A Performance Analysis of ADMM Applied to Coordinated Beamforming

Eduardo de O. Cavalcante, Yuri C. B. Silva, Walter C. Freitas Jr.

***Abstract*— This paper presents a performance analysis for a centralized solution to the Coordinated Beamforming Optimization Problem (CBOP), that uses an Alternating Direction Method of Multipliers (ADMM) approach in order to speed up calculations and to provide certificates of infeasibility. The ADMM algorithm is said to solve large-scale problems with modest accuracy, within a reasonably low amount of time, and to allow calculation in a parallel fashion, which would be very desirable for C-RAN networks. In this paper, the ADMM approach is applied to a coordinated beamforming scenario, but differently from other works, without considering joint transmission. The performance of the ADMM solution is analyzed by comparing its simulation results with a well known approach that uses a Semidefinite Programming (SDP) embedding based on semidefinite relaxation. It is shown that the ADMM approach requires significantly less time to solve large-scale problems when modest accuracy is required.**

***Keywords*— ADMM, Coordinated Beamforming, Optimization.**

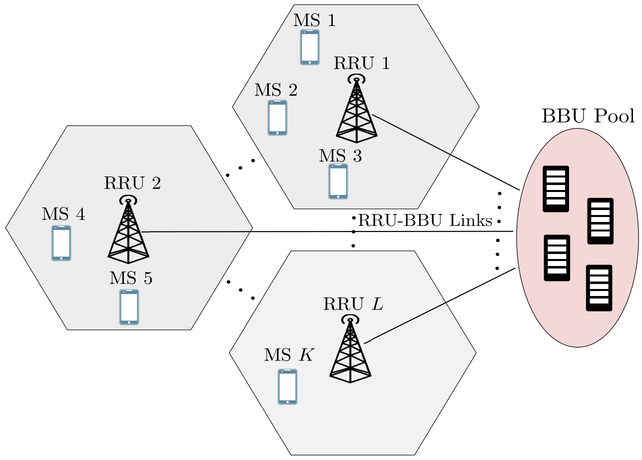


Fig. 1. Illustration of an example scenario

I. INTRODUCTION

In the years to come, it is expected that the number of users in the mobile systems and their need for capacity will continue to rise [1]. This fact will lead to a huge densification of the wireless networks, with the increase in the number of cells on a given area, in order to better support the higher Quality of Service (QoS) requirements, for these reasons the employment of Ultra Dense Networks is one of the leading ideas for the future of mobile communications [2].

The larger number of cells and users combined with the more strict QoS requirements in future mobile systems will lead to optimization problems with a large number of dimensions, variables and constraints, as well as very strict latency requirements. However, conventional algorithms that solve the Coordinated Beamforming Optimization Problem (CBOP) have their performance demonstrated for systems with small numbers of cells, typically less than a dozen [4] [5] [6]. This fact motivates the search for optimization methods that could be used to solve the CBOP in large-scale scenarios.

A promising method to optimize the beamforming was proposed by [3], which uses an Alternating Direction Method of Multipliers (ADMM) based algorithm. This method is said to solve large-scale problems with moderate accuracy, within a reasonably low amount of time. ADMM also allows parallel processing, what is very desirable for C-RAN systems, in which the baseband processing is centralized and shared among sites in a virtualized Baseband Unit (BBU) pool [7].

Eduardo de O. Cavalcante, Yuri C. B. Silva, Walter C. Freitas Jr.. Wireless Telecommunications Research Group (GTEL), Federal University of Ceará, Fortaleza, Brazil, E-mails: {eduardo, yuri, walter}@gtel.ufc.br.

ADMM is described in [8] as a simple but powerful algorithm that is well suited to distributed convex optimization problems and takes the form of a decomposition-coordination procedure, in which the solutions to small sub-problems are coordinated to find a solution to a large global problem.

In this paper we present an ADMM-based solution for the CBOP, based on [3], but without considering joint transmission, which allows for a fair comparison with the well known Semidefinite Programming (SDP) solution for the CBOP, presented by [9]. The main contributions of this paper are the performance analysis, which assesses the scalability of the ADMM solution applied to different network configurations, and the convergence analysis, which shows how the ADMM method iterates to find an optimal solution.

The rest of the paper is organized as follows. Section II defines the problem and presents the ADMM-based solution for the CBOP. Section III shows and analyzes the simulation results. Finally, Section IV draws the main conclusions.

II. ADMM BASED COORDINATED BEAMFORMING

The scenario where beamforming is applied is exemplified in Figure 1. It is formed by a set of K single-antenna Mobile Stations (MSs) and L Remote Radio Units (RRUs) with N antennas, managed by one single BBU pool. The signal received by a MS k, which is assigned to a RRU l, is $y_k \in \mathbb{C}$, and it is expressed as

$$y_k = \underbrace{\mathbf{h}_{kl}^H \mathbf{v}_{kl} s_k}_{\text{desired signal}} + \underbrace{\sum_{\substack{j=1 \\ j\neq k}}^{K} \mathbf{h}_{kl}^H \mathbf{v}_{jl} s_j}_{\text{intra-RRU interference}} + \underbrace{\sum_{\substack{i=1 \\ i\neq l}}^{L} \sum_{\substack{j=1 \\ j\neq k}}^{K} \mathbf{h}_{ki}^H \mathbf{v}_{ji} s_j}_{\text{inter-RRU interference}} + n_k, \ \forall k, \quad (1)$$

where $\mathbf{h}_{kl}^H \in \mathbb{C}^{N\times 1}$ is the channel vector between RRU l and MS k, $\mathbf{v}_{kl} \in \mathbb{C}^{N\times 1}$ is the beamforming vector from RRU l to MS k (we assume that if a user j is not assigned to an RRU i, then $\mathbf{v}_{ji}$ is a zero vector), $s_k \in \mathbb{C}$ is the data sent to user k, which is assumed to have unit variance, and $n_k \in \mathbb{C}$ is the additive white Gaussian noise at user k, with variance σ_k^2.

The Signal-to-Interference-plus-Noise Ratio (SINR) of each MS is given by

$$\text{SINR}_k = \frac{|\mathbf{h}_{kl}^H \mathbf{v}_{kl}|^2}{\sum\limits_{\substack{j=1\\ j\neq k}}^{K} |\mathbf{h}_{kl}^H \mathbf{v}_{jl}|^2 + \sum\limits_{\substack{i=1\\ i\neq l}}^{L} \sum\limits_{\substack{j=1\\ j\neq k}}^{K} |\mathbf{h}_{ki}^H \mathbf{v}_{ji}|^2 + \sigma_k^2}, \ \forall k. \tag{2}$$

We define the CBOP as a sum power minimization problem, with independent SINR constraints, γ_k, for each receiver. This problem can be conveniently written as an optimization task as

$$\begin{aligned} &\text{minimize} && \sum_{i=1}^{L}\sum_{j=1}^{K} \|\mathbf{v}_{ji}\|_2^2 \\ &\text{subject to} && \text{SINR}_k \geq \gamma_k, \ \forall k. \end{aligned} \tag{3}$$

However, in order to use the ADMM algorithm to solve this problem, we rewrite it into another form, called Homogeneous Self-Dual (HSD) embedding [10]. This new embedding is obtained by transforming a *primal-dual* pair of a convex conic optimization problem into an equivalent feasibility problem of finding a nonzero point in the intersection of a subspace and a cone. HSD has the feature of providing certificates of infeasibility when the solution is not feasible. In this sense, Problem (3) should be first rewritten in a standard conic form to then be transformed into the equivalent HSD embedding.

A. *Conic Reformulation and Matrix Stuffing*

The first step is to transform Problem (3) into a conic problem. The desired standard Second-Order Cone Programming (SOCP) form can be expressed as

$$\begin{aligned} &\text{minimize} && \boldsymbol{c}^T \boldsymbol{\nu} \\ &\text{subject to} && \mathbf{A}\boldsymbol{\nu} + \boldsymbol{\mu} = \boldsymbol{b} \\ &&& (\boldsymbol{\nu}, \boldsymbol{\mu}) \in \mathbb{R}^n \times \mathcal{K}, \end{aligned} \tag{4}$$

with variables $\boldsymbol{\nu} \in \mathbb{R}^n$ and $\boldsymbol{\mu} \in \mathbb{R}^m$. $\boldsymbol{A} \in \mathbb{R}^{m\times n}$, $\boldsymbol{b} \in \mathbb{R}^m$ and $\boldsymbol{c} \in \mathbb{R}^n$ are the problem parameters and $\mathcal{K}$ is the cone

$$\mathcal{K} = \mathcal{Q}^{m_1} \times \cdots \times Q^{m_q} \times \{0\}^r, \tag{5}$$

where

$$Q^p = \{(t, \mathbf{x}) \in \mathbb{R} \times \mathbb{R}^{p-1} |\, \|\mathbf{x}\|_2 \leq t\} \tag{6}$$

is the second-order cone of dimension p, Q^1 is defined as the cone of nonnegative reals, $\mathbb{R}_+$, and the sum of all the cone dimensions, $m_1 + \cdots + m_q + r$, is equal to m.

In order to speed up the transformation of (3) into (4), [3] also proposed the use of the previous knowledge about the network structure, by using a technique called matrix stuffing. It works by initially generating a fixed structure that maps the original problem dimensions (RRUs, MSs, antennas) into the new conic form, and then, at running time, by copying the problem data to the previously generated structure.

The process starts by writing an equivalent problem to (3). Now, instead of minimizing the sum of the squared norms of $\mathbf{v}_{ji}$, we introduce a relaxed minimization of the norm of the vector $\mathbf{v}$, formed by a composition of all $\mathbf{v}_{ji}$, $\mathbf{v} \triangleq [\mathbf{v}_1^T, \ldots, \mathbf{v}_K^T]^T \in \mathbb{R}^{LNK}$, where $\mathbf{v}_k \triangleq [\mathbf{v}_{k1}^T, \ldots, \mathbf{v}_{kL}^T]^T \in \mathbb{R}^{LN}$. Both objective functions have the same minimum and both are convex. The SINR constraints are still the same, but expressed in a form more convenient to our purpose. Thus, Problems (3) and (7) are equivalent for real parameters[1] .

$$\begin{aligned} &\text{minimize} && x_0 \\ &\text{subject to} && \|\mathbf{v}\|_2 \leq x_0 \\ &&& \|\mathbf{C}_k \mathbf{v} + \mathbf{g}_k\|_2 \leq \beta_k \mathbf{r}_k^T \mathbf{v}, \ k = 1 \ldots K, \end{aligned} \tag{7}$$

where $\mathbf{C}_k = [\text{blkdiag}\{\tilde{\mathbf{h}}_{k\tilde{1}}, \ldots, \tilde{\mathbf{h}}_{k\tilde{K}}\}, \mathbf{0}_{LNK}]^T \in \mathbb{R}^{(K+1)\times LNK}$, $\mathbf{r}_k = [\mathbf{0}_{(k-1)LN}^T, \tilde{\mathbf{h}}_{k\tilde{k}}^T, \mathbf{0}_{(K-k)LN}^T]^T \in \mathbb{R}^{LNK}$, $\beta_k = \sqrt{1 + 1/\gamma_k}$, $\mathbf{g}_k = [\mathbf{0}_K^T, \sigma_k]^T \in \mathbb{R}^{K+1}$ and $\tilde{\mathbf{h}}_{k\tilde{i}} = [\mathbf{0}_{(\tilde{i}-1)N}^T, \mathbf{h}_{k\tilde{i}}^T, \mathbf{0}_{(L-\tilde{i})N}^T]^T \in \mathbb{R}^{LN}$, $\tilde{i}$ represents the index of the RRU to which user i is allocated to, and blkdiag$\{\}$ is the block diagonal matrix formed by the sub-matrices on its argument.

Problem (7) is already a conic problem, thus a direct mapping of its parameters can be used to form the fixed structure of the parameters in the standard SOCP form of Problem (4). This way, the structure of the variables and parameters in Problem (4), that are generated from the conic mapping of Problem (3), can be written as

$$\boldsymbol{\nu} = [x_0, t_0^1, \ldots, t_0^K, \mathbf{v}], \quad \mathbf{c} = [1, \mathbf{0}_{n-1}],$$

$$\mathbf{A} = \left[\begin{array}{c|c|c} -1 & & -\mathbf{I}_{LNK} \\ \hline & 1 & -\beta_1 \mathbf{r}_1^T \\ & \ddots & \vdots \\ & 1 & -\beta_K \mathbf{r}_K^T \\ \hline & -1 & \\ & & -\mathbf{C}_1 \\ \hline & \vdots & \vdots \\ \hline & -1 & \\ & & -\mathbf{C}_K \end{array}\right], \mathbf{b} = \left[\begin{array}{c} 0 \\ \mathbf{0}_{LNK} \\ \hline 0 \\ \vdots \\ 0 \\ \hline 0 \\ \mathbf{g}_1 \\ \hline \vdots \\ \hline 0 \\ \mathbf{g}_k \end{array}\right], \tag{8}$$

$$\mathcal{K} = \mathcal{Q}^{LNK+1} \times \underbrace{\mathcal{Q}^1 \times \cdots \times \mathcal{Q}^1}_{K} \times \underbrace{\mathcal{Q}^{K+2} \times \cdots \times \mathcal{Q}^{K+2}}_{K},$$

where x_0 and t_0^k are slack variables.

Matrix Stuffing works by initially defining the dimensions of the problem, n and m, creating the general matrices and vectors in the form of $\mathbf{A}$, $\mathbf{b}$ and $\mathbf{c}$, and defining the cone dimensions, $\mathcal{K}$. These elements will be used as the "skeleton" of the problem, over which the run time parameters, $\mathbf{C}_k$, $\mathbf{r}_k$ and $\mathbf{g}_k$ will be copied at each problem instance.

[1]Complex parameters need to be rearranged into a real form, as indicated in the appendix section of [3].

B. Homogeneous Self-Dual Embedding

The HSD embedding is obtained by transforming the *primal-dual* pair of the convex conic optimization problem into an equivalent feasibility problem of finding a nonzero point in the intersection of a subspace and a cone [10]. A very important feature of the HSD embedding is that it can provide certificates of infeasibility if a *primal-dual* pair is not solvable.

The dual of Problem (4) is

$$\begin{array}{ll} \text{maximize} & -\mathbf{b}^T\boldsymbol{\eta} \\ \text{subject to} & -\mathbf{A}^T\boldsymbol{\eta} + \boldsymbol{\lambda} = \mathbf{c} \\ & (\boldsymbol{\lambda},\boldsymbol{\eta}) \in \{0\}^n \times \mathcal{K}^*, \end{array} \tag{9}$$

where $\boldsymbol{\lambda} \in \mathbb{R}^n$ and $\boldsymbol{\eta} \in \mathbb{R}^m$ are the dual variables and $\mathcal{K}^*$ is the dual cone of $\mathcal{K}$. We define p^* as the optimal primal solution, and d^* as the optimal dual, and we assume that strong duality holds, i.e. $p^* = d^*$, in all cases.

If strong duality holds, the problem is feasible and the following Karush-Kuhn-Tucker (KKT) conditions (embedded into a single system of equations) are necessary and sufficient for optimality

$$\underbrace{\begin{bmatrix} \boldsymbol{\lambda} \\ \boldsymbol{\mu} \\ \kappa \end{bmatrix}}_{\mathbf{y}} = \underbrace{\begin{bmatrix} 0 & \mathbf{A}^T & \mathbf{c} \\ -\mathbf{A} & 0 & \mathbf{b} \\ -\mathbf{c}^T & -\mathbf{b}^T & 0 \end{bmatrix}}_{\mathbf{Q}} \underbrace{\begin{bmatrix} \boldsymbol{\nu} \\ \boldsymbol{\eta} \\ \tau \end{bmatrix}}_{\mathbf{x}}, \tag{10}$$
$$(\boldsymbol{\nu},\boldsymbol{\mu},\boldsymbol{\lambda},\boldsymbol{\eta},\tau,\kappa) \in \mathbb{R}^n \times \mathcal{K} \times \{0\}^n \times \mathcal{K}^* \times \mathbb{R}_+ \times \mathbb{R}_+.$$

The inclusion of the variables τ and κ is made to allow a feasibility analysis based on their values.

The HSD embedding is, then, written as a feasibility problem to find a nonzero point that satisfies the KKT conditions

$$\begin{array}{ll} \text{find} & (\mathbf{x},\mathbf{y}) \\ \text{subject to} & \mathbf{y} = \mathbf{Q}\mathbf{x} \\ & (\mathbf{x},\mathbf{y}) \in \mathcal{C} \times \mathcal{C}^*, \end{array} \tag{11}$$

where $\mathcal{C} = \mathbb{R}^n \times \mathcal{K}^* \times \mathbb{R}_+$ and $\mathcal{C}^* = \{0\}^n \times \mathcal{K} \times \mathbb{R}_+$.

This approach is shown to be homogeneous and self-dual by [10], which also proposed an ADMM algorithm to solve it, as discussed in the following section.

C. ADMM Algorithm For The HSD Embedding

The HSD Embedding in Problem (11) can be reformulated into an ADMM form, by splitting the optimization variables in $(\mathbf{x},\mathbf{y})$ and $(\tilde{\mathbf{x}},\tilde{\mathbf{y}})$, with the objective function separable across this splitting

$$\begin{array}{ll} \text{minimize} & \mathcal{I}_{\mathcal{C}\times\mathcal{C}^*}(\mathbf{x},\mathbf{y}) + \mathcal{I}_{\mathbf{Q}\tilde{\mathbf{x}}=\tilde{\mathbf{y}}}(\tilde{\mathbf{x}},\tilde{\mathbf{y}}) \\ \text{subject to} & (\mathbf{x},\mathbf{y}) = (\tilde{\mathbf{x}},\tilde{\mathbf{y}}), \end{array} \tag{12}$$

where $\mathcal{I}_{\mathcal{S}}(z)$ is the indicator function of the set $\mathcal{S}$, that assumes value 1 if it is applied to an element of the set, and goes to $+\infty$ otherwise. The direct application of the ADMM algorithm [10] to problem (12) yields

$$\begin{aligned} (\tilde{\mathbf{x}}^{k+1},\tilde{\mathbf{y}}^{k+1}) &= \Pi_{\mathbf{Qx}=\mathbf{y}}(\mathbf{x}^k + \mathbf{s}^k, \mathbf{y}^k + \mathbf{r}^k) \\ \mathbf{x}^{k+1} &= \Pi_{\mathcal{C}}(\tilde{\mathbf{x}}^k - \mathbf{s}^k) \\ \mathbf{y}^{k+1} &= \Pi_{\mathcal{C}^*}(\tilde{\mathbf{y}}^k - \mathbf{r}^k) \\ \mathbf{s}^{k+1} &= \mathbf{s}^k - \tilde{\mathbf{x}}^{k+1} + \mathbf{x}^{k+1} \\ \mathbf{r}^{k+1} &= \mathbf{r}^k - \tilde{\mathbf{y}}^{k+1} + \mathbf{y}^{k+1}, \end{aligned} \tag{13}$$

where $\mathbf{s}$ and $\mathbf{r}$ are the dual variables for the equality constraints on $\mathbf{x}$ and $\mathbf{y}$, respectively, and $\Pi_{\mathcal{S}}(\mathbf{z})$ is the Euclidean projection of $\mathbf{z}$ over the set $\mathcal{S}$.

As Problem (12) is self-dual, we can verify that $\mathbf{s}^k = \mathbf{y}^k$ and $\mathbf{r}^k = \mathbf{x}^k$ if they are equally initialized ($\mathbf{s}^0 = \mathbf{y}^0$, $\mathbf{r}^0 = \mathbf{x}^0$) [10]. This fact allows us to eliminate the dual variable updates in (13). It also enables the change of the update of $\mathbf{y}^{k+1}$ using a cone projection by the equivalent $\mathbf{s}^{k+1}$ update, which is typically computationally cheaper than a projection. The resulting algorithm is

$$\begin{aligned} \tilde{\mathbf{x}}^{k+1} &= \Pi_{\mathbf{Qx}=\mathbf{y}}(\mathbf{x}^k + \mathbf{y}^k) \\ \mathbf{x}^{k+1} &= \Pi_{\mathcal{C}}(\tilde{\mathbf{x}}^k - \mathbf{y}^k) \\ \mathbf{y}^{k+1} &= \mathbf{y}^k - \tilde{\mathbf{x}}^{k+1} + \mathbf{x}^{k+1}. \end{aligned} \tag{14}$$

Another simplification that can be done is the projection performed in the first step of the algorithm, replacing it by its matrix form $\Pi_{\mathbf{Qx}=\mathbf{y}}(\mathbf{x}^k + \mathbf{y}^k) = (\mathbf{I} + \mathbf{Q})^{-1}(\mathbf{x}^k + \mathbf{y}^k)$.

The algorithm is then formed by three steps. The first and second steps involve projections over the set $\mathbf{Qx} = \mathbf{y}$ and the cone $\mathcal{C}$, respectively, and the third is a simple running sum of the error at the $(k+1)$th iteration.

III. SIMULATION RESULTS

For the ADMM simulation, we apply the matrix stuffing procedure by creating the base structure for the standard conic problem, and in each realization we copy the problem data into this structure. Afterwards, we use a numerical optimization package based on [10], called SCS, which solves conic programs in the form of Problem (4), using the ADMM algorithm for HSD embedding as defined in sections II-B and II-C.

The SDP formulation [9], used to assess the ADMM performance, is a well known semidefinite relaxed version of Problem (3). To solve it we use the interior point toolbox *SeDuMi*, from the CVX modeling framework.

Another important feature of the SDP formulation is its ability to determine an optimal RRU-MS allocation [9]. In this sense, both approaches use the same allocation, provided by SDP. However, when SDP is not capable of converging in a reasonable amount of time, we use a minimum RRU-MS distance based allocation for the ADMM approach.

The simulations parameters are shown in Table I.

The first simulation is made to assess the time performance for scenarios with 5 RRUs and 10 MSs, varying the number of antennas. For each of the instances, the simulation was run 100 times and the average of the resulting times is analyzed. The results of this simulation are shown in Figure 2.

In Figure 2 it can be seen that the ADMM based algorithm requires significantly less time than SDP in scenarios with large numbers of antennas. Additionally, SDP is not capable

TABLE I
SIMULATION PARAMETERS

Parameters	Value
Number of MS antennas	1
Scenario Area	$500\,\text{m}\times 500\,\text{m}$
Min. RRU-to-RRU distance	$40\,\text{m}$
Min. RRU-to-MS distance	$3\,\text{m}$
RRU and MS positioning	uniformly random
Noise Density	$-174\,\text{dBm/Hz}$
Min. SINR at each MS	$0.9\,\text{dB}$
Path-loss	$30.6 + 36.7\log_{10}(d)$ dB
Channel Model	IID

to provide solutions within a reasonable amount of time for scenarios with more than 60 antennas, while ADMM stills very time efficient in these cases. However, the highlighted part of Figure 2 shows that when the number of antennas is equal to or less than 5 the ADMM algorithm is slower than SDP, and that the demanded ADMM time decreases when the number of antennas increases from 2 to 5.

Instances of the problem with less antennas offer less degrees of freedom for the beamforming, which makes it a hard task to find a combination of beamforming vectors that is good enough to meet the SINR constraints. However, when the number of antennas is larger, the solution will have to deal with more variables, and then the problem will have larger scale, which would also increase the time requirement.

An interpretation for the behavior shown by the ADMM based algorithm is related to this trade-off and the fact that, as stated before, ADMM shows slow convergence for scenarios with high accuracy requirements, but it deals well with large-scale problems.

A problem with a very small number of antennas and very strict constraints, can be seen as equivalent to a problem with high accuracy requirements, since in order to find a specific solution that can meet very strict constraints, by only using very few degrees of freedom, it is necessary reach a really accurate solution. In this sense, ADMM may be slower when dealing with scenarios with a very small number of antennas. However, when the number of antennas is large enough to provide the minimum required amount of degrees of freedom, such impact is reduced and the ADMM strength on dealing well with large-scale problems is reflected in the results.

The next simulation was made by varying the number of RRUs, setting the number of MSs as twice the number of RRUs for each scenario. The results are shown in Figure 3, where the numbers of RRU antennas used in each simulation are indicated in the legend box.

In Figure 3 we see that for all cases the solving time increases when the number of RRUs increases, what makes sense as we are adding more problems to be solved. We also see that when the number of antennas increases from 2 to 5 the demanded SDP time increases, while the ADMM time decreases as in the previous simulation. When the number of antennas is less than 5, ADMM is more time demanding than SDP in all cases, however, the scenario with 5 antennas is the turning point, where ADMM spends more time in scenarios with less than 9 RRUs, but less time with more cells. With 10

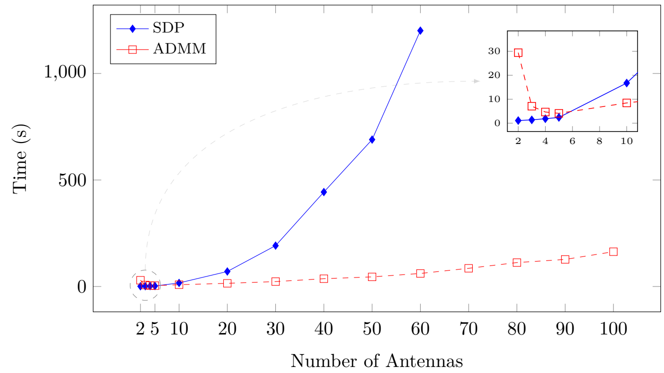


Fig. 2. Performance for 5 RRUs and 10 MSs

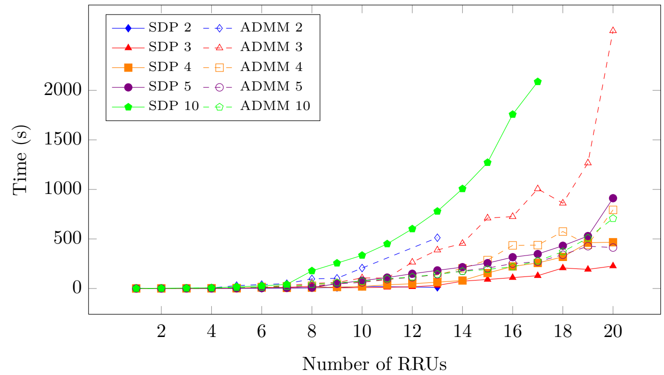


Fig. 3. Performance for x RRUs with y antennas and $2x$ MSs

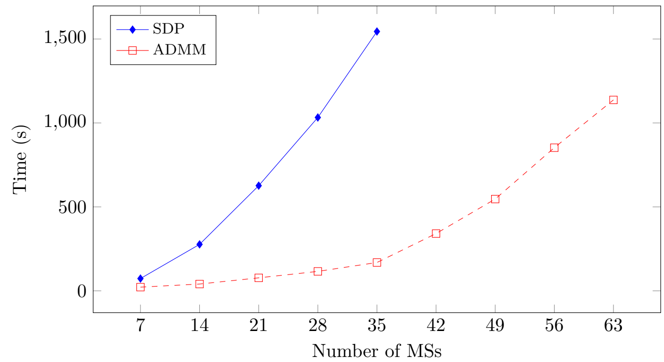
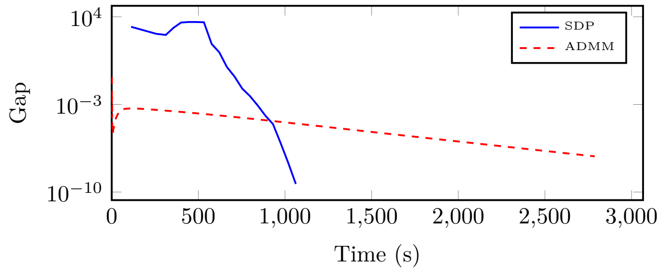


Fig. 4. Performance for 7 RRUs with 20 Antennas.

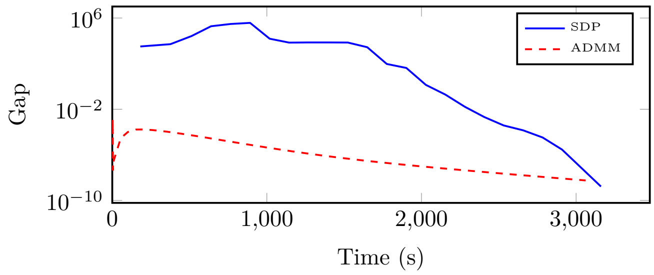
antennas, ADMM shows a much better result than SDP, and it can converge in more cases, as SDP failed to converge with more than 17 RRUs.

Another simulation was made by keeping the numbers of RRUs and antennas fixed, while varying the number of users. In Figure 4 we show the results for the scenario with 7 RRUs with 20 antennas each. It is seen that in all cases ADMM need less time than SDP to converge, and that both convergence times increase when more users are added. However, SDP did not converge in a reasonable amount of time for cases with more than 35 MSs.

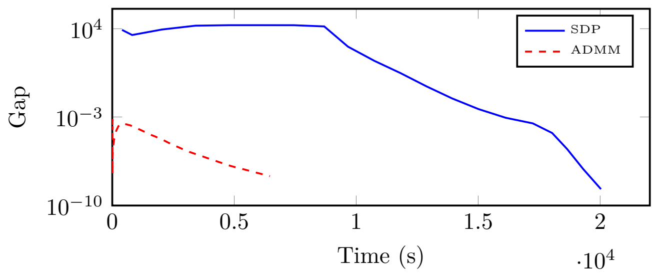
In order to better demonstrate and understand the statement that affirms that ADMM provides modest accuracy within a reasonably low amount of time, but fails to provide high accuracy in comparison to other methods, we analyze its



(a) 5 antenas



(b) 10 antenas



(c) 20 antenas

Fig. 5. Gaps with 4 RRUs and 8 MSs.

convergence behavior in different scenarios. This is done by an inspection of the primal-dual solution gap curve from both approaches, ADMM and SDP. The gap is used for both methods as an accuracy measure and parameter for the stopping criteria, since in convex problems the point with the smallest dual gap is the optimal point.

Figure 5 shows the gap curves for one realization with 8 MSs and 4 RRUs using 5, 10 or 20 antennas. In these figures, the gap is simply defined as the modulus of the difference between the squared primal and dual objectives.

It can be seen in Figure 5 that ADMM required more time to converge than SDP only in the case with 5 antennas, in the other scenarios ADMM converged faster, and the time difference increases when more antennas are added, what agrees with the previous results. The interesting result in these figures is that ADMM shows a smoother convergence rate than SDP, i.e the gap reduction happens in an almost linear rate after a small time from the start of the iterations, however, the initial gap values are much smaller than in SDP, what makes ADMM faster, when less accurate solutions are needed.

For example, in the case with 5 antennas, if the gap required by the stopping criterion were 10^{-4}, ADMM would have required less time to reach this point and provide a solution. When larger-scale problems are analyzed (10 and 20 antennas), ADMM scaled well and provided the required accuracy faster than SDP, however, if more accurate solutions are needed, ADMM may require more time than SDP.

IV. Conclusions

In this paper we have presented a performance analysis of a centralized ADMM based solution for the CBOP, which aims to scale well to large-scale problems, providing modest accuracy within a reasonable amount of time.

The simulation results showed that the ADMM based method was able to provide solutions for large-scale scenarios in much less time than the SDP based method. However, for the scenarios with small numbers of antennas, when compared with the number of users, ADMM required more time to converge than SDP. This result corroborates the statement that ADMM has fast convergence for modest accuracy requirement, but it fails to rapidly provide high accuracy results.

Another possibility of ADMM is its capability of performing the projection steps in a parallel form, what would represent even greater time reductions in C-RAN systems, where this processing can be distributed in multiple BBUs.

Acknowledgements

This work was supported by the Innovation Center Ericsson Telecomunicações S.A., Brazil, under EDB/UFC.42 and UFC.45 Technical Cooperation Contracts. Eduardo de O. Cavalcante would like to acknowledge CAPES for the scholarship support.

References

[1] ERICSSON, " Ericsson mobility report" June 2017.
[2] M. Kamel, W. Hamouda and A. Youssef, " Ultra-dense networks: a survey" *IEEE Communications Surveys & Tutorials*, v. 18, pp. 2522–2545, 2016.
[3] Y. Shi, J Zhang, B. O'Donoghue and K. B. Letaief, "Large-scale convex optimization for dense wireless cooperative networks" *IEEE Transactions on Signal Processing*, v. 63, pp. 4729–4743, 2015.
[4] H. Dahrouj and W. Yu, "Coordinated linear beamforming in downlink multi-cell wireless networks" *IEEE Transactions on Wireless Communications*, v. 9, no. 4, pp.1451–1461, 2010.
[5] L. Venturino, N. Prasad, and X. Wang, "Coordinated beamforming for the multicell multi-antenna wireless system" *IEEE Transactions on Wireless Communications*, v. 9, no. 5, pp.1748–1759, 2010.
[6] L. Venturino and S. Buzzi, "Energy-efficient coordinated beamforming in downlink OFDMA cellular networks" *Statistical Signal Processing (SSP), 2014 IEEE Workshop on*, pp.492–495, 2014.
[7] A Checko, H.L. Christiansen, Y. Yan, L Scolari, G. Kardaras, M. S. Berger and L. Dittmann, "Cloud RAN for mobile networks: a technology overview". *IEEE Communications surveys & tutorials*, v. 17, pp. 405–426, 2015.
[8] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers". *Foundations and Trends in Machine Learning*, v. 3, pp. 1–122, 2011.
[9] A.B. Gershman, N.D. Sidiropoulos, S Shahbazpanahi, M. Bengtsson and B Ottersten, "Convex optimization-based beamforming". *IEEE Signal Processing Magazine*, v. 27, pp. 62–75, 2010.
[10] B. O'Donoghue, E. Chu, N. Parikh and S. Boyd, "Conic optimization via operator splitting and homogeneous self-dual embedding" *Journal of Optimization Theory and Applications*, v. 169, pp. 1042–1068, 2016.
[11] E. Chu, N. Parikh A. Domahidi, and S. Boyd, "Code generation for embedded second-order cone programming" in *Proc. European Control Conference (ECC)*, pp. 1547–1552, 2013.