each CSI is sent back to the BS. Since the characteristic of MMIMO systems is the presence of several antennas at the BS, we witness not only an excess of pilot symbols to estimate the transmission channel but also the amount of feedback information becomes essential.
Therefore, a fundamental question arises: in a FDD MMIMO system, how to significantly reduce the feedback overhead?

The answer to this question is so unclear that early work on MMIMO systems [2, 3] chose the TDD mode which is more reasonable and easy to use in this context. Recall that in the TDD mode, even if the downlink and uplink channels are reciprocal, the front-end radios are not, and there are problems of calibration, synchronization, and pilot contamination. In addition, the most current wireless communication networks operate in FDD mode. Thus, the answer to this question becomes a capital necessity. In other words, it would be interesting to see how to use the FDD mode optimally in MMIMO systems. This is the reason this work will focus on this issue.

All works that have attempted to address this problem can be categorized as follows.

(i) The codebook principle is widely used. This technique consists of building a codebook (using the Grassmann manifold theory, for example) available at the transmitter and the receiver. This dictionary contains a number of predefined precoding matrices with their associated indices. Once the receiver obtains the CSI after channel estimation, it chooses from the dictionary matrix the one that best approximates the channel conditions and feeds back its index on the return path. Sending the index only on the return path significantly reduces the feedback overhead. The disadvantage of this technique is that the selected matrix is an approximate version: there will be quantization errors, which reduces the beamforming gain. In addition, the accuracy of the selected matrix is strongly related to the size of the dictionary: the larger the size, the better the accuracy matrix, but the longer waiting time is important. The size of the codebook is also proportional to the number of transmission antennas. Therefore, the codebook principle is only appreciated for a reduced transmission antenna number (thus a small amount of feedback). In [4], the authors proposed a new technique to use the codebook approach in a MMIMO context in the FDD mode. The proposed technique, based on the construction of a dictionary by the Grassmann manifold method and the minimization of the choral distance, allows a considerable reduction in the amount of feedback. However, the solution becomes ineffective when the number of antennas exceeds 64.

(ii) Antenna grouping beamforming (AGB) approach has also been proposed to solve this problem [5]. Here, the idea is to do the grouping of transmission antennas to have a very reduced representation of the number of excessive antennas by using the properties of the designed grouping patterns. This will allow a considerable reduction in the number of pilot symbols and the amount of feedback. However, this method is similar to that of the codebook but more complex to achieve. As a result, there are large quantization errors as the number of antennas becomes large.

(iii) In addition in this register of reducing feedback overhead, Shen et al. [6] proposed an adaptive angle-departure (AoD) subspace codebook for channel feedback in FDD massive MIMO systems. Their key idea is based on the observation that path AoDs vary more slowly than path gains. In the angular coherence time, by using the constant AoD information, the proposed AoD adaptive subspace codebook is able to quantify the channel vector more accurately. The authors show that the feedback overhead of the proposed codebook scales only linearly with a small number of dominant AoDs (paths) instead of the large number of antennas at the BS. Moreover, their proposal makes it possible to obtain a good quality of channel feedback, while requiring a low overhead.

(iv) In contrast to the previous techniques, other works have been based on the fact of making the compression of the CSI at the level of the users with specific compression techniques (arithmetic coding [7], Givens rotation [8], time domain quantization [8], and LZW compression [9]).

It should be noted that none of these works specify whether it is more optimal to return only the precoding matrix (partial CSI) or the channel matrix (full CSI). Indeed, in a classical MIMO system or MMIMO, the BS does not need the full channel matrix (full CSI) to beamform properly. Only the precoding matrix (partial CSI) is sufficient. Recall that the precoding matrix comes from the singular value decomposition of the full channel matrix. Therefore, in almost all works, to diagonalize the channel transmission and eliminate cochannel interference, users feedback only the precoding matrix after the estimated channel and singular value decomposition. However, it is quite possible for users to feedback the estimated channel matrix entirely, and in this case, it is up to the BS to do the singular value decomposition (SVD) of each subchannel to retrieve the precoding matrix relative to each subchannel.

In this paper, we will first make a comparative study between the full channel matrix and the precoding matrix resulting from the latter to choose the right candidate to be feedback. After that, we will make a comparative study of the application of compression techniques (GR, AC, and LZW) in a MMIMO context to know which of its techniques gives better performance in terms of compression ratio. Finally, we will propose two new approaches to reduce the downlink feedback overhead (GR&AC and GR&LZW) based on the combination of GR used in 802.11 and methods AC and LZW, respectively. The proposed techniques can reach 83% and 97% compression ratios, respectively. The rest of the paper is organized as follows: In Section 2, the system model is defined. Section 3 will present the comparison between the partial CSI and the full CSI. In Section 4, we will briefly describe the quantification of the Givens rotation, AC, and LZW compression in the MMIMO...
context. Section 5 will cover the proposals of improvement of the GR as well as the numerical results. Finally, Section 6 will serve as a conclusion.

2. The System Model

Consider a multiuser MIMO downlink channel operating in the FDD mode with $N_{BS}$ antennas at the base station (BS) and $K$ mobile stations (MS); each $K$ mobile station has $N_{MS_k}$ antennas for $k = 1, 2, \cdots, K$.

$N_{BS}$ is assumed to be greater than or equal to $\sum_{k=1}^{K} N_{MS_k}$: sum of all MS's antennas.

In Figure 1, the vector $x^k = [x_{1}, x_{2}, \cdots, x_{N_{MS_k}}]^T$ represents the transmitted data of the BS to the $k^{th}$ MS, where $k = 1, \cdots, K$.

A preprocessing is done on the BS side for each stream $x^k$, using the precoding matrices $P_k \in \mathbb{C}^{N_{BS} \times N_{MS_k}}$, where $k = 1, \cdots, K$. After DL transmitter preprocessing, the $N_{BS}$-component signal broadcast by the BS to the $K$ MSs can be expressed as

$$w = \sum_{k=1}^{K} w_k = P \times x,$$

where $w_k = P_k \times x^k$.

$$P = [P_1, P_2, \cdots, P_K],$$
$$x = [x^1, x^2, \cdots, x^K]^T.$$  \hspace{1cm} (2)

As shown in Figure 1, the frequency domain received signal $y_k$ of the $k^{th}$ MS for the considered OFDM subcarrier can then be expressed as

$$y_k = H_k w + n_k = H_k P x + n_k,$$

$$y_k = H_k P_k x^k + \sum_{i=1,i\neq k}^{K} H_k P_i x^i + n_k,$$  \hspace{1cm} (4)

where $n_k$ is a length-$N_{BS}$ additive white Gaussian noise with zero mean and a covariance matrix given by $\sigma^2 I_{N_{BS}}$, whereas $H_k$ is an ($N_{MS_k} \times N_{BS}$)-component channel transfer matrix connecting the $N_{BS}$ DL transmit antennas of the BS with the $k^{th}$ MS's $N_{MS_k}$ receive antennas, which can be expressed as

$$H_k = \begin{bmatrix} H_{1,1} & H_{1,2} & \cdots & H_{1,N_{MS_k}} \\ H_{2,1} & H_{2,2} & \cdots & H_{2,N_{MS_k}} \\ \vdots & \vdots & \ddots & \vdots \\ H_{N_{BS},1} & H_{N_{BS},2} & \cdots & H_{N_{BS},N_{MS_k}} \end{bmatrix}.$$  \hspace{1cm} (5)

A postprocessing on the postcoding matrix $P_k \in \mathbb{C}^{N_{MS_k} \times N_{BS}}$ is performed at each MS side to eliminate the multiuser interference (MUI). After that, the detected data on the considered OFDM subcarrier can be given by

$$\hat{x}^k = [\hat{x}_1, \hat{x}_2, \cdots, \hat{x}_{N_{MS_k}}].$$

The procedure described in this section is the same for all orthogonal frequency division multiplex (OFDM) subcarriers. To facilitate paper reading, in the following, we will consider a single OFDM subcarrier.

2.1. The SVD-Based Beamforming. Consider the SVD of the channel transfer matrix $H_k$. Now, let us perform the SVD of our matrix $H_k$ assuming that its rank is $N_k$: $\text{Rank}(H_k) = N_k$ and that $N_{BS} \geq \sum_{k=1}^{K} N_k$.

$$H_k = U_k [D_k 0] V_k^H = U_k [D_k 0] \begin{bmatrix} V_{k1}^H \\ V_{k1}^H \end{bmatrix} = U_k D_k V_{k1}^H,$$  \hspace{1cm} (6)

where $D_k \in \mathbb{C}^{N_k \times N_k}$ is diagonal and $U_k \in \mathbb{C}^{N_{BS} \times N_k}$ and $V_k \in \mathbb{C}^{N_{BS} \times N_k}$ are unitary matrices. In SVD-based beamforming, the transmit signal is precoded using the matrix $P_k = V_k^T$, where $^T$ stands for the transpose-conjugation.

In this context, Equation (3) becomes

$$y_k = U_k D_k V_{k1}^T P x + n_k.$$  \hspace{1cm} (7)

Let us now collect the $K$ received DL signal vectors $y_k$ into a vector $y = [y_1^T, y_2^T, \cdots, y_K^T]^T$. Then, according to Equation (7), it can be shown that the overall DL-received signal vector $y$ of the $K$ MSs can be expressed as

$$y = U D V_{1}^T P x + n,$$  \hspace{1cm} (8)

where

$$U = [U_{11}, U_{21}, \cdots, U_{K1}],$$
$$D = \text{diag}(D_{11}, D_{21}, \cdots, D_{K1}),$$
$$V_1 = [V_{11}, V_{21}, \cdots, V_{K1}],$$
$$n = [n_1^T, n_2^T, \cdots, n_K]^T.$$  \hspace{1cm} (9)

For ease of reading, it is assumed in the following that all MSs have the same number of antennas:

$$N_{MS_k} = N_{MS}$$ for $k = 1, 2, \cdots, K.$  \hspace{1cm} (10)

Based on Equation (8), it is not possible to perform separately the postprocessing which allows the transmitted data of each UE to be detected at the BS side. As shown in Figure 1, one postcoder matrix $P_k \in \mathbb{C}^{N_{MS} \times N_{BS}}$ is used to separate to separate the desired sign and the interfering signal at the $K$ different UEs.

Therefore, the postprocessing at each MS side can be done using the ZF solution:

$$P_k = (U_k^T U_k)^{-1} U_k^T.$$  \hspace{1cm} (11)
Then, Equation (8) becomes

\[ \hat{x} = D_l x + \tilde{n}. \]  

(12)

Since \( D_l \) is exactly diagonal, there is no MUI, and the MMIMO channel is reduced to \( N_{\text{BS}} \times K \) separate and independent SISO channels. However, the noise component \( \tilde{n} \) is colored by the postfilter and can be potentially enhanced.

Finally, the transmitted data from all UEs can be detected by performing a simple equalization process:

\[ \hat{x} = (D_l)^{-1} D_l x + (D_l)^{-1} \tilde{n} = x + \tilde{n}. \]  

(13)

2.2. Discussion. The SVD beamforming and ZF MUI cancellation presented in the two previous subsections can only be implemented if the DL CSI is known at both the BS and each UE. In the considered FDD system, the BS sends pilot sequences which are known at the MSs side. Based on the reception of these pilot sequences, each MS estimates its CSI. After that, each MS has to feed back the CSI to the BS. There are two possibilities:

(i) Each MS can feed back its channel matrix. However, this is not optimal for SVD beamforming scheme. Because in this case, the BS will be forced to perform a SVD decomposition to get its precoder matrix

(ii) To avoid this previous drawback, only the precoding matrices can be sent directly to the BS instead of the channel matrix

It is important to note that, in downlink MMIMO-OFDM system, each MS has to feed back its precoding matrix for each subcarrier. This feedback overhead can be essential and its reduction is necessary.

2.3. Channel Models for Massive MIMO. Let us talk a bit about the massive MIMO channel model. Indeed, the channel model plays an important role in system analysis as well as performance evaluation. One factor that critically determines the channel pattern is the configuration of an antenna array. Each measurement is made based on a given grating, and there are several typical antenna configurations that can be placed into three different groups: one-dimensional (1D), 2D, and 3D gratings. Each of these configurations as well as their size affects the properties of the channel by attributing different massive MIMO performance. For example, the distance between antenna elements essentially dictates the mutual coupling and correlation matrix, affecting the performance of massive MIMO. We can list two types of channel models found in the literature, namely correlation-based channel models and geometry-based stochastic channel models [10].

(i) Correlation-based models are mainly used for theoretical evaluations of system performance. This family of channel models can be sorted into three different types of channel models. These are, namely, the following:

(1) Rayleigh channel model: it assumes that there is no correlation between transmitting antennas or receiving antennas. Here, the elements of the fast fading matrix are i.i.d. complex Gaussian random variables. Moreover, the most important property of such a model is given by increasing the number of transmit antennas tends to infinity, which causes the rows of the transmit channel matrix \( H \) to become nearly orthogonal [2], which gives a property known as channel hardening and mitigates the impact of fast fading, which simplifies the complexity of scheduling schemes

(2) Correlation channel model: this model inserts antenna correlation due to antenna spacing and broadcast environments

---

**Figure 1:** Multiuser multiantenna downlink FDD with preprocessing and postprocessing.
(3) Mutual coupling channel model: as the number of antennas increases, the effect of mutual impedance also increases. Additionally, the load impedance and antenna impedance must also be characterized to reflect a more realistic channel

(ii) Geometry-based stochastic channel models are cluster-based models that define the propagation channel containing multiple clusters with different delays and power factors. In the case of a 3D channel model, the two strongest clusters are split into three subclusters with a fixed delay offset [10]. This model is used for practical evaluations of wireless communication systems

In this work, we use a channel model based on the correlation, more particularly that of Rayleigh.

3. Full CSI vs. Partial CSI

3.1. Simulation Settings. The considered simulation parameters are listed in Table 1. We consider several values of \( N_o \) in order to make simulations that concern a large part of the telecommunications systems using OFDM as long-term evolution (LTE) where \( N_o \) can be = 1024, IEEE802.11 where \( N_o \) is up to 512 or IEEE802.16 where \( N_o = 256 \).

3.2. Comparison. Figure 1 gives an overall view of the system with the feedback mechanism. In blue, it is the feedback of the precoding matrix (partial CSI) and in yellow that of the matrix of the estimated channel (full CSI). To determine which of these candidates is more beneficial to return, we have mapped the structures of the two matrices. Figures 2 and 3 show that the structures of the dies are almost similar to an MMIMO configuration but remain valid for all other configurations. This similarity is further proven with the application of the LZW compression algorithm. Thus, from Table 2, we can see that the compression ratios for the two matrices are slightly identical. Thus, instead of sending back the full CSI which requires the BS to do an SVD decomposition for each subchannel, it is better to send the partial CSI so that the BS will directly precode which will reduce the processing time.

In the rest of the document, we will consider that the users will return on the return path only the precoding matrix after SVD decomposition; thus, the feedback is partial. Therefore, we will focus on improving the method of the Givens rotation. Indeed, this method requires that the matrix be unitary.

Since the previous comparative study gives the reason to return only the precoding matrix (unitary) instead of the total matrix of the channel (nonunitary), it is then legitimate to look at how to improve the method of the Givens rotation which the method used in the IEEE 802.11/ac WLAN standard.

4. Givens Rotation, AC, and LZW

4.1. Givens Rotation Quantization. The Givens rotation (GR) is proposed by Kim and Aldana [11] to reduce the feedback overhead, exploiting the unitary property of the precoding matrix \( P_k \). The GR feedback compression method is adopted in the IEEE 802.11/ac WLAN standard, and its principle is to represent a unitary matrix in a special form with complex diagonal matrices as follows:

\[
P_k^* = \min(N_o - 1, N_i) \prod_{i=1}^{N_t} D_{k,i} \prod_{i=1}^{N_t} G_{k,i}(\psi_{k,i}) \times I_{N_o \times N_o},
\]

where

\[
D_{k,i} = \begin{bmatrix}
I_{i-1} & 0 & 0 & \cdots & 0 \\
0 & e^{\hat{\phi}_{k,i}} & 0 & \cdots & 0 \\
0 & 0 & \cdots & 0 & 0 \\
0 & 0 & 0 & e^{\psi_{k,N_{i+1}}} & 0 \\
0 & 0 & 0 & 0 & 1
\end{bmatrix},
\]

\[
G_{k,i} = \begin{bmatrix}
I_{i-1} & 0 & 0 & \cdots & 0 \\
0 & \cos(\psi_{k,i}) & 0 & \sin(\psi_{k,i}) & 0 \\
0 & 0 & I_{i-1} & 0 & 0 \\
0 & -\sin(\psi_{k,i}) & 0 & \cos(\psi_{k,i}) & 0 \\
0 & 0 & 0 & 0 & I_{N_{i-1}}
\end{bmatrix}
\]

The matrix \( D_{k,i} \) is an \( N_i \times N_i \) diagonal matrix, \( I_{N_o \times N_o} \) is an \( N_i \times N_i \) identity matrix, and \( G_{k,i} \) is an \( N_i \times N_i \) Givens rotation matrix.

According to Equations (14) and (15), the parameters to be determined in the GR compression are

\[
\psi_{k,i} \text{ for } i = 1, 2, \cdots, \min(N_t, N_o) \text{ and } i < l \leq N_t \\
\phi_{k,i} \text{ for } i = 1, 2, \cdots, \min(N_t, N_o) \text{ and } i < j \leq N_t - 1.
\]
Consider a unitary matrix $P$ of dimension $3 \times 3$. According to the GR method seen above (Equation (14)), $P$ can be decomposed as follows:

$$ P^\dagger = D_1(\psi_{21}) G_{21}^\dagger D_2(\psi_{32}) G_{32}^\dagger I_{3 \times 3}. $$  \hspace{1cm} (18)

The precoder matrix $P$ can then be reconstructed through the six parameters: $\psi_{21}, \psi_{31}, \psi_{32}, \phi_{1,1}, \phi_{1,2}$, and $\phi_{2,2}$.

Therefore, in a general way, instead of all elements of the matrix $P_k^i$, it is sufficient to consider the parameters $\psi_{k,li}$ and

| MMIMO-OFDM $N_{BS} \times N_{MS} \times N_o$ | Feedback amount without compression in bits | Compression rate of full CSI | Compression rate of partial CSI |
|---------------------------------------------|------------------------------------------|-------------------------------|-------------------------------|
| 64 $\times$ 4 $\times$ 128                 | 524288                                    | 80.4686%                      | 80.8355%                      |
| 64 $\times$ 4 $\times$ 256                 | 1048576                                   | 84.8101%                      | 85.2493%                      |
| 64 $\times$ 4 $\times$ 512                 | 2097152                                   | 89.3295%                      | 89.5338%                      |
| 64 $\times$ 4 $\times$ 1024                | 4194304                                   | 91.8439%                      | 91.9494%                      |
| 128 $\times$ 4 $\times$ 128                | 1048576                                   | 79.1023%                      | 80.8941%                      |
| 128 $\times$ 4 $\times$ 256                | 2097152                                   | 83.8996%                      | 85.0597%                      |
| 128 $\times$ 4 $\times$ 512                | 4194304                                   | 88.6253%                      | 89.5372%                      |
| 64 $\times$ 8 $\times$ 128                 | 1048576                                   | 79.2945%                      | 80.3640%                      |
| 64 $\times$ 8 $\times$ 256                 | 2097152                                   | 83.7158%                      | 84.6728%                      |
| 64 $\times$ 8 $\times$ 512                 | 4194304                                   | 88.6832%                      | 89.1703%                      |
| 64 $\times$ 8 $\times$ 1024                | 8388608                                   | 91.4242%                      | 91.7698%                      |
| 128 $\times$ 8 $\times$ 128                | 2097152                                   | 78.0467%                      | 81.0543%                      |

Figure 2: Full and partial CSI for $64 \times 4 \times 128$ MMIMO configuration.

Figure 3: Full and partial CSI for $128 \times 4 \times 128$ MMIMO configuration.
Note that these angle parameters can vary from 0 to $2\pi$ for $\psi_{k,li}$ and from 0 to $\pi/2$ for $\phi_{k,li}$.

Now, $\psi_{k,li}$ and $\phi_{k,li}$ can be quantized according to Equations (19) and (20): $\hat{\psi}_{k,li}$ and $\hat{\phi}_{k,li}$ represent the quantized angles.

$$
\hat{\psi}_{k,li} = \frac{Q_\psi (\pi/2)}{2^n} + \frac{\pi/2}{2^n + 1},
$$

(19)

where $Q_\psi = 0, 1, \cdots, 2^n - 1$ and $n_\psi$ the number of bits per angle $\psi$.

$$
\hat{\phi}_{k,li} = \frac{Q_\phi 2\pi}{2^n} + \frac{2\pi}{2^n + 1},
$$

(20)

where $Q_\phi = 0, 1, \cdots, 2^{n_\phi} - 1$ and $n_\phi$ is the number of bits per angle $\phi$.

Finally, the receiver feeds back the quantized parameters $Q_{\hat{\psi}_{k,li}}$ and $Q_{\hat{\phi}_{k,li}}$ to the transmitter which can recover the quantized precoder matrix $\hat{P}_k$ by using

$$
\hat{P}_k = \prod_{i=1}^{\min(N_f-1,N_c)} \left[ \hat{D}_{k,i} \prod_{l=i+1}^{N_t} \hat{G}_{k,l}^T(\hat{\psi}_{k,li}) \right] \times I_{N_t \times N_c},
$$

(21)

where

$$
\hat{D}_{k,i} = \begin{bmatrix}
I_{i-1} & 0 & 0 & \cdots & 0 \\
0 & e^{j\hat{\phi}_{k,li}} & 0 & \cdots & 0 \\
0 & 0 & \cdots & 0 & 0 \\
0 & 0 & 0 & e^{j\hat{\phi}_{k,li-1}} & 0 \\
0 & 0 & 0 & 0 & 1
\end{bmatrix},
$$

and

$$
\hat{G}_{k,li} = \begin{bmatrix}
I_{i-1} & 0 & 0 & \cdots & 0 \\
0 & \cos(\hat{\psi}_{k,li}) & 0 & \sin(\hat{\psi}_{k,li}) & 0 \\
0 & 0 & I_{i-1} & 0 & 0 \\
0 & -\sin(\hat{\psi}_{k,li}) & 0 & \cos(\hat{\psi}_{k,li}) & 0 \\
0 & 0 & 0 & 0 & I_{N-1}
\end{bmatrix}.
$$

(22)

When we assume that $N_c = N_r = N$, the total number of bits for each user $N_{FGR}$ can be calculated using the formula $N_{FGR} = N_r \times N_c \times N (N - 1)$, where $N_c$ represents the number of OFDM subcarriers. Based on the fact that the feedback overhead without compression is $N_n \times N_f \times 2 \times N^2$, the feedback overhead compression rate can be expressed by $(N + 1)/2N$. Unfortunately, this feedback overhead compression rate decreases with the growth of antenna ports. This last disadvantage limits the performance of GR in MMIMO.

However, the GR remains an interesting method because, as mentioned previously, the SVD decomposition is only performed at each MS side. Therefore, to take advantage of this particularity of the GR approach, we propose its improvement in Section 5.

### 4.2. Arithmetic Compression

Figure 4 gives an overall view of the transmission model with arithmetic coding from the insertion of the pilot signs to the estimation of the channel to the preprocessing of precoding and feedback.

#### 4.2.1. Description of Encoding and Decoding Principles

Arithmetic coding is a statistical coding, that is to say that more than one character is represented, the less bits need to be encoded [12]. The principle of the algorithm is as follows: First, we begin by subdividing the interval $[0,1)$ in $M$ intervals:

$$
[F_x(i-1), F_x(i)), i = 1, \cdots, M \text{ and } F_x(0) = 0.
$$

(23)

Then we must make a mapping of the symbols of the alphabet that make up the sequence to compress to the intervals:

$$
a_k \leftrightarrow [F_x(i-1), F_x(i)) k = 1, \cdots, N \text{ and } F_x(0) = 0.
$$

(24)

So any real in the interval $[F_x(i-1), \cdots, F_x(i))$ may represent the symbol $a_k$.

To better understand arithmetic coding, we will use a simple example. Let us go back to our sequence to compress: “MIMOMISO.” The length of this sequence is 8. Table 3 defines the probabilities of the characters in our sequence as well as the relative intervals, respectively.

Once this is done, we can now apply the arithmetic coding algorithm defined previously and we get Table 4.

Thus, 0.2983979258125 is the arithmetic representation of the “MIMOMISO” message. In practice, any number between 0.2983979258125 and 0.2984185253125 represents the correct message.

Figure 5 shows the coding process.

Decoding is just as simple: knowing that the encoded message is in the interval $[0.25; 0.625)$; thus, the first letter is indeed an $M$. We then subtract the lower bound (for $M$), and we divide by the probability (for $M$); we thus obtain 0.1290611355 which is included in the interval $[0; 0.25)$, and so, the next letter will be an $I$. We continue to subtract the lower bound (for $I$) and divide by the probability (for $I$); this gives 0.516244542 which is in the interval $[0.25; 0.625)$, and therefore, the next letter will be an $M$, etc.

#### 4.2.2. Solving the Arithmetic Coding Overflow Problem

The two preceding algorithms give the classical arithmetic encoding and decoding. However, one of the main problems with arithmetic coding is the phenomenon of overflow: overflow and underflow. These are issues related to the precision of machines (hardware). To overcome this, the most
The used technique is that of scaling. The principle of this can be described as follows:

(i) Resizing interval: we increase the size of the interval, but we must preserve the information

(ii) Incremental encoding: transmission of the code portion by portion as the encoding progresses

(iii) We start from the observation of 3 possibilities concerning the interval during encoding:
   (1) Interval is included in the lower half: \([0, 0.5)\)
   (2) Interval is included in the upper half: \([0.5, 1)\)
   (3) The interval contains the center point 0.5

(iv) When the gap is in one of the halves, it will stay in that half throughout the encoding of the sequence

(v) The most significant bit (MSB) is 0 if the interval is in \([0, 0.5)\)

(vi) MSB is 1 if the interval is in \([0.5, 1)\)

(vii) So when the interval is in one of the halves, the most significant bit becomes known

(viii) Once the encoder and decoder know which part we are in, we can ignore the other half

(ix) To implement the algorithm on a finite precision system, which is our case, we can resize the half in question to regain the interval \([0, 1)\) in full. This will result in the loss of information on MSB, but it does not matter because we have already sent the MSB. We continue the coding, and each time we are in one half, we send a bit. This is an incremental coding

(1) If \((high < 0.5)\)
   (i) Transmit bit 0 and add a 0 to the \(p\) number of pending bits
   (ii) \(Low = 2 \times low\)
   (iii) \(High = 2 \times high\)

(2) If \((low > 0.5)\)
   (i) Transmit bit 1 and add a 0 to the \(p\) number of pending bits
   (ii) \(Low = 2 \times (low - 0.5)\)
   (iii) \(High = 2 \times (high - 0.5)\)
If $low > 0$:

(i) Add a 1 to the $p$ number of pending bits

(ii) Low = 2 \((low - 0.25)\)

(iii) High = 2 \((high - 0.25)\)

Note that the interval is scaled (zoomed) for each output bit. This multiplication by 2 means moving the binary point one position to the right. Sometimes it is easier to work with integers than to work with comma numbers. To apply arithmetic coding to integers, just do the following:

(i) Use symbol frequencies instead of probability

(ii) Replace \([0, 1)\) by \([0, 2^{k-1} - 1)\)

(iii) Replace 0.5 by \(2^{k-1} - 1\)

(iv) Replace 0.25 by \(2^{k-2} - 1\)

(v) Replace 0.75 by \(3 \times 2^{k-2} - 1\)

Here is the formula to calculate the following interval:

\[
\begin{aligned}
\text{high} &= \text{low} + (\text{rangecum}[\text{symbol}] / \text{total freq}) - 1, \\
\text{low} &= \text{low} + (\text{rangecum}[\text{symbol} - 1] / \text{total freq}).
\end{aligned}
\]

Thus, Algorithms 1 and 2 become Algorithms 3 and 4, respectively: First we ask: \(\text{First}_\text{qtr} = 2^{k-1} - 1, \text{half} = 2^{k-2} - 1, \text{and} \ \text{Third}_\text{qtr} = 3 \times 2^{k-2} - 1.\)

4.3. LZW Compression. Among the most powerful data compression techniques, we can mention the compression algorithms based on the use of a dictionary. These are lossless data compression techniques that uses a dictionary with a predetermined state, but its value may be changed during the encoding or decoding process. One of the best-known dynamic dictionary-based compression algorithms is the LZW algorithm, which was created as a modification of the LZ78 algorithm [13]. It is a general-purpose compression algorithm in the sense that it can work with almost any type of data. LZW is a dictionary-based compression method [9] that associates a variable number of symbols (subsequences) with a code of fixed length. The algorithm works by traversing the input sequence to search a subsequence that is not in the dictionary. When such a subsequence is found, the index of the subsequence without the last element (that is, the longest subsequence of the dictionary) is extracted from the dictionary and sent to the output; then the new subsequence (including the last element) is added to the dictionary with the next available code. The last element entered is then used as the next starting point for searching for subsequences [9]. Increasingly long subsequences are stored in...
Algorithm 2: AC feedback overhead decoding.

1: Input: $A = a_i, i = 1, \cdots, N$; sequence to compress; $p_1, p_2, \cdots, p_M$: the symbol probabilities are known.
2: Output $A$: a binary number that represents sequence $A$.
3: (Cum = cumulative probability and coincides with the various terminals: $F_x$); $cum[0] = 0$
4: for $i = 1 : M$ do
5:       $cum[i] = cum[i - 1] + p_i$; \(\triangleright\) $M$ is the number of symbols in the alphabet (the number of intervals)
6: end for
7: $First_{qtr}=2^{k-1}$; $half=2^{k-2}$; $Third_{qtr}=3.2^{k-2}$;
8: bits_To_follow:= 0; \(\triangleright\) Number of bits to look at in order to know which part we are on
9: for $i = 1 : N$ do
10:      $Range = high - low + 1$;
11:      $high = (low + Range \cdot cum[a_i - 1]/cum[1]) - 1$;
12:      $low = low + Range \cdot cum[a_i]/cum[1]$;
13:      while $1$ do
14:          if $high < Half$ \(\triangleright\) the lower part
15:              transmission of bit "0"; $B = [B0]$;
16:          end if
17:          else if ($low >= Half$) \(\triangleright\) the high part
18:              transmission of bit "1"; $B = [B1]$;
19:          end if
20:          $low = low-Half$; $high = high-Half$;
21:          else if ($low > First_{qtr}$and(high < $Third_{qtr}$) \(\triangleright\) n the middle
22:              bits_To_follow =bits_To_follow + 1;
23:          end if
24:          $low = low-First_{qtr}$; $high = high-First_{qtr}$;
25:          $low =2 low$; $high =2 high$;
26:      end while
27:      break;
28: end for
29: return $A = a_1, a_2, \cdots, a_N$;

Algorithm 3: AC feedback overhead encoding.

1: Input: $val$: bit stream (compressed version of the sequence); $N$: the number of symbols to be decoded; $p_1, p_2, \cdots, p_M$: the symbol probabilities are known.
2: Output: the decoded sequence $A$.
3: Begin-decoding \(\triangleright\) (Cum = cumulative probability and coincides with the various terminals: $F_x$);
4: $cum[0]=p_1$;
5: for $i = 1 : N$ do
6:       $cum[i] = cum[i - 1] + p_i$; \(\triangleright\) the lower bound
7: end for
8: $high = 1$; \(\triangleright\) the upper bound
9: for $i = 1 : N$ do
10:      $Range = high - low$;
11:      $z = (val - low)/Range$;
12:      $j = 2$;
13:      while ($j < N$ and $(cum[j - 1] > zor > cum[j])$ \(\triangleright\) we find the index $j$ such this condition is fulfilled
14:          if $cum[j - 1] < z < cum[j]$ then
15:              $a_j = j$;
16:          else
17:              $j = j + 1$;
18:      end if
19: end for
20: $high = low + Range \cdot cum[j - 1]$;
21: $low = low + Range \cdot cum[j]$;
22: return $A = a_1, a_2, \cdots, a_N$;
the dictionary and made available for later encoding as single output values.

Figure 6 gives an overall view of the transmission model with LZW compressing from the insertion of the pilot signs to the estimation of the channel with the preprocessing of precoding and feedback.

The algorithm works best on data with repeated models. Like any compression algorithm, the LZW is defined by compression: obtained by replacing the sub-sequences with the codes associated with them in the dictionary and by decompression: performed by replacing sets of code words with their associated subsequences in the dictionary.

Thus, the coding and decoding algorithms are, respectively, presented in Algorithms 1 and 2, where BDC and DBC stand for, respectively, binary to decimal conversion and decimal to binary conversion.

In the LZW encoding and decoding, we can see the following:

(i) The LZW algorithm is very simple. In addition to this, the dictionary is designed at the same time as the encoding or decoding and does not need to be saved

(ii) The efficiency of the algorithm increases as the number of repetitive sequences in the input data increases

(iii) At each MS side, after the GR compression, all $N_o \times N_{MS_k} (N_{MS_k} - 1)$ output sequences of length $n_{b}$ bits are concatenated to get the AC or LZW input binary sequence

(iv) At the BS side, the AC or LZW decoding algorithm is used before the GR one to retrieve the precoding matrix $P_k$, for $k = 1, \cdots, K$

5. Improved GR

To improve the GR feedback compression, we have proposed two methods in which the AC or LZW algorithm is applied to the output quantized angles $\hat{\psi}$ and $\hat{\phi}$ which are sequences of length $n_{b}$ bits.

5.1. Validation of the Number of Quantizer Bits per Angle.

Figures 7–10 show the bit error rate (BER) as a function of signal-to-noise ratio (SNR) in the multiuser UL FDD MMIMO-OFDM context. The modulations are, respectively,
4-QAM and 16-QAM. Two schemes are considered, one with quantized feedback overhead compression by considering several values of \( n_b \) and one without compression. Several values of the parameters \( M \), \( N \), and \( K \) are also considered.

As we can see, the quantization error is negligible when \( n_b \) reaches 8 bit. The performances with or without quantization are the same in all considered configurations. Therefore, \( n_b = 8 \) is enough for the elimination of the quantization noise affect in the system performance. It is this number of quantizer bit \( n_b = 8 \) that has been adopted in the IEEE 802.11ac standard [14].

5.2. Compression Rate. The compression rates of conventional GR and the proposed GR with AC- (GR & AC) and
LZW- (GR & LZW) based feedback overhead compression for different MMIMO configurations and number of OFDM subcarriers (No) are listed in Table 5. Note that the number of quantizer bits is fixed to nb = 8 as previously validated.

As demonstrated in Section 4, the compression rate of conventional GR is \((N_{MS} + 1)/2N_{MS}\). That is why, in Table 5, for any MMIMO configuration and any number of OFDM subcarriers \(N_{sc}\), the compression rates are 56.25.5% and 62.5% when the numbers of antennas at each UE are, respectively, \(N_{MS} = 4\) and \(N_{MS} = 8\).

Unlike the conventional GR compression rate which only depends on the number of antennas per UE, the GR

---

**Figure 7:** BER performance in the DL FDD MMIMO-OFDM context \(M = 64, N = 4, K = 16\), and 256-QAM.

**Figure 8:** BER performance in the DL FDD MMIMO-OFDM context \(M = 64, N = 4, K = 16\), and 64-QAM.
LZW one takes advantage on the evolution of others parameters, especially the number of OFDM subcarriers. Because, the more \( N_\text{sub} \) increases, the more there is repetitive binary sequences of length \( n_b \). Accordingly, the compression rate increases considerably as the structure of LZW algorithm enables improvement of compression capacity when the repetitive sequences become large.

Then, observing the behavior of the GR & LZW compression rate in Table 5, we can note that, for the same MMIMO configuration (when \( N_{\text{BS}} \) and \( N_{\text{MS}} \) are fixed), the compression rate of the proposed GR & LZW increases with the number of OFDM subcarriers \( N_\text{sub} \).

In order to always push the compression rate, we have tried to improve the LZW algorithm that we have just applied on the GR method. To fully explain the new method proposed, GR & LZWM, we will be based on a simple example of compression with LZW.

Let us take the sequence “MIMOMISO” to compress with LZW. In this case, the incoming stream will correspond to the ASCII codes of the letters contained in the sequence.
For example, the ASCII code of B is 98. So, the ASCII version of our sequence is as follows:

To better understand LZW compression as well as the proposed modification, let us apply it to the text “MIMOMOSI” (see Tables 6 and 7 where the ASCII decimal codes of M, I, O, and S are 77, 73, 79, and 83). This example shows

(i) The LZW algorithm is very simple. In addition to this, the dictionary is designed at the same time as the encoding or the decoding and does not need to be saved.

(ii) The efficiency of the algorithm increases as the number of repetitive sequences in the input data increases.

5.3. LZW Modified. It is important to note that all outputs of the LZW encoder must be with the same number of bits which depends on the largest value in the dictionary. In the previous example, each output must be encoded with 9 bits. Indeed, it is the length of the number 259 which will be that of the other numbers of the output. However, apart from this number, the others can be coded with 8 bits because they do not exceed 256 = 2^8.

So, it will be interesting to see how to make the outputs of the LZW encoder individually coded with the minimum

Table 5: Compression rate of the conventional Givens rotation and the proposed Givens rotation with AC and LZW algorithms in different MMIMO-OFDM configurations. \(N_{BS}\), \(N_{MS}\), and \(N_o\), respectively, stand for the number of antennas at the BS, the number of antennas at each UE, the number of UEs, and the number of OFDM subcarriers.

| MMIMO OFDM \(M \times N \times N_o\) | Feedback amount in bits | Amount with GR in bits | Compression rate | Amount with amount with GR & LZW in bits | Compression rate | Amount with amount with GR & AC in bits | Compression rate |
|-------------------------------------|------------------------|-----------------------|-----------------|----------------------------------------|-----------------|---------------------------------------|-----------------|
| 64 \times 4 \times 128              | 524288                 | 196608                | 62.5%           | 26784                                  | 94.89%          | 80504                                 | 83.18%          |
| 64 \times 4 \times 256              | 1048576                | 393216                | 62.5%           | 38304                                  | 96.35%          | 179104                                | 83.15%          |
| 64 \times 4 \times 512              | 2097152                | 786432                | 62.5%           | 60480                                  | 97.12%          | 340472                                | 83.37%          |
| 64 \times 4 \times 1024             | 4194304                | 1572864               | 62.5%           | 86080                                  | 97.95%          | 679576                                | 83.17%          |
| 128 \times 4 \times 128             | 1048576                | 393216                | 62.5%           | 53568                                  | 94.89%          | 170920                                | 83.38%          |
| 128 \times 4 \times 256             | 2097152                | 786432                | 62.5%           | 76608                                  | 96.35%          | 356848                                | 83.29%          |
| 128 \times 4 \times 512             | 4194304                | 1572862               | 62.5%           | 120960                                 | 97.12%          | 679584                                | 83.25%          |
| 128 \times 4 \times 1024            | 8388608                | 3145728               | 62.5%           | 257730                                 | 96.93%          | 1423304                               | 83.26%          |
| 64 \times 8 \times 128              | 1048576                | 458752                | 56.25%          | 76296                                  | 92.72%          | 317552                                | 70.25%          |
| 64 \times 8 \times 256              | 2097152                | 917504                | 56.25%          | 109120                                 | 94.80%          | 624176                                | 70.31%          |
| 64 \times 8 \times 512              | 4194304                | 1835008               | 56.25%          | 155140                                 | 96.30%          | 1244040                               | 70.50%          |
| 64 \times 8 \times 1024             | 8388608                | 3670016               | 56.25%          | 203970                                 | 97.57%          | 2519568                               | 70.36%          |
| 128 \times 8 \times 128             | 2097152                | 1179648               | 56.25%          | 196420                                 | 90.63%          | 626720                                | 69.99%          |
| 128 \times 8 \times 256             | 4194304                | 1835008               | 56.25%          | 218240                                 | 94.80%          | 1249128                               | 70.68%          |
| 128 \times 8 \times 512             | 8388608                | 3670016               | 56.25%          | 310470                                 | 96.30%          | 2509384                               | 70.56%          |
| 128 \times 8 \times 1024            | 16777216               | 7430032               | 56.25%          | 487050                                 | 97.10%          | 4829072                               | 70.56%          |

Table 6: LZW encoding of “MIMOMOSI.”

| Steps | C | W | Code | Dictionary |
|-------|---|---|------|------------|
| 1     | M |   | M    | M is known |
| 2     | I | I | 77   | MI is encoded to 257 |
| 3     | M | M | 73   | IM is encoded to 258 |
| 4     | O | O | 77   | MO is encoded to 259 |
| 5     | M | M | 79   | OM is encoded to 260 |
| 6     | O | MO|      | MO is known |
| 7     | S | S | 259  | MOS is encoded to 261 |
| 8     | I | I | 83   | SI is encoded to 262 |
| 9     | I | 73 |      | I is known |

The output is 77, 73, 77, 79, 259, 83, and 73.

Table 7: LZW decoding of the code “77, 73, 77, 79, 259, 83, and 73.”

| Steps | OLD | CURRENT | Decoding char or string | Dictionary |
|-------|-----|---------|-------------------------|------------|
| 1     | 77  | I       | M                       | M is encoded to 257 |
| 2     | 73  | 73      | I                       | MI is encoded to 257 |
| 3     | 77  | 77      | M                       | IM is encoded to 258 |
| 4     | 79  | 79      | O                       | MO is encoded to 259 |
| 5     | 259 | 259     | MO                      | OM is encoded to 260 |
| 6     | 83  | 83      | S                       | MOS is encoded to 261 |
| 7     | 73  | 73      | I                       | SI is encoded to 262 |

The output is “MIMOMOSI.”

For example, the ASCII code of B is 98. So, the ASCII version of our sequence is as follows:

(i) The LZW algorithm is very simple. In addition to this, the dictionary is designed at the same time as the encoding or the decoding and does not need to be saved.

(ii) The efficiency of the algorithm increases as the number of repetitive sequences in the input data increases.

5.3. LZW Modified. It is important to note that all outputs of the LZW encoder must be with the same number of bits which depends on the largest value in the dictionary. In the previous example, each output must be encoded with 9 bits. Indeed, it is the length of the number 259 which will be that of the other numbers of the output. However, apart from this number, the others can be coded with 8 bits because they do not exceed 256 = 2^8.

So, it will be interesting to see how to make the outputs of the LZW encoder individually coded with the minimum
The number of bits required. To do this, we have adopted another demarcation from the LZW algorithm name LZWM.

(i) First, we have defined a word from the dictionary as a reserved word: for the initial dictionary (0 to 255), the number 255 is reserved. This number will serve as a flag to warn the decoder about the number of bits encoded. So our initial dictionary becomes 0 to 254.

(ii) The first word of the output is 255: to signal to the decoder that the number of coded bits is 9.

(iii) Each time we encode a word, before putting it in the output sequence, we check whether it is greater than $2^9 - 1$:

1. If so, we place in the output sequence $2^{9-1}$ to indicate that now the number of bits is 10.
2. And so on, until the end of the sequence is encoded.

(iv) Otherwise, we place the latter code in the output sequence.

5.3.1. Comparison between LZW and LZWM. Table 8 gives the LZWM encoding of our previous sequence. So in this example, the gain from the modification is not very visible (we even lose; Table 9). Indeed, the length of the sequence is very reduced; suddenly, there is no jump, i.e., going to 10 bits, 11 bits, and so on.

However, this article is part of a MMIMO context, which means that the sequences that we will be called upon to compress will take very large dimensions. Thus, this gain could become interesting. Moreover, the study allowed us to define the minimum and maximum gains of this method compared to the classical one.

To generalize, let us set $N_d$ the number of starting bits and $N_f$ the final number of bits. For $N_d = 8$ and $N_f = 12$, we have the following:

(i) At a minimum, we have $(2^{9-1})(12 - 9) - 8$ before going to 10 bits. 8 is the number of bits inserted so the flag.

(ii) At a minimum, we have $(2^{10-9})(12 - 10) - 9$ before going to 11 bits.

(iii) At least we have $(2^{11-10})(12 - 11) - 10$ before going to 12 bits.

In general, we will have a gain:

(i) At least we have

$$G_{\text{min}} = \sum_{n=N_f+1}^{N_f} (2^n - 1)(N_f - n + 1) - n$$

which gives

$$G_{\text{min}} = \sum_{n=N_f+1}^{N_f} (2^n - 1)(N_f - n - 1 - n)$$

Table 8: Encoding of “MIMOMOSI” by LZWM.

| Steps | CW | Code | Dictionary | Number of coded bits |
|-------|----|------|------------|---------------------|
| 1     | M  | 255  | $M$ reserved | 8                   |
| 2     | M  | M    | $M$ known   |                     |
| 3     | I  | I    | 77          |                     |
| 4     | O  | O    | 77          |                     |
| 5     | M  | M    | 79          |                     |
| 6     | O  | MO   | 79          |                     |
| 7     | S  | S    | 259         |                     |
| 8     | I  | I    | 83          |                     |
| 9     | I  | I    | 73          |                     |

The output is 77, 73, 77, 79, 259, 83, and 73.

Table 9: Number of bits for LZW and LZWM of the code “77, 73, 77, 79, 259, 83, and 73.”

|         | LZW                   | LZWM                   |
|---------|-----------------------|-----------------------|
| Output sequence | 77, 73, 77, 79, 259, 83 and 73 | 77, 73, 77, 79, 259, 83 and 73 |
| Number of bits | $9 + 9 + 9 + 9 + 9 + 9 = 63$ bits | $8 + 9 + 9 + 9 + 9 + 9 + 9 + 9 = 71$ bits |
\[
G_{\text{max}} = \sum_{n=N_\text{f}+1}^{N_f-N_\text{f}-1} \left( (2^n - 1) \left( (N_f - n - 1) - n \right) \right) 
\]  

(ii) Likewise, at most

\[
N_f = N_\text{f} - N_\text{fd} - 1 
\]

Therefore, let us keep the same previous simulation parameters, and we consider the case where we have a single user and a multipath MMIMO channel; the comparison of the two algorithms gives Table 10.

Through Table 10 confirms that the modified version of the LZW algorithm has a gain of about 2% compared to the classic version, this percentage gain may seem small, but let us just remember that we are in a MMIMO context, so 2% of the data handled is necessarily significant. In addition, we will see later that if we apply these two algorithms to the GR method, the difference in favor of LZWM becomes more important.

Thus, Table 11 gives the comparison of the application of the new method proposed for some MMIMO-OFDM configurations.

In summary, we can say that excessive feedback overload is a major handicap in FDD MMIMO systems. Therefore, this excess downlink feedback must be compressed by sophisticated algorithms. Significant efforts are being made in this direction, and several solutions are proposed. We have proposed two lossless compression solutions based on the GR & LZW compression algorithm and a derivative of the latter: GR & LZWM. These algorithms can be considered evolutions of the Givens rotation method which is adopted in the WLAN standard IEEE 802.11/c. The results of the simulations show that the proposed methods can achieve more than 95% and 98% compression ratios, respectively.

6. Conclusion

Multiuser MMIMO-OFDM systems operating in the FDD mode are a promising solution to deliver unprecedented spectral efficiency in future wireless communication systems. Several powerful preprocessing and postprocessing techniques are proposed in the literature to both diffuse and eliminate potential multiuser interference.

However, since UL and DL CSI are mandatory on each side of the transceiver, MMIMO-OFDM systems in FDD mode require a large amount of feedback from the CSI. Therefore, limited feedback is now an important research topic.

In this work, we have proposed three improvements to the conventional Givens rotation feedback compression which is adopted in the IEEE 802.11 standard. The first GR & AC proposal is based on a lossless compression technique, the arithmetic coding. By opting for arithmetic coding with resizing to avoid the overflow problem and to be able to transmit as the compression principle unfolds, the proposed GR & AC algorithm has achieved a compression rate of up to 83%.

The second GR & LZW method is based on the LZW compression technique which is also a lossless compression method. With the LZW-based solution, the simulation results show that up to 95% compression ratio can be achieved.

### Table 10: Comparisons between LZW and LZWM.

| Number of subcarriers | Feedback amount without compression in bits | Feedback amount with GR & LZW in bits | Compression rate | Feedback amount with GR & LZWM in bits | Compression rate |
|-----------------------|--------------------------------------------|--------------------------------------|-----------------|----------------------------------------|-----------------|
| 16 x 16               | 128                                        | 256                                  | 512             | 1024                                   |                 |
| LZW                   | 80.5752%                                   | 84.7580%                             | 89.3175%        | 91.8718%                               |                 |
| LZWM                  | 81.8413%                                   | 86.1474%                             | 90.0122%        | 92.6036%                               |                 |

### Table 11: Compression rate of LZW and LZWM algorithms in different MMIMO-OFDM configurations. \(N_{\text{BS}}, N_{\text{UEs}}, \text{and } N_{\text{f}}, \text{respectively, stand for the number of antennas at the BS, the number of antennas at each UE, the number of UEs, and the number of OFDM subcarriers.}

| MMIMO-OFDM \(M \times N \times N_{\text{f}}\) | Feedback amount without compression in bits | Feedback amount with GR & LZW in bits | Compression rate | Feedback amount with GR & LZWM in bits | Compression rate |
|---------------------------------------------|--------------------------------------------|--------------------------------------|-----------------|----------------------------------------|-----------------|
| 64 x 4 x 128                               | 524288                                     | 35664                                | 93.20%          | 7373                                   | 98.59%          |
| 64 x 4 x 256                               | 1048576                                    | 55224                                | 94.73%          | 10878                                   | 98.96%          |
| 64 x 4 x 512                               | 2097152                                    | 78611                                | 96.25%          | 15817                                   | 99.24%          |
| 64 x 8 x 128                               | 1048576                                    | 90051                                | 91.42%          | 27282                                   | 97.39%          |
| 64 x 8 x 256                               | 2097152                                    | 138628                               | 93.38%          | 39870                                   | 98.09%          |
| 64 x 8 x 512                               | 4194304                                    | 197288                               | 95.29%          | 58895                                   | 98.59%          |
| 128 x 8 x 256                              | 4194304                                    | 296805                               | 92.92%          | 58895                                   | 98.59%          |
| 128 x 8 x 512                              | 8388608                                    | 422475                               | 94.96%          | 86182                                   | 98.97%          |
| 128 x 8 x 1024                             | 16777216                                   | 640704                               | 96.18%          | 126957                                   | 99.24%          |
achieved. Finally, this second proposition has in turn been improved by modifying the coding principle of the LZW algorithm: GR & LZWM (modified). And this proposal brings improvements over the second in terms of compression rates which can reach 98%.

All the proposed algorithms brought a feedback reduction gain compared to that of GR used in the IEEE 802.ac standard and presented the same performance in terms of BER if the number of quantization bits became greater than or equal to 8.

Data Availability
In this work, there is underlying data.

Conflicts of Interest
The authors declare that they have no conflicts of interest.

References
[1] T. L. Marzetta, "How much training is required for multiuser MIMO?,” in IEEE Asilomar Conference on Signals, Systems, and Computers, Proc., 2006.
[2] T. L. Marzetta, "Noncooperative cellular wireless with unlimited numbers of base station antennas," IEEE Transactions on Wireless Communications, vol. 9, no. 11, pp. 3590–3600, 2010.
[3] J. Jose, A. Ashikhmin, T. L. Marzetta, and S. Vishwanath, "Pilot contamination and precoding in multi-cell TDD systems," IEEE Transactions on Wireless Communications, vol. 10, no. 8, pp. 2640–2651, 2011.
[4] J. Choi, Z. Chance, D. J. Love, and U. Madhow, "Noncoherent trellis coded quantization: a practical limited feedback technique for massive MIMO systems," IEEE Transactions on Communications, vol. 61, no. 12, pp. 5016–5029, 2013.
[5] B. Lee, J. Choi, J. Seol, D. J. Love, and B. Shim, "Antenna grouping based feedback reduction for FDD-based massive MIMO systems," in Proc. IEEE ICC, pp. 4477–4482, 2014.
[6] W. Shen, L. Dai, B. Shim, Z. Wang, and R. W. Heath, "Channel feedback based on AoD-adaptive subspace codebook in FDD massive MIMO systems," IEEE Transactions on Communications, vol. 66, no. 11, pp. 5235–5248, 2018.
[7] K. Mittal, S. Pathania, P. Reddy, and D. Rawal, "Channel state information feedback overhead reduction using arithmetic coding in massive MIMO systems," in 2016 3rd International Conference on Signal Processing and Integrated Networks (SPIN), pp. 328–331, Noida, 2016.
[8] P. Ndiaye, M. Diallo, M. Mbaye, and I. Diop, "Feedback overhead comparison between frequency SVD and time domain beamforming in WLAN context," in I2018 International Conference on Advanced Communication Technologies and Networking (CommNet), 2018.
[9] P. Ndiaye, M. Diallo, M. Mbaye, and M. R. Seye, "Efficient lossless feedback compression for FDD massive MIMO," in International Wireless Communications Mobile Computing Conference (IWCMC), pp. 579–585, 2019.
[10] K. Zheng, S. Ou, and X. Yin, "Massive MIMO channel models: a survey," International Journal of Antennas and Propagation, vol. 2014, Article ID 848071, 10 pages, 2014.
[11] J. Kim and C. Aldana, "Efficient feedback of the channel information for closedloop beamforming in WLAN," in IEEE VTC-spring, pp. 2227–2230, 2006.
[12] I. H. Witten, R. M. Neal, and J. G. Cleary, "Arithmetic coding for data compression," ACM, vol. 30, no. 6, pp. 520–540, 1987.
[13] J. Ziv and A. Lempel, "Compression of individual sequences via variable-rate coding," IEEE Transactions on Information Theory, vol. 24, no. 5, article 530536, pp. 530–536, 1978.
[14] I. C. Studer and G. Durisi, "Quantized massive MU-MIMO-OFDM uplink," IEEE Transactions on Communications, vol. 64, no. 6, pp. 2387–2399, 2016.