Non-negative vector optimization: application to precoding for massive antennas

Florian Polster-Prieto, Inbar Fijalkow
ETIS, UMR 8051 / CY Cergy Paris University, ENSEA, CNRS, F-95000 Cergy, France
florian.polster-prieto@ensea.fr

Abstract - We present the mathematical approaches and implementations of precoding architectures for a massive multiuser (MI) multiple-input multiple-output (MIMO) wireless system. It has been proven in [6] that the Quantized Zero-Forcing (Quantized ZF) precoder is already able to achieve the optimal symbol error rate for a high enough $\gamma = \frac{K}{M}$ ($\gamma > 10$). The proposed algorithms are thus provided as an improvement in the case where ZF fails in low γ regimes and not a replacement. Both work hand in hand as they use iterative optimization and need the ZF precoded vector as an initialization. Our simulations however show that our proposed methods perform worse than the method proposed in [1], due to various reasons analized later on.

I. Introduction

Massive MIMO [5] systems use a high amount of antennas at the Base Station (BS) to transmit wirelessly to a smaller amount of separate users. In such a system, the number of antennas M is much higher than the number of users K.

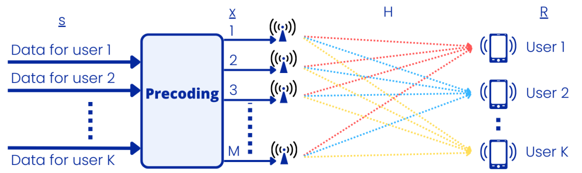

Fig. 1. System Model

This dimension increase (input space to transmitter space) is made trough a precoding step, where the input vector $\underline{\mathbf{s}}$ is multiplied with the precoding matrix $\mathbf{P}$. The transmitted vector is thus the quantization of this result, with the quantization step denoted as

$$\mathbb{Q}(z) = \text{sign}(\Re(z)) + i\ \text{sign}(\Im(z))$$

Where $\Re(z)$ and $\Im(z)$ denote respectively the real and imaginary part of z. The channel is thus modeled as

$$\underset{K\times 1}{\underline{\mathbf{R}}} = \underset{K\times M}{\mathbf{H}} \times \mathbb{Q}(\underset{M\times 1}{\underline{\mathbf{x}}}) = \underset{K\times M}{\mathbf{H}} \times \mathbb{Q}(\underset{M\times K}{\mathbf{P}} \times \underset{K\times 1}{\underline{\mathbf{s}}})$$

in the case of infinite signal to noise ration.

In this paper, we will focus on 1-bit precoding optimization for massive MIMO, implying that every signal sent is in a Q-PSK constellation. This implies that the quatization functions output is a Q-PSK symbol, such as:

$$\forall z, \quad \mathbb{Q}(z) = \pm\frac{1}{\sqrt{2}} \pm j\frac{1}{\sqrt{2}}$$

Here are the notations we use throughout the paper :

TABLE I
Notations used in this paper

Notation	Description
$\underline{\mathbf{x}}$	Vector x
$\mathbf{x}_i, \forall i \in [\![1;K]\!]$	i-th element of the vector x of size K
$\mathbf{A}$	Matrix A
$\mathbf{A}_{i,j}, \forall i \in [\![1;K]\!]$, $\forall j \in [\![1;M]\!]$	Element i,j of the matrix A of size $K \times M$
$\mathbb{Q}(z)$	Quantized vector z
$\text{sign}(x)$	Sign of x
$\Re(z)$	Real part of z
$\Im(z)$	Imaginary part of z
$\text{Diag}(\mathbf{s_1}, \ldots, \mathbf{s_K})$	Diagonal matrix of size $K \times K$ containing the elements of $\underline{\mathbf{x}}$
$\underline{\mathbf{d}} \otimes \underline{\mathbf{x}}$	Element wise multiplication of vectors $\underline{\mathbf{d}}$ and $\underline{\mathbf{x}}$
$\mathbf{A}^T$	Matrix transpose of A
$\mathbf{A}^H$	Hermitian transpose of A

II. State of the Art

A. Quantized Zero-Forcing

The principal algorithm for 1-bit precoding for massive MIMO is the Quantized Zero-Forcing, as shows [6] by analyzing it and proving that it achieves asymptotically (in M and K for $\gamma = \frac{M}{K} > 10$) the best symbol error rate possible. ZF consists in computing the pseudo inversion of the channel matrix $\mathbf{H}$, such that

$$\mathbf{P} = \mathbf{H}^H(\mathbf{H}\mathbf{H}^H)^{-1}$$

The quantization function $\mathbb{Q}$ is then applied to $\mathbf{P}\underline{\mathbf{s}}$ to compute $\underline{\mathbf{x}} = \mathbb{Q}(\mathbf{P}\underline{\mathbf{s}})$.

B. C2PO

In the case of cases of smaller γ, Quantized ZF is outperformed by the method proposed in [1], [3]. This method models the problem as an optimisation problem and introduces an amplification of the input vector $\underline{\mathbf{s}}$

$$\{\underline{\hat{\mathbf{x}}}, \hat{\alpha}\} = \arg\min_{\underline{\mathbf{x}},\alpha} \|\alpha\underline{\mathbf{s}} - \mathbf{H}\underline{\mathbf{x}}\|_2^2$$

where α is a complex coefficient. For a given $\underline{\mathbf{x}}$ the optimal $\hat{\alpha}$ value can be computed as $\hat{\alpha} = \underline{\mathbf{s}}^H\mathbf{H}\underline{\mathbf{x}}/\|\underline{\mathbf{s}}\|_2^2$. This lets us rewrite the equation as an optimisation problem solely over $\underline{\mathbf{x}}$, where $\underline{\mathbf{x}}$ belongs to a finite constellation (Q-PSK in our case):

$$\underline{\hat{\mathbf{x}}} = \arg\min_{\underline{\mathbf{x}}} \|\mathbf{A}\underline{\mathbf{x}}\|_2^2$$

where $\mathbf{A} = (\mathbf{I}_K - \underline{\mathbf{s}}\underline{\mathbf{s}}^H/\|\underline{\mathbf{s}}\|_2^2)\mathbf{H}$. A concave regularizer $-\frac{\delta}{2}\|\underline{\mathbf{x}}\|_2^2$ is then added to avoid the all zero solution. To solve this problem, a forward-backward splitting [4], [2] (FBS) algorithm is used, called C2PO. FBS is an efficient method for convex optimization problems, and is thus not guaranteed to converge to an optimal solution for this problem. The experimental results however show excellent performances in [1] and [3].

```
Algorithm 1: C2PO
  Input: $\mathbf{s}, \mathbf{H}, P, \tau^{(t)}, \delta$
  Initialize $\mathbf{x}^{(0)} = \mathbf{H}^H\mathbf{s}$
  Compute $\rho^{(t)}$
  for $t \in [1, t_{max}]$ do
      $\mathbf{z}^{(t)} = \mathbf{x}^{(t-1)} - \tau^{(t)}\mathbf{A}^H\mathbf{A}\mathbf{x}^{(t-1)}$
      $\mathbf{x}^{(t)} = \text{prox}_g(\mathbf{z}^{(t)}; \rho^{(t)}, \xi)$
  end
  Quantize the output $\mathbf{x}^{(t_{max})}$ to the used alphabet
  Output: $\mathbf{x}^{(\mathbf{tmax})}$
```

Where $\rho = \frac{1}{1-\delta\tau}$, $\xi = \sqrt{\frac{P}{2M}}$ with P an instantaneous power constraint over the transmitted signal $\mathbf{x}$, and

$$\text{prox}_g = \text{clip}(\rho\Re(\mathbf{z}), \xi) + j\text{clip}(\rho\Im(\mathbf{z}), \xi)$$

$$\text{clip}(z, \xi) = \min(\max(z_i, -\xi), \xi)$$

The original implementation of this algorithm[3] used $\tau^{(t)}$ and $\rho^{(t)}$ as constant values due to the complexity of tunning each parameter for each iteration. The initialization here uses the maximal-ratio transmission (MRT) solution, but the ZF initialization is also valid (tested briefly during our simulations, but dropped for the final results). The authors of this method also implemented a neural-network to learn the hyperparameters of C2PO in [1].

C. Problem settings

The main objective of this paper is to improve the performances of the Original C2PO by relaxing the space of the amplification vector. To do so, we first replace the complex amplification coefficient α used in [1], [3] by a vector containing real and positive coefficients:

$$\alpha\underline{\mathbf{s}} \rightarrow \begin{pmatrix} \mathbf{d}_1 \\ \vdots \\ \mathbf{d}_k \end{pmatrix} \otimes \underline{\mathbf{s}}$$

We noticed however that restraining the amplification to real coefficient yields worse parameters than the original C2PO, and we thus added a shared complex phase to each element of the vector $\underline{\mathbf{d}}$, as a non constrained optimization of $\underline{\mathbf{d}}$ is not possible (the problem writes as $\underline{\mathbf{x}^*} = \|\underline{0}\|_2^2$).

III. Simulation settings

In this section, we present the basis of our evaluation of the implemented algorithms, and compared to some state of the art algorithms.

A. Construction of the datasets

To evaluate our algorithms, we built different datasets for different K and M values containing 1000 different pairs of $\underline{\mathbf{s}}$ and $\mathbf{H}$. As the proposed algorithms are not meant to replace the Quantized Zero-Forcing (Quantized ZF), but to take over when Quantized ZF fails, we construct our dataset with pairs of $(\mathbf{H}, \underline{\mathbf{s}})$ on which the Quantized ZF fails (i.e. where the quantized output vector $\mathbb{Q}(\underline{\mathbf{R}}) = \mathbb{Q}(\mathbf{H}\mathbb{Q}(\mathbf{P}\underline{\mathbf{s}}))$ is different from the input vector $\underline{\mathbf{s}}$). Due to the asymptotic behavior of Quantized ZF, the percent of pairs on which Quantized ZF fails decreases when γ grows. We provide bellow a table representing the number of iterations needed to fill our dataset, while making sure that there where no duplicates. This means that the ratio will not exactly represent the percentage of the pairs where Quantized ZF fails, as we could not save the whole drafted pairs to compare them with the new ones (due to the size of the resulting dataset). However, it will be a good approximate if we consider that drafting twice the exact same pair with the from randomly generated numbers is unlikely for high enough values of K and M.

TABLE II
Number of iterations for the dataset construction to obtain 1000 settings

γ	K	M	Realisations
4	5	20	9617
4	25	100	2566
5	5	25	26042
5	20	100	6634
10	5	50	2 293 382
10	10	100	1 203 298

For each individual pair, $\underline{\mathbf{s}}$ is randomly chosen from a Q-PSK constellation while $\mathbf{H}$ is drafted from a complex white gaussian noise.

B. Hyperparameters

All the algorithms are implemented equally, meaning they first compute the Quantized ZF method, use the MRT as initialization if it fails (which will be the case for our dataset), and finally compares its output to the Quantized ZF output to keep the best option. The MRT initialization is used as it yields better results than both ZF (as it is a local optimum of the function and thus does not improve the performances of Quantized ZF) and Quantized ZF (empirical results). For the simulations, we fixed t_{max} to 100 for the C2PO variants for good measure, but only a dozen iterations is sufficient to obtain better results than Quantized ZF. δ and τ have been set by tuning the hyperparameters by hand to obtain the seeminlgy best results, but an hyperparameter optimization would improve our results as we notice a drastic modification of the results with small changes. We thus chose:

TABLE III
HYPERPARAMETERS USED

Method	t_{max}	τ	δ
Original C2PO [1], [3]	100	10^{-3}	1
Real vector C2PO	100	10^{-9}	1
Sigmoid C2PO	100	10^{-9}	100
Shared phase C2PO	100	10^{-2}	1

We will developp the equation of the Real vector, Sigmoid and Shared Phase C2POs in the sequel (respectively in section IV-A, section IV-E and section V).

C. Simulation metric

The metric we use is the symbol error rate (SER), defined for a Q-PSK vector of size K as

$$\mathrm{SER}(\underline{s},\underline{r}) = \frac{\|1_{\underline{s}-\underline{r}}\|_1}{K}$$

where $1_{\underline{s}-\underline{r}}$ is a vector whose i-th element is 0 if $s_i - r_i = 0$ and 1 if $s_i \neq r_i$.

Our plots represent the SER (between the input vector $\underline{\mathbf{s}}$ and the quantized output $\mathbb{Q}(\underline{\mathbf{R}})$) in function of the signal to noise ratio (SNR), $\frac{2\rho M}{\sigma_n^2}$, with ρ and σ_n^2 defined through

$$\underline{\mathbf{R}} = \sqrt{\rho}\mathbf{H}\underline{\mathbf{x}} + \underline{\mathbf{n}}$$

where $\underline{\mathbf{n}}$ is a vector of independent Gaussian noise of variance σ_n^2.

IV. First proposed method

A. Non-negative vector optimization

Our first approach of the problem was to amplify separately each element of the input vector $\mathbf{s}_i$ with a real and positive coefficient d_i. This renders the problem to the following equation

$$\{\hat{\underline{\mathbf{x}}}, \hat{\underline{\mathbf{d}}}\} = \arg \min_{\underline{\mathbf{x}} \in \mathcal{S}^M, \mathbf{d_i} \geq 0} \|\mathbf{H}\underline{\mathbf{x}} - \underline{\mathbf{d}} \otimes \underline{\mathbf{s}}\|_2^2$$

where $\otimes$ denotes the element-wise product, and can also be written as

$$\underline{\mathbf{d}} \otimes \underline{\mathbf{s}} = \mathbf{D}\underline{\mathbf{s}} = \mathbf{S}\underline{\mathbf{d}}$$

with $\mathbf{D}$ and $\mathbf{S}$ defined as

$$\mathbf{D} = \mathrm{Diag}(\mathbf{d_1}, \ldots, \mathbf{d_K}),\ \mathbf{S} = \mathrm{Diag}(\mathbf{s_1}, \ldots, \mathbf{s_K})$$

B. Optimization of the "positive" vector

To solve this problem, we first compute $\hat{\underline{\mathbf{d}}}$ using convex analysis tools

$$\hat{\underline{\mathbf{d}}} = \arg \min_{\mathbf{d}_i \geq 0} \|\mathbf{D}\underline{\mathbf{s}} - \underline{\mathbf{R}}\|_2^2$$

The function to optimize can be expressed as

$$\begin{aligned} &\|\underline{\mathbf{d}} \otimes \underline{\mathbf{s}} - \underline{\mathbf{R}}\|_2^2 \\ &= \sum_{i=1}^{K} (\mathbf{d}_i \Re(\mathbf{s}_i) - \Re(\mathbf{R}_i))^2 + (\mathbf{d}_i \Im(\mathbf{s}_i) - \Im(\mathbf{R}_i))^2 \\ &\text{subject to } \mathbf{d}_i \geq 0 \end{aligned}$$

We then minimize this expression to using the Karush-Kuhn-Tucker conditions

$$\begin{aligned} f(\mathbf{d}_1, \ldots, \mathbf{d}_K) = &\sum_{i=1}^{K} (\mathbf{d}_i \Re(\mathbf{s}_i) - \Re(\mathbf{R}_i))^2 \\ &+ (\mathbf{d}_i \Im(\mathbf{s}_i) - \Im(\mathbf{R}_i))^2 + \lambda_i g_i(\mathbf{d}_i) \end{aligned}$$

with $\lambda_i \in \mathbb{R}$ and $g_i(\mathbf{d}_i) = -\mathbf{d}_i$.
The optimal solution is characterised by

$$\begin{cases} \left(\mathrm{grad}_{\underline{\mathbf{d}}}\Big(f(\underline{\mathbf{d}})\Big)\Big|_{\underline{\mathbf{d}}^*}\right)_i = \underline{0} \\ -\lambda_i \mathbf{d}_i = 0 \end{cases}$$

Solving $\frac{df(\underline{\mathbf{d}})}{d\mathbf{d}_n} = 0$ for each n gives us

$$\mathbf{d}_n = \frac{\frac{\lambda_n}{2} + \Re(\mathbf{s}_n)\Re(\mathbf{R}_n) + \Im(\mathbf{s}_n)\Im(\mathbf{R}_n)}{\Re(\mathbf{s}_n)^2 + \Im(\mathbf{s}_n)^2}$$

$\underline{\mathbf{s}}$ being in a QPSK alphabet (i.e. the transmitted values are on the unit circle), we have $\Re(\mathbf{s}_n)^2 + \Im(\mathbf{s}_n)^2 = 1$. Additionally, we have

$$-\lambda_n \mathbf{d}_n = 0 \implies \begin{cases} \lambda_n = 0 \ \ \& \ \ -\mathbf{d}_n \leq 0 \\ \text{or} \\ \mathbf{d}_n = 0 \ \ \& \ \ \lambda_n \geq 0 \end{cases}$$

We finally have $\mathbf{d}_n = \Big(\Re(\mathbf{s}_n)\Re(\mathbf{R}_n) + \Im(\mathbf{s}_n)\Im(\mathbf{R}_n)\Big)^+$ with $(x)^+ = \max(x, 0)$
Or, in vector form:

$$\begin{aligned} \underline{\mathbf{d}} &= \Big(\Re(\underline{\mathbf{s}}) \otimes \Re(\underline{\mathbf{R}}) + \Im(\underline{\mathbf{s}}) \otimes \Im(\underline{\mathbf{R}})\Big)^+ \\ &= \Big(\Re(\mathbf{S})\Re(\underline{\mathbf{R}}) + \Im(\mathbf{S})\Im(\underline{\mathbf{R}})\Big)^+ \end{aligned}$$

C. Rewriting of the real vector problem

We rewrite $\underline{\mathbf{d}}$ in function of $\underline{\mathbf{x}}$ for the next optimization step using $\underline{\mathbf{R}} = \mathbf{H}\underline{\mathbf{x}}$:

$$\Re(\underline{\mathbf{R}}) = \Re(\mathbf{H})\Re(\underline{\mathbf{x}}) - \Im(\mathbf{H})\Im(\underline{\mathbf{x}})$$

$$\Im(\underline{\mathbf{R}}) = \Re(\mathbf{H})\Im(\underline{\mathbf{x}}) + \Im(\mathbf{H})\Re(\underline{\mathbf{x}})$$

This gives

$$\underline{\mathbf{d}} = \left(\underline{\underline{\mathbf{S}}}\mathbf{A}\underline{\underline{\mathbf{x}}}\right)^{+} = \left(\mathbf{M}\underline{\underline{\mathbf{x}}}\right)^{+} \qquad (1)$$

with the following notation:

$$\underline{\underline{\mathbf{S}}} = \Big(\mathrm{Diag}\big(\Re(\underline{\mathbf{s}})\big),\ \mathrm{Diag}\big(\Im(\underline{\mathbf{s}})\big)\Big)$$

$$\mathbf{A} = \begin{pmatrix} \Re(\mathbf{H}) & -\Im(\mathbf{H}) \\ \Im(\mathbf{H}) & \Re(\mathbf{H}) \end{pmatrix}$$

and

$$\underline{\underline{\mathbf{x}}} = \begin{pmatrix} \Re(\underline{\mathbf{x}}) \\ \Im(\underline{\mathbf{x}}) \end{pmatrix}$$

Additionally, we rewrite $\underline{\mathbf{R}}$ as

$$\begin{aligned} \underline{\mathbf{R}} &= \begin{pmatrix} \mathbf{I}_K & j\mathbf{I}_K \end{pmatrix} \begin{pmatrix} \Re(\underline{\mathbf{R}}) \\ \Im(\underline{\mathbf{R}}) \end{pmatrix} = \mathcal{I}_K \begin{pmatrix} \Re(\underline{\mathbf{R}}) \\ \Im(\underline{\mathbf{R}}) \end{pmatrix} \\ &= \mathcal{I}_K \begin{pmatrix} \Re(\mathbf{H}) & -\Im(\mathbf{H}) \\ \Im(\mathbf{H}) & \Re(\mathbf{H}) \end{pmatrix} \begin{pmatrix} \Re(\underline{\mathbf{x}}) \\ \Im(\underline{\mathbf{x}}) \end{pmatrix} = \mathcal{I}_K \mathbf{A}\underline{\underline{\mathbf{x}}} \qquad (2) \end{aligned}$$

(1) and (2) allow us to rewrite our problem in function of $\underline{\underline{\mathbf{x}}}$ with exclusively real inputs, while having relatively simple matrices expressions (in opposition to the complex input case, see VIII, case 2):

$$\begin{aligned} \hat{\underline{\underline{\mathbf{x}}}} &= \arg \min_{\mathbf{P},\mathbf{d_i} \geq 0} \|\underline{\mathbf{d}} \otimes \underline{\mathbf{s}} - \underline{\mathbf{R}}\|_2^2 \\ &= \arg \min_{\underline{\underline{\mathbf{x}}}} \|\mathbf{S}\left(\mathbf{M}\underline{\underline{\mathbf{x}}}\right)^{+} - \mathcal{I}_K \mathbf{A}\underline{\underline{\mathbf{x}}}\|_2^2 \qquad (3) \end{aligned}$$

Identically to [1], equation (3) has a trivial all-zeros solution, and we introduce a regularizer $-\frac{\delta}{2}\|\underline{\underline{\mathbf{x}}}\|_2^2$ (with $\delta > 0$) to prevent converging towards it.

D. Derivativon of the non-negative vector problem

To solve this problem, we will use the same implementation of FBS as [1] and evaluate its performance on simulated signals. This implies computing the derivative of the function $f(\underline{\underline{\mathbf{x}}}) = \|\mathbf{S}\left(\mathbf{M}\underline{\underline{\mathbf{x}}}\right)^{+} - \mathcal{I}_K \mathbf{A}\underline{\underline{\mathbf{x}}}\|_2^2$. To do so, we first rewrite $\left(\mathbf{M}\underline{\underline{\mathbf{x}}}\right)^{+}$ as

$$\begin{aligned} \left(\mathbf{M}\underline{\underline{\mathbf{x}}}\right)^{+} &= \begin{pmatrix} \sum_{i=1}^{2M} \mathbf{M}_{1,i}\mathbf{x}_i \\ \vdots \\ \sum_{i=1}^{2M} \mathbf{M}_{K,i}\mathbf{x}_i \end{pmatrix} \otimes \begin{pmatrix} U\left(\sum_{i=1}^{2M} \mathbf{M}_{1,i}\mathbf{x}_i\right) \\ \vdots \\ U\left(\sum_{i=1}^{2M} \mathbf{M}_{K,i}\mathbf{x}_i\right) \end{pmatrix} \\ &= \left(\mathbf{M}\underline{\underline{\mathbf{x}}}\right) \otimes U\left(\mathbf{M}\underline{\underline{\mathbf{x}}}\right) \\ &= \mathrm{Diag}\left(\mathbf{M}\underline{\underline{\mathbf{x}}}\right) U\left(\mathbf{M}\underline{\underline{\mathbf{x}}}\right) \end{aligned}$$

With U the Heaviside step function

$$U(x) = \begin{cases} 0 \text{ if } x \leq 0 \\ 1 \text{ if } x > 0 \end{cases}$$

whose derivative can be numerically approximated with $\frac{dU(x)}{dx} = 0$. This approximation can be made as the only issue would be for $x = 0$, where U is not differentiable, which is unlikely to happen due to the approximation errors of the machine used. Despite this, we also computed and implemented a method using a sigmoid function (instead of Heaviside) to be more precise (see part IV-E).
This leaves us with

$$\begin{aligned} \frac{d\left(\mathbf{M}\underline{\underline{\mathbf{x}}}\right)^{+}}{d\underline{\underline{\mathbf{x}}}} &= \mathbf{M} \otimes \mathcal{U} \\ &= \mathbf{M} \otimes \begin{pmatrix} U\left(\sum_{i=1}^{2M} \mathbf{M}_{1,i}\underline{\underline{\mathbf{x}}}_i\right) & \dots & U\left(\sum_{i=1}^{2M} \mathbf{M}_{1,i}\underline{\underline{\mathbf{x}}}_i\right) \\ \vdots & \ddots & \vdots \\ U\left(\sum_{i=1}^{2M} \mathbf{M}_{K,i}\underline{\underline{\mathbf{x}}}_i\right) & \dots & U\left(\sum_{i=1}^{2M} \mathbf{M}_{K,i}\underline{\underline{\mathbf{x}}}_i\right) \end{pmatrix} \end{aligned}$$

To compute the derivative of our function, we write:

$$\begin{aligned} f(\underline{\underline{\mathbf{x}}}) &= \|\mathbf{S}\left(\mathbf{M}\underline{\underline{\mathbf{x}}}\right)^{+} - \mathcal{I}_K \mathbf{A}\underline{\underline{\mathbf{x}}}\|_2^2 \\ &= \sum_{i=1}^{K} \left| \left(\mathbf{S}\left(\mathbf{M}\underline{\underline{\mathbf{x}}}\right)^{+} - \mathcal{I}_K \mathbf{A}\underline{\underline{\mathbf{x}}}\right)_i \right|^2 \\ &= \sum_{i=1}^{K} \overline{\left(\mathbf{S}\left(\mathbf{M}\underline{\underline{\mathbf{x}}}\right)^{+} - \mathcal{I}_K \mathbf{A}\underline{\underline{\mathbf{x}}}\right)_i} \left(\mathbf{S}\left(\mathbf{M}\underline{\underline{\mathbf{x}}}\right)^{+} - \mathcal{I}_K \mathbf{A}\underline{\underline{\mathbf{x}}}\right)_i \end{aligned}$$

Finally, we obtain $\mathrm{grad}_{\underline{\underline{\mathbf{x}}}}\left(f(\underline{\underline{\mathbf{x}}})\right)$ in the form of a matrix multiplication :

$$\begin{aligned} &\frac{d\|\mathbf{S}\left(\mathbf{M}\underline{\underline{\mathbf{x}}}\right)^{+} - \mathcal{I}_K \mathbf{A}\underline{\underline{\mathbf{x}}}\|_2^2}{d\underline{\underline{\mathbf{x}}}} \\ &\quad = \left(\mathbf{S}\left[\mathbf{M} \otimes \mathcal{U}\right] - \mathcal{I}_K \mathbf{A}\right)^T \overline{\left(\mathbf{S}\left(\mathbf{M}\underline{\underline{\mathbf{x}}}\right)^{+} - \mathcal{I}_K \mathbf{A}\underline{\underline{\mathbf{x}}}\right)} \\ &\quad + \overline{\left(\mathbf{S}\left[\mathbf{M} \otimes \mathcal{U}\right] - \mathcal{I}_K \mathbf{A}\right)}^T \left(\mathbf{S}\left(\mathbf{M}\underline{\underline{\mathbf{x}}}\right)^{+} - \mathcal{I}_K \mathbf{A}\underline{\underline{\mathbf{x}}}\right) \\ &\quad = 2\Re\left[\left(\mathbf{S}\left[\mathbf{M} \otimes \mathcal{U}\right] - \mathcal{I}_K \mathbf{A}\right)^T \overline{\left(\mathbf{S}\left(\mathbf{M}\underline{\underline{\mathbf{x}}}\right)^{+} - \mathcal{I}_K \mathbf{A}\underline{\underline{\mathbf{x}}}\right)}\right] \end{aligned}$$

This allows us to apply the C2PO algorithm to our method. The algorithm presented bellow is the one used in our simulations (see section III), The Quantized ZF algorithm is first applied to the system using $\underline{\mathbf{s}}$ and $\mathbf{H}$ as inputs. The output vector $\underline{\mathbf{R}}$ is then computed for the infinite SNR case (no added noise). If there is a single error during the transmission, the itterative method is used. The final $\underline{\mathbf{R}}$ output computed is compared to the first one, and the precoded vector $\underline{\mathbf{x}}$ achieving the best symbol error rate is retained. The SER(x,s) function used in the algorithm computes the symbol error rate of x with respect to s.

Algorithm 2: C2PO with non-negative vector for a Q-PSK input constellation

```
Input: $\underline{\mathbf{s}}, \mathbf{H}, \tau^{(t)}, \delta$
Let $\mathbf{P} = \mathbf{H}^H(\mathbf{H}\mathbf{H}^H)$
if $\mathit{SER}(\mathbb{Q}(\mathbf{H}\mathbb{Q}(\mathbf{P}\mathbf{s})), \mathbf{s}) \neq 0$ then
    Initialize $\mathbf{x}^{(0)} = \mathbf{P}\mathbf{s}$
    Compute $\underline{\underline{\mathbf{x}}}^{(0)}$ with $\mathbf{x}^{(0)}$
    for $t \in [1, t_{max}]$ do
        $\mathbf{z}^{(t)} = \underline{\underline{\mathbf{x}}}^{(t-1)} - \tau^{(t)} \frac{d\|\mathbf{S}(\mathbf{M}\underline{\underline{\mathbf{x}}})^{+} - \mathcal{I}_K \mathbf{A}\underline{\underline{\mathbf{x}}}\|_2^2}{d\underline{\underline{\mathbf{x}}}}$
        $\underline{\underline{\mathbf{x}}}^{(t)} = \min(\max(\rho\mathbf{z}^{(t)}, -\xi), \xi)$
    end
end
if $\mathit{SER}(\mathbb{Q}(\mathbf{H}\mathbb{Q}(\mathbf{P}\mathbf{s})), \mathbf{s}) > \mathit{SER}(\mathbb{Q}(\mathbf{H}\mathbb{Q}(\mathbf{x}^{(t_{max})})), \mathbf{s})$
 then
    $\mathbf{x}_{out} = \mathbf{x}^{(tmax)}$
else
    $\mathbf{x}_{out} = \mathbb{Q}(\mathbf{P}\mathbf{s})$
end
Output: $\mathbf{x}_{out}$
```

E. Derivation with a sigmoid function

Due to poor results (see III, red curves) of the previous algorithm compared to state of the art algorithm, we improved the previous proposed method using a differentiable function. To do so, we used the sigmoid function instead of the Heaviside U, with:

$$\mathrm{sigmoid}(x, \lambda) = f_\lambda(x) = \frac{1}{1 + e^{-\lambda x}} = \frac{e^{\lambda x}}{e^{\lambda x} + 1}$$

And its derivative:

$$\frac{df_\lambda(x)}{dx} = \lambda f_\lambda(x)(1 - f_\lambda(x))$$

This definition will allow us to avoid the differentiability issue we had in the case of the Heaviside function. Additionally, for a high value of λ, the behavior of this model is the same as with a Heaviside function. We thus set the hyperparameter λ to 100 for our evaluations.

The new model leads us to

$$
\begin{aligned}
\big(\mathbf{M}\underline{\underline{\mathbf{x}}}\big)^{+} &= \begin{pmatrix} \sum_{i=1}^{2M} \mathbf{M}_{1,i}\mathbf{x}_i \\ \vdots \\ \sum_{i=1}^{2M} \mathbf{M}_{K,i}\mathbf{x}_i \end{pmatrix} \otimes \begin{pmatrix} f_\lambda\left(\sum_{i=1}^{2M} \mathbf{M}_{1,i}\mathbf{x}_i\right) \\ \vdots \\ f_\lambda\left(\sum_{i=1}^{2M} \mathbf{M}_{K,i}\mathbf{x}_i\right) \end{pmatrix} \\
&= \big(\mathbf{M}\underline{\underline{\mathbf{x}}}\big) \otimes f_\lambda\big(\mathbf{M}\underline{\underline{\mathbf{x}}}\big) \\
&= \mathrm{Diag}\big(\mathbf{M}\underline{\underline{\mathbf{x}}}\big) f_\lambda\big(\mathbf{M}\underline{\underline{\mathbf{x}}}\big)
\end{aligned}
$$

The derivative resulting from this formula is:

$$
\begin{aligned}
\frac{d\big(\mathbf{M}\underline{\underline{\mathbf{x}}}\big)^{+}}{d\underline{\underline{\mathbf{x}}}} =& \mathrm{Diag}\big(f_\lambda(\mathbf{M}\underline{\underline{\mathbf{x}}})\big)\mathbf{M} + \lambda \mathrm{Diag}\big(f_\lambda(\mathbf{M}\underline{\underline{\mathbf{x}}})\big) \\
& \times \mathrm{Diag}\big(\mathbf{M}\underline{\underline{\mathbf{x}}}\big)\Big(I_K - \mathrm{Diag}\big(f_\lambda(\mathbf{M}\underline{\underline{\mathbf{x}}})\big)\Big)\mathbf{M} \\
=& \mathrm{Diag}\big(f_\lambda(\mathbf{M}\underline{\underline{\mathbf{x}}})\big)\Big[I_K + \lambda \mathrm{Diag}\big(\mathbf{M}\underline{\underline{\mathbf{x}}}\big) \\
& \times \Big(I_K - \mathrm{Diag}\big(f_\lambda(\mathbf{M}\underline{\underline{\mathbf{x}}})\big)\Big)\Big]\mathbf{M}
\end{aligned}
$$

We compute the derivative with the help of (IV-D), and find

$$
\begin{aligned}
&\frac{d\|\mathbf{S}\big(\mathbf{M}\underline{\underline{\mathbf{x}}}\big)^{+} - \mathcal{I}_K \mathbf{A}\underline{\underline{\mathbf{x}}}\|_2^2}{d\underline{\underline{\mathbf{x}}}} = \\
&= \Big(\mathbf{S}\ \mathrm{Diag}\big(f_\lambda(\mathbf{M}\underline{\underline{\mathbf{x}}})\big)\Big[I_K + \lambda \mathrm{Diag}\big(\mathbf{M}\underline{\underline{\mathbf{x}}}\big) \times \\
&\Big(I_K - \mathrm{Diag}\big(f_\lambda(\mathbf{M}\underline{\underline{\mathbf{x}}})\big)\Big)\Big]\mathbf{M} - \mathcal{I}_K \mathbf{A}\Big)^{t} \times \\
&\overline{\Big(\mathbf{S}\big(\mathbf{M}\underline{\underline{\mathbf{x}}}\big)^{+} - \mathcal{I}_K \mathbf{A}\underline{\underline{\mathbf{x}}}\Big)} + \\
&\overline{\Big(\mathbf{S}\ \mathrm{Diag}\big(f_\lambda(\mathbf{M}\underline{\underline{\mathbf{x}}})\big)\Big[I_K + \lambda \mathrm{Diag}\big(\mathbf{M}\underline{\underline{\mathbf{x}}}\big) \times} \\
&\overline{\Big(I_K - \mathrm{Diag}\big(f_\lambda(\mathbf{M}\underline{\underline{\mathbf{x}}})\big)\Big)\Big]\mathbf{M} - \mathcal{I}_K \mathbf{A}\Big)^{t}} \times \\
&\Big(\mathbf{S}\big(\mathbf{M}\underline{\underline{\mathbf{x}}}\big)^{+} - \mathcal{I}_K \mathbf{A}\underline{\underline{\mathbf{x}}}\Big) \\
&= 2\Re\Bigg[\Big(\mathbf{S}\ \mathrm{Diag}\big(f_\lambda(\mathbf{M}\underline{\underline{\mathbf{x}}})\big)\Big[I_K + \lambda \mathrm{Diag}\big(\mathbf{M}\underline{\underline{\mathbf{x}}}\big) \times \\
&\Big(I_K - \mathrm{Diag}\big(f_\lambda(\mathbf{M}\underline{\underline{\mathbf{x}}})\big)\Big)\Big]\mathbf{M} - \mathcal{I}_K \mathbf{A}\Big)^{t} \times \\
&\overline{\Big(\mathbf{S}\big(\mathbf{M}\underline{\underline{\mathbf{x}}}\big)^{+} - \mathcal{I}_K \mathbf{A}\underline{\underline{\mathbf{x}}}\Big)}\Bigg]
\end{aligned}
$$

We implemented a version with this slight improvement.The results and analysis of this method will be presented in section IV-F. The hyperparameters used are presented in Table III.

F. Analysis of the non-negative vector approach

The resulting plots on Figure 2 and 3 show us that our first method is a good upgrade to the simple Quantized ZF algorithm, but is outclassed by the method proposed by [1]. Additionally, the improvement with the sigmoid function does not increase significantly enough the performances of the algorithm to consider it worth it (due to additional computations), while not being detrimental. This method however performs worse than the one proposed in [1]. This can be explained by the fact that even if our amplification step handles each $\mathbf{s}_i$ individually to allocate the right amplification, it can not cancel out a potential rotation of the symbols during the transmission. On the contrary, the original C2PO method addresses this issue while not processing each $\mathbf{s}_i$ individually. This lead us to consider a second approach in the hope of outperforming our first method.

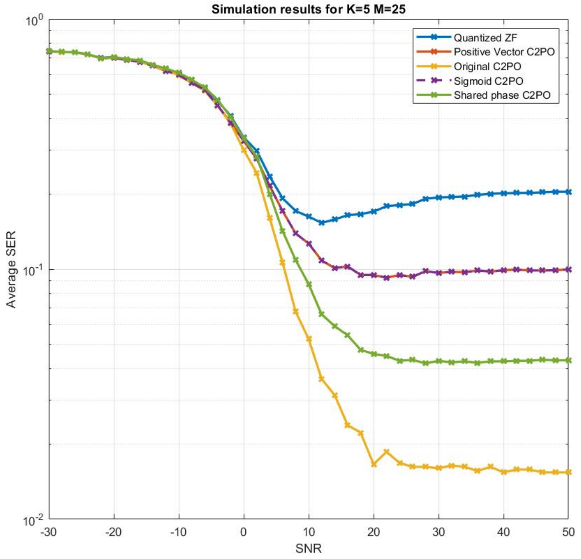

Fig. 2. Results for $K = 5$ and $M = 25$

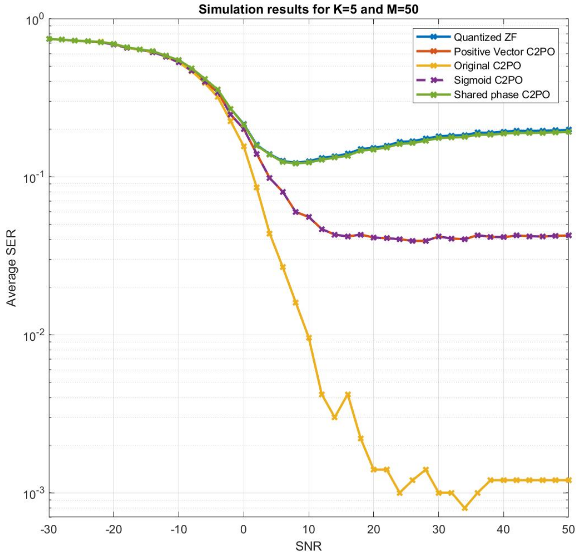

Fig. 3. Results for $K = 5$ and $M = 50$

V. Shared phase method

A. Limitation of the complex amplification vector

Due to the real coefficients $\mathbf{d}_i$, the amplification step does not take into account a phase shift of the Q-PSK inputs. This shift could however be beneficial during the precoding step. To take into account this complex case, the most obvious solution would be to amplify the vector $\underline{\mathbf{s}}$ by complex values $\mathbf{d}_i$ instead of real values. This would imply

$$\{\hat{\underline{\mathbf{x}}}, \hat{\underline{\mathbf{d}}}\} = \arg \min_{\underline{\mathbf{x}} \in \mathcal{S}^M, \mathbf{d_i} \in \mathbb{C}} \|\mathbf{H}\underline{\mathbf{x}} - \underline{\mathbf{d}} \otimes \underline{\mathbf{s}}\|_2^2$$

Solving this optimization problem over $\underline{d}$ without constraining it is however not easy, as both an analytical and closed form computations lead us to

$$\mathbf{d}_i = \frac{\mathbf{R}_i}{\mathbf{s}_i}, \text{ or in matrix form } \underline{\mathbf{d}} = \mathbf{S}^{-1}\underline{\mathbf{R}}$$

Injecting (V-A) into equation (V-A) leads us to a dead end:

$$\{\hat{\underline{\mathbf{x}}}\} = \arg \min_{\underline{\mathbf{x}} \in \mathcal{S}^M} \|\underline{0}\|_2^2$$

To avoid this issue, we constrain the elements of $\underline{\mathbf{d}}$ to share the same argument and to each have their own modulus:

$$\mathbf{d}_j \in \mathbb{C} \text{ with } \mathbf{d}_j = r_j e^{i\varphi}, \forall j \in [\![1; K]\!]$$

This will allow us to consider a different amplification for each input while adding a global rotation to the inputs, in the same way as [1] but with more freedom for each input.

B. Optimization of the complex amplification vector

We first compute $\hat{\underline{\mathbf{d}}}$

$$\hat{\underline{\mathbf{d}}} = \arg\min \|\mathbf{D}\underline{\mathbf{s}} - \underline{\mathbf{R}}\|_2^2$$

by expressing $\|\mathbf{D}\underline{\mathbf{s}} - \underline{\mathbf{R}}\|_2^2$ in function of the r_j $\forall j \in [\![1; K]\!]$ and φ:

$$\begin{aligned}
&\|\mathbf{D}\underline{\mathbf{s}} - \underline{\mathbf{R}}\|_2^2 = f(\mathbf{d}_1, \dots, \mathbf{d}_K) = f(r_1, \dots, r_K, \varphi) \\
&= \sum_{j=1}^{K} \Big[r_j \Re(e^{i\varphi}\mathbf{s}_j) - \Re(\mathbf{R}_j) \Big]^2 + \Big[r_j \Im(e^{i\varphi}\mathbf{s}_j) - \Im(\mathbf{R}_j) \Big]^2 \\
&= \sum_{j=1}^{K} \Big[r_j \big[\Re(e^{i\varphi})\Re(\mathbf{s}_j) - \Im(e^{i\varphi})\Im(\mathbf{s}_j) \big] - \Re(\mathbf{R}_j) \Big]^2 \\
&\quad + \Big[r_j \big[\Re(e^{i\varphi})\Im(\mathbf{s}_j) + \Im(e^{i\varphi})\Re(\mathbf{s}_j) \big] - \Im(\mathbf{R}_j) \Big]^2 \\
&= \sum_{j=1}^{K} \Big[r_j \big[\cos(\varphi)\Re(\mathbf{s}_j) - \sin(\varphi)\Im(\mathbf{s}_j) \big] - \Re(\mathbf{R}_j) \Big]^2 \\
&\quad + \Big[r_j \big[\cos(\varphi)\Im(\mathbf{s}_j) + \sin(\varphi)\Re(\mathbf{s}_j) \big] - \Im(\mathbf{R}_j) \Big]^2
\end{aligned}$$

We then compute the gradient of this function, i.e. the derivatives with respect to every r_j and φ

$$\begin{aligned}
&\forall n \in [\![1; K]\!], \quad \frac{df(r_1, \dots, r_K, \varphi)}{dr_n} \\
&= 2\Re\big(e^{i\varphi}\mathbf{s}_n\big) \Big[r_n \Re\big(e^{i\varphi}\mathbf{s}_n\big) - \Re\big(\mathbf{R}_n\big) \Big] \\
&\quad + 2\Im\big(e^{i\varphi}\mathbf{s}_n\big) \Big[r_n \Im\big(e^{i\varphi}\mathbf{s}_n\big) - \Im\big(\mathbf{R}_n\big) \Big]
\end{aligned}$$

$$\begin{aligned}
&\frac{df(r_1, \dots, r_K, \varphi)}{d\varphi} \\
&= \sum_{j=1}^{K} 2 r_j \Re\big(e^{i\varphi}\mathbf{s}_j\big) \Big[r_j \Im\big(e^{i\varphi}\mathbf{s}_j\big) - \Im\big(\mathbf{R}_j\big) \Big] \\
&\quad - 2 r_j \Im\big(e^{i\varphi}\mathbf{s}_j\big) \Big[r_j \Re\big(e^{i\varphi}\mathbf{s}_j\big) - \Re\big(\mathbf{R}_j\big) \Big]
\end{aligned}$$

Solving $\frac{df}{dr_n} = 0$ gives us

$$r_n = \frac{\Re(e^{i\varphi}\mathbf{s}_n)\Re(\mathbf{R}_n) + \Im(e^{i\varphi}\mathbf{s}_n)\Im(\mathbf{R}_n)}{\Re(e^{i\varphi}\mathbf{s}_n)^2 + \Im(e^{i\varphi}\mathbf{s}_n)^2}$$

As $\mathbf{s}_n$ is in a Q-PSK constellation, we have $|\mathbf{s}|^2 = \Re(\mathbf{s}_n)^2 + \Im(\mathbf{s}_n)^2 = 1$, and thus $|e^{i\varphi}\mathbf{s}_n| = 1$ as multiplying a complex number by $e^{i\varphi}$ does not change its modulus. This would work for any M-PSK constellation (as well as any constellation with constant modulus, where r_n should be divided by it), and allows us to write r_n as

$$\begin{aligned} r_n =& \Re(e^{i\varphi}\mathbf{s}_n)\Re(\mathbf{R}_n) + \Im(e^{i\varphi}\mathbf{s}_n)\Im(\mathbf{R}_n) \\ =& \big[\cos(\varphi)\Re(\mathbf{s}_n) - \sin(\varphi)\Im(\mathbf{s}_n)\big]\Re(\mathbf{R}_n) \\ &+ \big[\cos(\varphi)\Im(\mathbf{s}_n) + \sin(\varphi)\Re(\mathbf{s}_n)\big]\Im(\mathbf{R}_n) \\ =& \cos(\varphi)\big[\Re(\mathbf{s}_n)\Re(\mathbf{R}_n) + \Im(\mathbf{s}_n)\Im(\mathbf{R}_n)\big] \\ &+ \sin(\varphi)\big[\Re(\mathbf{s}_n)\Im(\mathbf{R}_n) - \Im(\mathbf{s}_n)\Re(\mathbf{R}_n)\big] \end{aligned}$$

From now on, we denote a_j and b_j as

$$\begin{aligned} a_j &= \Re(\mathbf{s}_n)\Re(\mathbf{R}_n) + \Im(\mathbf{s}_n)\Im(\mathbf{R}_n) \\ b_j &= \Re(\mathbf{s}_n)\Im(\mathbf{R}_n) - \Im(\mathbf{s}_n)\Re(\mathbf{R}_n) \end{aligned}$$

allowing us to write

$$r_n = \cos(\varphi)a_n + \sin(\varphi)b_n$$

We now solve $\frac{df}{d\varphi} = 0$ using (V-B) and (V-B):

$$\begin{aligned} &\frac{d\,f}{d\varphi} = 0 \\ \Leftrightarrow &\sum_{j=1}^{K} r_j \Re\big(e^{i\varphi}\mathbf{s}_j\big)\Im\big(\mathbf{R}_j\big) = \sum_{j=1}^{K} r_j \Im\big(e^{i\varphi}\mathbf{s}_j\big)\Re\big(\mathbf{R}_j\big) \\ \Leftrightarrow &\sum_{j=1}^{K} a_j b_j \big[\cos^2(\varphi) - \sin^2(\varphi)\big] \\ &\qquad + \cos(\varphi)\sin(\varphi)\big[b_j^2 - a_j^2\big] = 0 \\ \Leftrightarrow &\cos(2\varphi)\sum_{j=1}^{K} a_j b_j = \frac{1}{2}\sin(2\varphi)\sum_{j=1}^{K}(a_j^2 - b_j^2) \\ \Leftrightarrow &\varphi = \frac{1}{2}\arctan\left(\frac{2\sum_{j=1}^{K} a_j b_j}{\sum_{j=1}^{K}(a_j^2 - b_j^2)}\right) \end{aligned}$$

Developing $a_j b_j$ and $a_j^2 - b_j^2$ as a function of $\mathbf{s}_j$ and $\mathbf{R}_j$ allows us to simplify both numerator and denominator

$$\begin{aligned} \sum_{j=1}^{K} a_j b_j =& \sum_{j=1}^{K} \Re(\mathbf{R}_j)\Im(\mathbf{R}_j)\big[\Re(\mathbf{s}_j)^2 - \Im(\mathbf{s}_j)^2\big] \\ &+ \Re(\mathbf{s}_j)\Im(\mathbf{s}_j)\big[\Im(\mathbf{R}_j)^2 - \Re(\mathbf{R}_j)^2\big] \\ =& \sum_{j=1}^{K} \Re(\mathbf{s}_j)\Im(\mathbf{s}_j)\big[\Im(\mathbf{R}_j)^2 - \Re(\mathbf{R}_j)^2\big] \end{aligned}$$

and

$$\begin{aligned} \sum_{j=1}^{K} a_j^2 - b_j^2 =& \sum_{j=1}^{K} \big[\Re(\mathbf{s}_j)^2 - \Im(\mathbf{s}_j)^2\big]\big[\Re(\mathbf{R}_j)^2 - \Im(\mathbf{R}_j)^2\big] \\ &+ 4\Re(\mathbf{R}_j)\Im(\mathbf{R}_j)\Re(\mathbf{s}_j)\Im(\mathbf{s}_j) \\ =& 4\sum_{j=1}^{K} \Re(\mathbf{R}_j)\Im(\mathbf{R}_j)\Re(\mathbf{s}_j)\Im(\mathbf{s}_j) \end{aligned}$$

As $\mathbf{s}_j$ is in a Q-PSK constellation, $|\Im(\mathbf{s}_j)| = |\Re(\mathbf{s}_j)|$. This allows us to do the above simplifications as $\Re(\mathbf{s}_j)^2 - \Im(\mathbf{s}_j)^2 = 0$. This simplification can however not be made for $\mathbf{R}_j$, as there is no guarantee that the output vector belongs to a Q-PSK constellation (as the computations we made do not take into account the quantization steps).

Thus

$$\varphi = \frac{1}{2}\arctan\left(\frac{\sum_{j=1}^{K}\Re(\mathbf{s}_j)\Im(\mathbf{s}_j)\big[\Im(\mathbf{R}_j)^2 - \Re(\mathbf{R}_j)^2\big]}{2\sum_{j=1}^{K}\Re(\mathbf{R}_j)\Im(\mathbf{R}_j)\Re(\mathbf{s}_j)\Im(\mathbf{s}_j)}\right)$$

And $\forall n \in [\![1;K]\!]$

$$\begin{aligned} \mathbf{d}_n =& \big[\cos^2(\varphi)a_n + \cos(\varphi)\sin(\varphi)b_n\big] \\ &+ i\big[\cos(\varphi)\sin(\varphi)a_n + \sin^2(\varphi)b_n\big] \end{aligned}$$

C. *Derivative of our function*

Similarly to section IV-A, we use a regularization term $-\frac{\delta}{2}\|\underline{\mathbf{x}}\|_2^2$ to prevent the all-zero solution, and we will implement an adaptation of C2PO with our equations. We thus need to derive the function with respect to $\underline{\mathbf{x}}$. To do so, we split the vector $\underline{\mathbf{x}}$ as in IV-D in its Real and Imaginary part. We thus need to express separately $\forall l \in [\![1;K]\!]$

$$\frac{d\,\|\underline{\mathbf{d}} \otimes \underline{\mathbf{s}} - \underline{\mathbf{R}}\|_2^2}{d\Re(\mathbf{x}_l)}$$

and

$$\frac{d\,\|\underline{\mathbf{d}} \otimes \underline{\mathbf{s}} - \underline{\mathbf{R}}\|_2^2}{d\Im(\mathbf{x}_l)}$$

For the sake of clarity and readability, we will decompose the functions (V-C) and (V-C) into smaller sub-functions and notations instead of writing the entire functions in massive blocs.

For our computations, we use the fact that $\underline{\mathbf{R}} = \mathbf{H}\underline{\mathbf{x}}$ and thus

$$\forall n \in [\![1;K]\!], \quad \mathbf{R}_n = \sum_{i=1}^{M}\mathbf{H}_{n,i}\mathbf{x}_i$$

To express (V-C), we split it in two separate terms:

$$\begin{aligned} &\frac{d\,\|\underline{\mathbf{d}} \otimes \underline{\mathbf{s}} - \underline{\mathbf{R}}\|_2^2}{d\Re(\mathbf{x}_l)} \\ &= 2\sum_{i=1}^{K}\Re\big(\mathbf{d}_i\mathbf{s}_i - \mathbf{R}_i\big)\frac{d}{d\Re(\mathbf{x}_l)}\Big(\Re\big(\mathbf{d}_i\mathbf{s}_i - \mathbf{R}_i\big)\Big) \\ &+ 2\sum_{i=1}^{K}\Im\big(\mathbf{d}_i\mathbf{s}_i - \mathbf{R}_i\big)\frac{d}{d\Re(\mathbf{x}_l)}\Big(\Im\big(\mathbf{d}_i\mathbf{s}_i - \mathbf{R}_i\big)\Big) \end{aligned}$$

with

$$\begin{aligned} &\Re\big(\mathbf{d}_i\mathbf{s}_i - \mathbf{R}_i\big) = \Re(\mathbf{d}_i\mathbf{s}_i) - \Re(\mathbf{R}_i) \\ &= \Re(\mathbf{s}_i)\big(\cos^2(\varphi)a_i + \cos(\varphi)\sin(\varphi)b_i\big) \\ &\quad - \Im(\mathbf{s}_i)\big(\cos(\varphi)\sin(\varphi)a_i + \sin^2(\varphi)b_i\big) - \sum_{j=1}^{K}\Re(\mathbf{H}_{i,j}\mathbf{x}_j) \end{aligned}$$

and

$$\begin{aligned} &\Im\big(\mathbf{d}_i\mathbf{s}_i - \mathbf{R}_i\big) = \Im(\mathbf{d}_i\mathbf{s}_i) - \Im(\mathbf{R}_i) \\ &= \Im(\mathbf{s}_i)\big(\cos^2(\varphi)a_i + \cos(\varphi)\sin(\varphi)b_i\big) \\ &\quad + \Re(\mathbf{s}_i)\big(\cos(\varphi)\sin(\varphi)a_i + \sin^2(\varphi)b_i\big) - \sum_{j=1}^{K}\Im(\mathbf{H}_{i,j}\mathbf{x}_j) \end{aligned}$$

This leads to

$$
\begin{aligned}
&\frac{d}{d\Re(\mathbf{x}_l)}\Big(\Re\big(\mathbf{d}_i\mathbf{s}_i-\mathbf{R}_i\big)\Big)=\Re(\mathbf{s}_i)\Big[\cos^2(\varphi)\frac{d\,a_i}{d\Re(\mathbf{x}_l)}\\
&-2a_i\cos(\varphi)\sin(\varphi)\frac{d\,\varphi}{d\Re(\mathbf{x}_l)}+\cos(\varphi)\sin(\varphi)\frac{d\,b_i}{d\Re(\mathbf{x}_l)}\\
&+\cos^2(\varphi)b_i\frac{d\,\varphi}{d\Re(\mathbf{x}_l)}-\sin^2(\varphi)b_i\frac{d\,\varphi}{d\Re(\mathbf{x}_l)}\Big]\\
&-\Im(\mathbf{s}_i)\Big[\sin^2(\varphi)\frac{d\,b_i}{d\Re(\mathbf{x}_l)}+2b_i\cos(\varphi)\sin(\varphi)\frac{d\,\varphi}{d\Re(\mathbf{x}_l)}\\
&+\cos(\varphi)\sin(\varphi)\frac{d\,a_i}{d\Re(\mathbf{x}_l)}+\cos^2(\varphi)a_i\frac{d\,\varphi}{d\Re(\mathbf{x}_l)}\\
&-\sin^2(\varphi)a_i\frac{d\,\varphi}{d\Re(\mathbf{x}_l)}\Big]-\Re(\mathbf{H}_{i,l})\\
&=\Re(\mathbf{s}_i)\times\mathbf{A}-\Im(\mathbf{s}_i)\times\mathbf{B}-\Re(\mathbf{H}_{i,l})
\end{aligned}
$$

Allowing us to write

$$
\begin{aligned}
&\frac{d}{d\Re(\mathbf{x}_l)}\Big(\Im\big(\mathbf{d}_i\mathbf{s}_i-\mathbf{R}_i\big)\Big)\\
&\qquad=\Im(\mathbf{s}_i)\times\mathbf{A}+\Re(\mathbf{s}_i)\times\mathbf{B}-\Im(\mathbf{H}_{i,l})
\end{aligned}
$$

We finally need to express $\frac{d\,a_i}{d\Re(\mathbf{x}_l)}$, $\frac{d\,b_i}{d\Re(\mathbf{x}_l)}$ and $\frac{d\,\varphi}{d\Re(\mathbf{x}_l)}$ $\forall l,i\in[\![1;K]\!]$

$$
\frac{d\,a_i}{d\Re(\mathbf{x}_l)}=\Re(\mathbf{s}_i)\Re(\mathbf{H}_{i,l})+\Im(\mathbf{s}_i)\Im(\mathbf{H}_{i,l})
$$

$$
\frac{d\,b_i}{d\Re(\mathbf{x}_l)}=\Re(\mathbf{s}_i)\Im(\mathbf{H}_{i,l})-\Im(\mathbf{s}_i)\Re(\mathbf{H}_{i,l})
$$

$$
\begin{aligned}
&\frac{d\,\varphi}{d\Re(\mathbf{x}_l)}=\frac{1}{1+\big(\tan(2\varphi)\big)^2}\sum_{i=1}^{K}\Im(\mathbf{s}_i)\Re(\mathbf{s}_i)\\
&\times\Bigg[\frac{\Big[\Im(\mathbf{H}_{i,l})\Im(\mathbf{R_i})-\Re(\mathbf{H}_{i,l})\Re(\mathbf{R_i})\Big]\frac{\sum_{j=1}^{K}a_j^2-b_j^2}{2}}{\left(\frac{\sum_{j=1}^{K}a_j^2-b_j^2}{2}\right)^2}+\\
&\frac{\Re(\mathbf{R_i})^2\sum_{j=1}^{K}\Im(\mathbf{s}_j)\Re(\mathbf{s}_j)[\Re(\mathbf{R}_j)\Im(\mathbf{H}_{j,l})+\Im(\mathbf{R}_j)\Re(\mathbf{H}_{j,l})]}{\left(\frac{\sum_{j=1}^{K}a_j^2-b_j^2}{2}\right)^2}-\\
&\frac{\Im(\mathbf{R_i})^2\sum_{j=1}^{K}\Im(\mathbf{s}_j)\Re(\mathbf{s}_j)[\Re(\mathbf{R}_j)\Im(\mathbf{H}_{j,l})+\Im(\mathbf{R}_j)\Re(\mathbf{H}_{j,l})]}{\left(\frac{\sum_{j=1}^{K}a_j^2-b_j^2}{2}\right)^2}\Bigg]
\end{aligned}
$$

We use similar notations to express (V-C)

$$
\begin{aligned}
&\frac{d\,\|\underline{\mathbf{d}}\otimes\underline{\mathbf{s}}-\underline{\mathbf{R}}\|_2^2}{d\Im(\mathbf{x}_l)}\\
&=2\sum_{i=1}^{K}\Re\big(\mathbf{d}_i\mathbf{s}_i-\mathbf{R}_i\big)\frac{d}{d\Im(\mathbf{x}_l)}\Big(\Re\big(\mathbf{d}_i\mathbf{s}_i-\mathbf{R}_i\big)\Big)\\
&+2\sum_{i=1}^{K}\Im\big(\mathbf{d}_i\mathbf{s}_i-\mathbf{R}_i\big)\frac{d}{d\Im(\mathbf{x}_l)}\Big(\Im\big(\mathbf{d}_i\mathbf{s}_i-\mathbf{R}_i\big)\Big)
\end{aligned}
$$

Using (V-C) and (V-C), we find

$$
\begin{aligned}
&\frac{d}{d\Im(\mathbf{x}_l)}\Big(\Re\big(\mathbf{d}_i\mathbf{s}_i-\mathbf{R}_i\big)\Big)=\Re(\mathbf{s}_i)\Big[\cos^2(\varphi)\frac{d\,a_i}{d\Im(\mathbf{x}_l)}\\
&-2a_i\cos(\varphi)\sin(\varphi)\frac{d\,\varphi}{d\Im(\mathbf{x}_l)}+\cos(\varphi)\sin(\varphi)\frac{d\,b_i}{d\Im(\mathbf{x}_l)}\\
&+\cos^2(\varphi)b_i\frac{d\,\varphi}{d\Im(\mathbf{x}_l)}-\sin^2(\varphi)b_i\frac{d\,\varphi}{d\Im(\mathbf{x}_l)}\Big]\\
&-\Im(\mathbf{s}_i)\Big[\sin^2(\varphi)\frac{d\,b_i}{d\Im(\mathbf{x}_l)}+2b_i\cos(\varphi)\sin(\varphi)\frac{d\,\varphi}{d\Im(\mathbf{x}_l)}\\
&+\cos(\varphi)\sin(\varphi)\frac{d\,a_i}{d\Im(\mathbf{x}_l)}+\cos^2(\varphi)a_i\frac{d\,\varphi}{d\Im(\mathbf{x}_l)}\\
&-\sin^2(\varphi)a_i\frac{d\,\varphi}{d\Im(\mathbf{x}_l)}\Big]-\Re(\mathbf{H}_{i,l})\\
&=\Re(\mathbf{s}_i)\times\mathbf{C}-\Im(\mathbf{s}_i)\times\mathbf{D}+\Im(\mathbf{H}_{i,l})
\end{aligned}
$$

Leading to

$$
\begin{aligned}
&\frac{d}{d\Im(\mathbf{x}_l)}\Big(\Im\big(\mathbf{d}_i\mathbf{s}_i-\mathbf{R}_i\big)\Big)\\
&\qquad=\Im(\mathbf{s}_i)\times\mathbf{C}+\Re(\mathbf{s}_i)\times\mathbf{D}-\Re(\mathbf{H}_{i,l})
\end{aligned}
$$

We finally need to express $\frac{d\,a_i}{d\Im(\mathbf{x}_l)}$, $\frac{d\,b_i}{d\Im(\mathbf{x}_l)}$ and $\frac{d\,\varphi}{d\Im(\mathbf{x}_l)}$ $\forall l,i\in[\![1;K]\!]$

$$
\frac{d\,a_i}{d\Im(\mathbf{x}_l)}=-\Re(\mathbf{s}_i)\Im(\mathbf{H}_{i,l})+\Im(\mathbf{s}_i)\Re(\mathbf{H}_{i,l})
$$

$$
\frac{d\,b_i}{d\Im(\mathbf{x}_l)}=\Re(\mathbf{s}_i)\Re(\mathbf{H}_{i,l})+\Im(\mathbf{s}_i)\Im(\mathbf{H}_{i,l})
$$

$$
\begin{aligned}
&\frac{d\,\varphi}{d\Re(\mathbf{x}_l)}=\frac{1}{1+\big(\tan(2\varphi)\big)^2}\sum_{i=1}^{K}\Im(\mathbf{s}_i)\Re(\mathbf{s}_i)\\
&\times\Bigg[\frac{\Big[\Re(\mathbf{H}_{i,l})\Im(\mathbf{R_i})+\Im(\mathbf{H}_{i,l})\Re(\mathbf{R_i})\Big]\frac{\sum_{j=1}^{K}a_j^2-b_j^2}{2}}{\left(\frac{\sum_{j=1}^{K}a_j^2-b_j^2}{2}\right)^2}+\\
&\frac{\Re(\mathbf{R_i})^2\sum_{j=1}^{K}\Im(\mathbf{s}_j)\Re(\mathbf{s}_j)[\Re(\mathbf{R}_j)\Re(\mathbf{H}_{j,l})-\Im(\mathbf{R}_j)\Im(\mathbf{H}_{j,l})]}{\left(\frac{\sum_{j=1}^{K}a_j^2-b_j^2}{2}\right)^2}-\\
&\frac{\Im(\mathbf{R_i})^2\sum_{j=1}^{K}\Im(\mathbf{s}_j)\Re(\mathbf{s}_j)[\Re(\mathbf{R}_j)\Re(\mathbf{H}_{j,l})-\Im(\mathbf{R}_j)\Im(\mathbf{H}_{j,l})]}{\left(\frac{\sum_{j=1}^{K}a_j^2-b_j^2}{2}\right)^2}\Bigg]
\end{aligned}
$$

The results let us implement a C2PO variant with a shared argument for the amplifictation vector. Due to the non-convexity of the function, there is no guarantee to an optimal convergence. The results of the simulations are presented bellow, and use the hyperparameters as in Table III.

D. Analysis of Shared phase C2PO results

The results of the shared phase C2PO are not as good as expected for our simulations. Figure 2 shows an improvement compared to our first implementation, but shows that our second method is outperformed by the amplification by a complex coefficient. However, Figure 3 shows that for $K=5$

and $M = 50$, the Shared phase C2PO is almost as wrong as Quantized ZF. This behavior is shared by our other result with a higher value of M, while the curves for $K = 5$ and $M = 20$ are similar to Figure 2. This under-performance can be explained by multiple factors. The first explanation to the worse results of Shared phase C2PO compared to the original C2PO (but still better results than the first method) could be the hyperparameter tuning. In fact, we noticed than a little nudge on the τ value could greatly increase the SER. For example, going from $\tau = 10^{-2}$ to $\tau = 10^{-3}$ leads to worse performances while maintaining $t_{max} = 100$. Setting $t_{max} = 1000$ solves this problem (performances are better than our first proposed method), implying that a decay of τ would greatly help the algorithm to converge faster, and would prevent the optimization to find local minima. Such an hyperparameter tuning would however be unfeasible manually due to its complexity. A good solution to this issue would be to optimize the hyperparameters as [1] did for C2PO to make a fair comparison of the methods.

The other explanation to this under-performance could be the non-convexity of the function we try to optimize. The expression of the function is more complex and may thus have numerous local minima, from which the C2PO method is not able to exit. This explanation is also backed by the fact that the performances are worse for higher M values. A higher M implies a higher dimension of $\underline{\mathbf{x}}$, meaning more potential local minima. This implies that the global minimum could be diluted in a subspace of local minima with a greater cardinality than for small M values, making it harder for our algorithm to find the optimum. To prevent this issue, we could use gradient descents with momentum or alternatives, as it is effectively used in machine learning.

We noticed a certain pattern in the behavior of the amplification vector $\underline{\mathbf{d}}$ of the Shared phase C2PO when it had a SER $\neq 0$ (in the case where it performs still better than the Positive vector C2PO). In fact, one element is inevitably way smaller than the others.

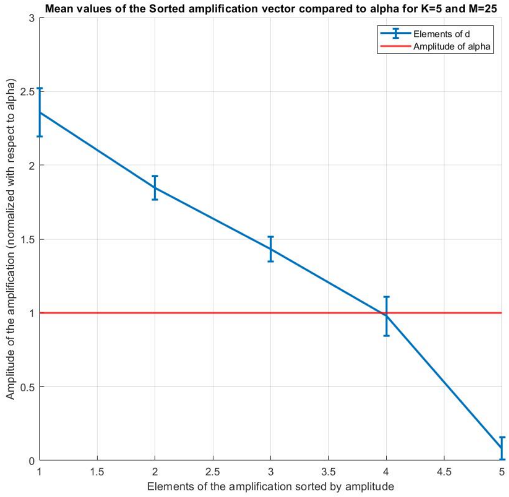

Fig. 4. Comparison between $\underline{\mathbf{d}}$ and α over the whole dataset for $K = 5$ and $M = 25$. The errorbars represent the variance of the sorted element

We retrieved the values of the absolute value of $\underline{\mathbf{d}}$ and α (see II) over the trials with SER $\neq 0$, sorted them and computed the mean over each different vector for $K = 5$ and $M = 25$. The plot on Figure 4 represent the mean of the sorted elements of $\mathbf{d}$ (greatest to lowest) in blue and the mean value of $|\alpha|$ in red. We show here that the element which is transmitted with an error has the smallest absolute value of $\mathbf{d}_i$(right) . We also noticed that some incorrect results showed $d_i < 0$ (for i such as $\mathbf{s}_i \neq \mathbf{R}_i$).

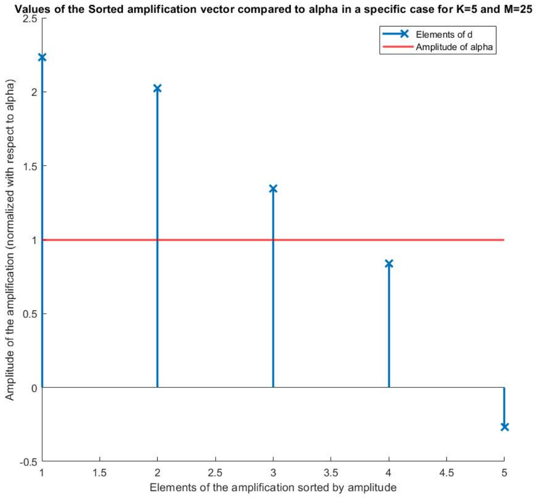

Fig. 5. Comparison between $\underline{\mathbf{d}}$ and α for a specific case

This implies a rotation of π of the specific input, which reinforces the hypothesis of bad hyperparameters for these cases (the algorithm may not have converged yet) as such an amplification is counter intuitive. The general case of $\mathbf{d}_i$ being too small reinforces our hypothesis of a local minimum.

VI. Conclusion

We studied the mathematical approaches of two methods for massive MIMO 1-bit precoding. We showed that our first method is robust even without a thorough hyperparameter tuning, but is outclassed by the method proposed by [3] and [1]. On the opposite, the second method we proposed can not be seen as a major improvement for Quantized ZF for high values of M. This issue may be related to the complexity of the proposed function and to our implementation using forward-backward splitting (through C2PO), which is an efficient tool designed for convex optimization, but has no guarantee of optimal convergence in our case.

VII. Perspectives

An hyperparameter optimization step is required (for all the tested algorithms) to make a fair evaluation of the different methods. This step could also be kept for the real implementation as the Base Station has a high computing capability. Additionally, if the hyperparameter tuning is not sufficient to improve the performances, another implementation of the Shared phase method is deemed necessary to optimize correctly the proposed function (instead of converging to a local minima). This would mean to implement it with a method suited for non-convex optimisation to find the global minima instead of a local one. This new implementation (other than C2PO) could also be made for every presented method to compare them with our implementations.

We also studied a third method to solve the 1-bit precoding optimization problem with a prior amplification of the input $\underline{\mathbf{s}}$. As the amplifiction by a non-constrained complex vector $\underline{\mathbf{d}}$ leads to a dead end, we tried to constraint each $\mathbf{d}_i$ so that the receiver loses no information in the optimal case. In fact, adding a rotation (complex phase) to the amplification gives no guarantee that the quantized amplified vector $\mathbb{Q}(\underline{\mathbf{d}} \otimes \underline{\mathbf{s}})$ is equal to the initial input vector $\underline{\mathbf{s}}$, i.e. the received vector $\underline{\mathbf{R}}$ may not be equal to $\underline{\mathbf{s}}$ even if we optimize correctly $\|\underline{\mathbf{d}} \otimes \underline{\mathbf{s}} - \underline{\mathbf{R}}\|_2^2$. This seems however not to be an issue for the algorithms we tested (empirical observation) when using a good initialization (MRT or Quantized ZF). Optimizing $\underline{\mathbf{s}}$ with this constraint however could help to express the optimization problem in a feasible way. For a Q-PSK constellation, this would mean to force $\arg(\mathbf{d}_i) \in] - \frac{\pi}{4}; \frac{\pi}{4}[$ and $\|\mathbf{d}_i| > 0$ as a different value would give $\mathbb{Q}(\underline{\mathbf{d}} \otimes \underline{\mathbf{s}})_i \neq \mathbf{s}_i$. Alternatively, we can express this as:

$$\mathbf{d}_i = r_i \; e^{j\varphi_i} = a_i + j \; b_i$$

$$r_i > 0 \;\; \& \;\; \varphi_i \in] - \frac{\pi}{4}; \frac{\pi}{4}[\quad \Leftrightarrow \quad |b_i| < a_i$$

It may also be interesting to add a safety margin (restrict φ_i to a smaller range and/or r_i to higher values) to add robustness against noise and miss-calculations.

References

[1] A. Balatsoukas-Stimming, O. Castañeda, S. Jacobsson, G. Durisi, and C. Studer. Neural-network optimized 1-bit precoding for massive MU-MIMO. *CoRR*, abs/1903.03718, 2019.

[2] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

[3] O. Castañeda, S. Jacobsson, G. Durisi, M. Coldrey, T. Goldstein, and C. Studer. 1-bit massive MU-MIMO precoding in VLSI. *CoRR*, abs/1702.03449, 2017.

[4] T. Goldstein, C. Studer, and R. G. Baraniuk. A field guide to forward-backward splitting with a FASTA implementation. *CoRR*, abs/1411.3406, 2014.

[5] E. G. Larsson, O. Edfors, F. Tufvesson, and T. L. Marzetta. Massive mimo for next generation wireless systems. *IEEE Communications Magazine*, 52(2):186–195, 2014.

[6] A. K. Saxena, I. Fijalkow, and A. L. Swindlehurst. Analysis of one-bit quantized precoding for the multiuser massive mimo downlink. *IEEE Transactions on Signal Processing*, 65(17):4624–4634, 2017.

VIII. Appendix

The code and detailed calculations can be found at https://github.com/Flopp88/Non-negative-vector-optimization-application-to-precoding-for-massive-antennas