

Efficient Algorithms with Modified QR Decomposition for Distributed Interference Alignment Based on Cognitive Radio

Hufei Zhu, Bin Li (*Member, IEEE*)

Dept. of Corporate Research,
Huawei Technologies Co., Ltd.,
Shenzhen, P.R. China

Wen Chen (*Member, IEEE*)

Dept. of Electronic Engineering,
Shanghai Jiao Tong Univ.,
Shanghai, P.R.China

Abstract—We propose efficient minor subspace tracking (MST) algorithms to implement Distributed Interference Alignment (IA) based on Cognitive Radio (CR) that employs Interference Subspace Tracking (IST). The proposed MST algorithms utilize modified QR decomposition that decomposes a matrix into a unitary matrix and a lower triangular matrix. With respect to the existing practical and efficient MST algorithm for IST-IA, the proposed MST algorithms require less computational complexity.

Keywords—Interference Alignment (IA); Distributed IA; Cognitive Radio (CR); Interference Subspace Tracking (IST)

I. INTRODUCTION

Recently, interference alignment (IA) was proposed for wireless networks [1], which can theoretically achieve the capacity that is much higher than previously believed [1]. IA designs interfering signals from different sources such that they “cast overlapping shadows” on the signal subspace of interfered receivers. Then part of the signal space at the receiver is free from interference. In the canonical example of IA, every user is able to access one half of the spectrum free from interference from other users, regardless of the number of interferers [1].

To obtain closed form solutions, the IA schemes in [1] require global channel knowledge at all the transmitters, which can be an overwhelming overhead in practice. Furthermore, closed form solutions are difficult to obtain, and have only been found in certain cases. To alleviate these problems, a Distributed IA scheme taking a cognitive radio (CR) approach was proposed in [2], where each transmitter only requires local channel knowledge and the covariance matrix of its effective noise (consisting of the AWGN and the interference from all other users). In the Distributed IA scheme, each transmitter adopts an unselfish approach to minimize the interference to unintended receivers, which is the defining feature of CR [3]. This CR approach then leads to interference alignment [2]. In CR systems, it is a key challenge for a given transmitter to estimate the strength of interference caused at unintended receivers. To meet this challenge, the CR-based Distributed IA scheme utilizes the reciprocity of the two-way communication channel in Time-division duplexing (TDD) networks. Due to reciprocity, the signal subspace with the least interference for a given receiver also causes the least interference to other users during the reciprocal network transmission.

In [4], the Interference Subspace Tracking IA (IST-IA) algorithm was proposed, as a practical solution to implement the Distributed IA scheme of [2] in a cellular communication system. In the IST-IA algorithm, each transmitter requires neither local channel knowledge nor the covariance matrix of its effective noise, and only utilizes a few received signal vectors to track the beamforming matrices. With respect to the Distributed IA algorithm in [2], the IST-IA algorithm [4] provides a similar performance, and requires much less system overhead and computational complexity. Recently a practical and efficient MST algorithm for IST-IA was deduced [5] to utilize the parallelizable matrix-vector products instead of the back and forward substitution in [4], which is an inherent serial process unsuitable for the parallel implementation [6]. On the other hand, the computational complexity of the algorithm in [5] is less than that in [4].

In this paper, we propose efficient MST algorithms for IST-IA, which require less computational complexity than the MST algorithm in [5]. In the proposed MST algorithms, modified QR decomposition is utilized which decomposes a matrix into a unitary matrix and a lower triangular matrix.

This paper is organized as follows. Section II overviews the system model. Section III describes the existing Distributed IA algorithms. The efficient MST algorithms for IST-IA are proposed in Section IV. Then the implementation complexity is analyzed in Section V. Finally, we make conclusion in Section VI.

In the following sections, $(\bullet)^T$ and $(\bullet)^H$ denote matrix transposition and matrix conjugate transposition, respectively.

II. SYSTEM MODEL

As in [4], we consider a Time Division Duplex (TDD) cellular network with K Base Stations (BSs) and K User Equipments (UEs). All the UEs and BSs are equipped with M and N antennas, respectively. For the uplink transmission (from UEs to BSs), the $N \times 1$ received signal vector $\mathbf{y}^{[k]}$ at BS k ($k=1, 2, \dots, K$) can be represented as

$$\mathbf{y}^{[k]} = \sum_{i=1}^K g_{ki} \mathbf{H}^{[ki]} \mathbf{x}^{[i]} + \mathbf{w}^{[k]}, \quad (1)$$

where g_{ki} is the distance dependent path-loss gain between UE i and BS k ; the $N \times M$ matrix $\mathbf{H}^{[ki]}$ consists of fading channel coefficients between UE i and BS k that are independent and identically distributed block Rayleigh fading, i.e., the coefficients are assumed to be constant during training and data transmission phases; $\mathbf{x}^{[i]}$ is the $M \times 1$ transmit signal vector by UE i with transmit power $E\|\mathbf{x}^{[i]}\|^2 = P$; $\mathbf{w}^{[i]}$ is the $N \times 1$ AWGN noise vector with zero mean and unit variance. On the other hand, for the downlink transmission (from BSs to UEs), the $M \times 1$ received signal vector $\mathbf{z}^{[i]}$ at UE i ($i=1, 2, \dots, K$) can be represented as

$$\mathbf{z}^{[i]} = \sum_{k=1}^K g_{ki} (\mathbf{H}^{[ki]})^T \bar{\mathbf{x}}^{[k]} + \bar{\mathbf{w}}^{[i]}, \quad (2)$$

where $\bar{\mathbf{x}}^{[k]}$ and $\bar{\mathbf{w}}^{[i]}$ are the $N \times 1$ transmit signal vector by BS k and the $M \times 1$ AWGN noise vector, respectively.

BS k only serves UE k in its own cell, for $k=1, 2, \dots, K$. Correspondingly from (1) and (2), it can be seen that in the uplink, BS k receives inter-cell interference from all $K-1$ UEs in other cells, while in the downlink, UE k receives inter-cell interference from all $K-1$ non-serving BSs. As in [4], we focus on uplink transmission in what follows, but the algorithm also applies to downlink.

III. EXISTING DISTRIBUTED IA ALGORITHMS

In this section, we review the Distributed IA algorithm in [2] and the IST-IA algorithms in [4,5].

A. Distributed IA

The Distributed IA scheme [2] is proposed for the TDD interference network. Unlike the prior “selfish” transmission strategy where each transmitter tries to maximize its own transmission signal-to-interference-and-noise ratio (SINR) at the desired receiver [1], the CR-based Distributed IA scheme follows an “unselfish” (i.e. “altruistic”) transmission strategy where each transmitter primarily tries to minimize the interference to other concurrent transmissions. By network duality [7], this “unselfish” strategy only requires the transmitter to find the subspace over which it observes the minimal interference when as a receiver.

The Distributed IA algorithm starts with arbitrary transmit and receive filters $\mathbf{V}^{[k]}$, $\mathbf{U}^{[k]}$, and iteratively update these filters to approach interference alignment. Assume that it works in the system model of Section II. Then in each iteration, there are uplink and downlink transmissions. During the first uplink transmission, UE k ($k = 1, 2, \dots, K$) at the transmitter side precodes the $D \times 1$ symbol vector $\hat{\mathbf{x}}^{[k]} = [\hat{x}_1^{[k]} \ \hat{x}_2^{[k]} \ \dots \ \hat{x}_D^{[k]}]$ by an arbitrary $M \times D$ precoding matrix $\mathbf{V}^{[k]}$, i.e., in (1) we have

$$\mathbf{x}^{[k]} = \mathbf{V}^{[k]} \hat{\mathbf{x}}^{[k]} \quad (3)$$

At the receiver side, BS k applies an $N \times D$ interference suppression matrix $\mathbf{U}^{[k]}$ to obtain

$$\hat{\mathbf{y}}^{[k]} = (\mathbf{U}^{[k]})^H \mathbf{y}^{[k]} \quad (4)$$

In (4), $\mathbf{U}^{[k]}$ is chosen to make the signal $\hat{\mathbf{y}}^{[k]}$ lie in the D -dimensional subspace with the minimal interference. Correspondingly BS k needs to know the interference covariance matrix (including the AWGN), i.e.,

$$\mathbf{Q}^{[k]} = \sum_{i=1, i \neq k}^K \frac{P}{D} \mathbf{H}^{[ki]} \mathbf{V}^{[i]} \mathbf{V}^{[i]H} \mathbf{H}^{[ki]H} + \mathbf{I}_N \quad (5)$$

Then $\mathbf{U}^{[k]}$ is chosen as the D eigenvectors corresponding to the smallest D eigenvalues of $\mathbf{Q}^{[k]}$.

In the downlink transmission, the transmitters and receivers switch roles, while the steps are similar. Firstly the transmitters (BSs) set the precoding matrices

$$\bar{\mathbf{V}}^{[k]} = \mathbf{U}^{[k]} \quad (6)$$

for $k=1, 2, \dots, K$, where $\mathbf{U}^{[k]}$ is calculated in the uplink phase. The receivers (UEs) obtain the interference covariance matrix $\bar{\mathbf{Q}}^{[k]}$. Then the interference suppression matrix $\bar{\mathbf{U}}^{[k]}$ is updated with the D eigenvectors corresponding to the smallest D eigenvalues of $\bar{\mathbf{Q}}^{[k]}$. Correspondingly the precoding matrices for the next uplink phase are updated by

$$\mathbf{V}^{[k]} = \bar{\mathbf{U}}^{[k]} \quad (7)$$

The algorithm iterates uplink and downlink transmissions for T times. In each of the T iterations, it updates $\mathbf{V}^{[k]} = \bar{\mathbf{U}}^{[k]}$ and $\mathbf{U}^{[k]} = \bar{\mathbf{V}}^{[k]}$, which include the uplink and downlink precoding matrices and the uplink and downlink interference suppression matrices. The total interference in the system is monotonically reduced after each iteration, and the convergence of the Distributed IA algorithm can be proved [2].

B. IST-IA

Similarly to [2], the IST-IA algorithm [4] requires a training phase consisting of T iterations, to update the precoding and interference suppression matrices to approach interference alignment. However, IST-IA does not need a priori knowledge of the interference covariance matrix (5). For each uplink or downlink iteration, IST-IA spends KL training symbol periods to estimate the interference suppression matrices and obtain the precoding matrices accordingly.

In the uplink transmission of the first iteration, the $D \times 1$ training symbol vector $\hat{\mathbf{x}}^{[k]}$ is precoded by an arbitrary precoding matrix $\mathbf{V}^{[k]}$, as shown in (3). Then $\mathbf{x}^{[k]} = \mathbf{V}^{[k]} \hat{\mathbf{x}}^{[k]}$ is sent by UE k . These symbols are used by BSs to update the interference suppression matrix $\mathbf{U}^{[k]}$, $k = 1, \dots, K$. To train BS k , the UEs in all other cells transmit L symbols, while the direct transmission from UE k is silenced. From (1), it can be seen that generally the l -th ($l = 1, \dots, L$) symbol vector received by BS k is

$$\mathbf{y}^{[k]}(l) = \sum_{i=1, i \neq k}^K g_{ki} \mathbf{H}^{[ki]}(l) \mathbf{x}^{[i]}(l) + \mathbf{w}^{[k]}(l). \quad (8)$$

The network schedules the transmissions from UE 2, UE 3, ..., UE K to BS 1 firstly to train BS 1; then it schedules the transmissions from UE 1, UE 3, ..., UE K to BS 2 to train BS 2. This procedure continues until each BS receives L training symbol vectors from the UEs in all other cells.

A simple approach to update the interference suppression matrix $\mathbf{U}^{[k]}$ is to estimate the covariance matrix (5) from the L received training vector and then obtain $\mathbf{U}^{[k]}$ via eigenvector decomposition (EVD), which is called as EVD-IA [4]. The IST-IA was proposed in [4] as an alternative solution, which is based on the minor subspace tracking (MST) algorithm [7]. With respect to EVD-IA, IST-IA significantly reduces the computational complexity, with minor performance degradation [4]. Moreover, when the channel is time varying during data transmission after the training phase, IST-IA could be easily deployed to track the interference suppression matrices [4].

Downlink transmissions begin after the BSs have updated their interference suppression matrices $\mathbf{U}^{[k]}$ via MST. Each BS precodes D independent symbols with the precoding matrix

$\bar{\mathbf{V}}^{[k]} = \mathbf{U}^{[k]}$, as shown in (6). Similar to the uplink, each UE receives L training symbol vectors and then compute the interference suppression matrix $\bar{\mathbf{U}}^{[k]}$ via MST. The precoding matrices for the next uplink phase are updated by $\mathbf{V}^{[k]} = \bar{\mathbf{U}}^{[k]}$, as shown in (7). The training period continues for T training iterations.

The IST-IA algorithm [4] is summarized as follows.

The IST-IA algorithm

Training Period:

Initialize $\mathbf{U}^{[k]}, \mathbf{V}^{[k]}$ ($k = 1, \dots, K$): random matrix with orthonormal vectors;

for $t = 1$ to T (T training iterations) **do**

 Uplink: UE \rightarrow BS

for $k = 1$ to K (BS k) **do**

for $l = 1$ to L **do**

for $i = 1$ to $K, i \neq k$ (UE i) **do**

 generate training vector $\mathbf{x}^{[i]}(l) = \mathbf{V}^{[k]} \hat{\mathbf{x}}^{[i]}(l)$

end for

 BS k receives concurrent interference symbol vectors transmitted from the $K - 1$ UEs (i.e. UE $i, i = 1$ to $K, i \neq k$), as shown in (8).

end for

 Find $\mathbf{U}_t^{[k]}(L)$ by the MST Algorithm.

end for

Downlink: BS \rightarrow UE, dual to the uplink part: obtain $\mathbf{V}_t^{[k]}(L)$.

Update $\mathbf{U}^{[k]} = \mathbf{U}_t^{[k]}(L), \mathbf{V}^{[k]} = \mathbf{V}_t^{[k]}(L)$.

end for

DATA Transmission Period.

The efficient MST algorithm utilized in [4] requires the back and forward substitution that is an inherent serial process unsuitable for the parallel implementation [6]. Thus a practical and efficient MST algorithm for IST-IA was deduced [5], which utilizes the parallelizable matrix-vector products instead of the back and forward substitution in [4]. The MST algorithm in [5] is summarized as follows, where

$$\mathbf{W}_x^{[k]}(l) = (\Phi_x^{[k]}(l))^{-1}. \quad (9)$$

- **Initialization** (for the 1st iteration): let $\mathbf{U}^{[k]}(0)$ be a random matrix with orthonormal vectors; $\mathbf{W}_x^{[k]}(0) = (1/\Delta) \cdot \mathbf{I}_N$, where Δ is a small positive real number.

- **Recursion:**

for $l = 1$ to L **do**

Input: $\mathbf{y}^{[k]}(l)$

Compute $\mathbf{W}_x^{[k]}(l)$ from $\mathbf{W}_x^{[k]}(l-1)$ and $\mathbf{y}^{[k]}(l)$ by

$$\underline{\mathbf{y}}_w^{[k]}(l) = \mathbf{W}_x^{[k]}(l-1) \mathbf{y}^{[k]}(l) \quad (10)$$

 and

$$\mathbf{W}_x^{[k]}(l) = \frac{\mathbf{W}_x^{[k]}(l-1)}{\beta} - \frac{\underline{\mathbf{y}}_w^{[k]}(l) \underline{\mathbf{y}}_w^{[k]}(l)^H}{\beta^2 / (1-\beta) + \beta \underline{\mathbf{y}}_w^{[k]}(l)^H \underline{\mathbf{y}}_w^{[k]}(l)}. \quad (11)$$

Compute

$$\mathbf{A}^{[k]}(l) = \mathbf{W}_x^{[k]}(l) \mathbf{U}^{[k]}(l-1). \quad (12)$$

QR decompose $\mathbf{A}^{[k]}(l)$ **into**

$$\mathbf{A}^{[k]}(l) = \mathbf{U}^{[k]}(l) \tilde{\mathbf{R}}^{[k]}(l). \quad (13)$$

end for

- **Output:** $\mathbf{U}^{[k]}(L)$. Moreover, initialize

$\mathbf{U}^{[k]}(0) = \mathbf{U}^{[k]}(L)$ and $\mathbf{W}_x^{[k]}(0) = \mathbf{W}_x^{[k]}(L)$ for the following t^{th} ($2 \leq t \leq T$) iteration.

Notice that the matrix inverse lemma [8, p.30] is utilized to deduce (10) and (11) from

$$\Phi_x^{[k]}(l) = \beta \Phi_x^{[k]}(l-1) + (1-\beta) \mathbf{y}^{[k]}(l) \mathbf{y}^{[k]}(l)^H. \quad (14)$$

Moreover, to some extent, the deduction of (10) and (11) is inspired by that in [9] to obtain the recursive V-BLAST algorithm [5].

IV. EFFICIENT ALGORITHMS WITH MODIFIED QR DECOMPOSITION FOR IST-IA

In what follows, we deduce efficient algorithms for IST-IA.

Let us substitute (9) into (12) to obtain

$$\mathbf{A}^{[k]}(l) = (\Phi_x^{[k]}(l))^{-1} \mathbf{U}^{[k]}(l-1). \quad (15)$$

Then substitute (13) into (15) to obtain

$$(\Phi_x^{[k]}(l))^{-1} \mathbf{U}^{[k]}(l-1) = \mathbf{U}^{[k]}(l) \tilde{\mathbf{R}}^{[k]}(l). \quad (16)$$

By left multiplying (16) by $\Phi_x^{[k]}(l)$, we can obtain

$$\mathbf{U}^{[k]}(l-1) = \Phi_x^{[k]}(l) \mathbf{U}^{[k]}(l) \tilde{\mathbf{R}}^{[k]}(l), \quad (17)$$

which is then right multiplied by $(\tilde{\mathbf{R}}^{[k]}(l))^{-1}$ to obtain

$$\mathbf{U}^{[k]}(l-1) (\tilde{\mathbf{R}}^{[k]}(l))^{-1} = \Phi_x^{[k]}(l) \mathbf{U}^{[k]}(l). \quad (18)$$

Right multiply (18) by $(\mathbf{U}^{[k]}(l))^H$ to obtain

$$\mathbf{U}^{[k]}(l-1) (\tilde{\mathbf{R}}^{[k]}(l))^{-1} (\mathbf{U}^{[k]}(l))^H = \Phi_x^{[k]}(l), \quad (19)$$

where $\mathbf{U}^{[k]}(l) (\mathbf{U}^{[k]}(l))^H = \mathbf{I}$ is utilized. Then similarly we left multiply (19) by $(\mathbf{U}^{[k]}(l-1))^H$ to obtain

$$(\tilde{\mathbf{R}}^{[k]}(l))^{-1} (\mathbf{U}^{[k]}(l))^H = (\mathbf{U}^{[k]}(l-1))^H \Phi_x^{[k]}(l). \quad (20)$$

From (20), we can obtain

$$\mathbf{U}^{[k]}(l) (\tilde{\mathbf{R}}^{[k]}(l))^{-H} = (\Phi_x^{[k]}(l))^H \mathbf{U}^{[k]}(l-1), \quad (21)$$

i.e.,

$$\mathbf{U}^{[k]}(l) \tilde{\mathbf{R}}'^{[k]}(l) = \Phi_x^{[k]}(l) \mathbf{U}^{[k]}(l-1), \quad (22)$$

where $\tilde{\mathbf{R}}'^{[k]}(l) = (\tilde{\mathbf{R}}^{[k]}(l))^{-H}$ is lower triangular since $\tilde{\mathbf{R}}^{[k]}(l)$ is upper triangular. In (22), we need to compute

$$\tilde{\mathbf{A}}^{[k]}(l) = \Phi_x^{[k]}(l) \mathbf{U}^{[k]}(l-1) \quad (23)$$

firstly, and then decompose $\tilde{\mathbf{A}}^{[k]}(l)$ into

$$\tilde{\mathbf{A}}^{[k]}(l) = \mathbf{U}^{[k]}(l) \tilde{\mathbf{R}}'^{[k]}(l). \quad (24)$$

Now instead of (10), (11), (12) and (13), we can utilize (14), (23) and (24), to update $\Phi_x^{[k]}(l)$ by (14), compute $\tilde{\mathbf{A}}^{[k]}(l)$ by (23), and then QR decompose $\tilde{\mathbf{A}}^{[k]}(l)$ into the unitary $\mathbf{U}^{[k]}(l)$ and the lower triangular $\tilde{\mathbf{R}}'^{[k]}(l)$. Notice that the usual QR decomposition decomposes a matrix into a unitary matrix and an upper triangular matrix, while in (24), the modified QR decomposition is required to decompose $\tilde{\mathbf{A}}^{[k]}(l)$ into a unitary matrix and a lower triangular matrix.

Moreover, we can further reduce the computational complexity. Left multiply (22) by $(\mathbf{U}^{[k]}(l))^H$ to obtain

$$(\tilde{\mathbf{R}}^{[k]}(l))^{-H} = (\mathbf{U}^{[k]}(l))^H \Phi_x^{[k]}(l) \mathbf{U}^{[k]}(l-1). \quad (25)$$

Then right multiply (25) by $(\mathbf{U}^{[k]}(l-1))^H$ to obtain

$$(\tilde{\mathbf{R}}^{[k]}(l))^{-H} (\mathbf{U}^{[k]}(l-1))^H = (\mathbf{U}^{[k]}(l))^H \Phi_x^{[k]}(l), \quad (26)$$

i.e.,

$$\mathbf{U}^{[k]}(l-1) (\tilde{\mathbf{R}}^{[k]}(l))^{-1} = \Phi_x^{[k]}(l) \mathbf{U}^{[k]}(l). \quad (27)$$

Now let us substitute (14) into $\Phi_x^{[k]}(l) \mathbf{U}^{[k]}(l-1)$ in (22), to obtain

$$\Phi_x^{[k]}(l) \mathbf{U}^{[k]}(l-1) = [\beta \Phi_x^{[k]}(l-1) + (1-\beta) \mathbf{y}^{[k]}(l) \mathbf{y}^{[k]}(l)^H] \mathbf{U}^{[k]}(l-1) =$$

$$\beta \Phi_x^{[k]}(l-1) \mathbf{U}^{[k]}(l-1) + (1-\beta) \mathbf{y}^{[k]}(l) \mathbf{y}^{[k]}(l)^H \mathbf{U}^{[k]}(l-1),$$

into which we substitute (27) to obtain

$$\Phi_x^{[k]}(l) \mathbf{U}^{[k]}(l-1) = \beta \mathbf{U}^{[k]}(l-2) (\tilde{\mathbf{R}}^{[k]}(l-1))^{-1}, \quad (28)$$

$$+ (1-\beta) \mathbf{y}^{[k]}(l) \mathbf{y}^{[k]}(l)^H \mathbf{U}^{[k]}(l-2)$$

i.e.,

$$\tilde{\mathbf{A}}^{[k]}(l) = \beta \mathbf{U}^{[k]}(l-2) \tilde{\mathbf{R}}'^{[k]}(l-1) + (1-\beta) \mathbf{y}^{[k]}(l) \mathbf{y}^{[k]}(l)^H \mathbf{U}^{[k]}(l-2). \quad (29)$$

It can be seen that instead of (14) and (23), we can use (29) to compute $\tilde{\mathbf{A}}^{[k]}(l)$ directly.

Now we modify the MST algorithm in [5] into the following MST algorithm.

- **Initialization** (for the 1st iteration): let $\mathbf{U}^{[k]}(0)$ be a random matrix with orthonormal vectors; $\Phi_x^{[k]}(0) = \Delta \cdot \mathbf{I}_N$, where Δ is a small positive real number.
- **Recursion:**

for $l = 1$ to L **do**

Input: $\mathbf{y}^{[k]}(l)$

Compute $\Phi_x^{[k]}(l)$ from $\Phi_x^{[k]}(l-1)$ and $\mathbf{y}^{[k]}(l)$ by (14).

Compute $\tilde{\mathbf{A}}^{[k]}(l) = \Phi_x^{[k]}(l)\mathbf{U}^{[k]}(l-1)$, i.e. (23).

Utilize the modified QR decomposition to decompose $\tilde{\mathbf{A}}^{[k]}(l)$ **into**

$\tilde{\mathbf{A}}^{[k]}(l) = \mathbf{U}^{[k]}(l)(\tilde{\mathbf{R}}^{[k]}(l))^{-H}$, i.e. (24).

end for

Output: $\mathbf{U}^{[k]}(L)$. Moreover, initialize

$\mathbf{U}^{[k]}(0) = \mathbf{U}^{[k]}(L)$ and $\Phi_x^{[k]}(0) = \Phi_x^{[k]}(L)$ for

 the following t^{th} ($2 \leq t \leq T$) iteration.

Moreover, instead of (14) and (23), we can use (29) to compute $\tilde{\mathbf{A}}^{[k]}(l)$ directly, as mentioned above.

V. IMPLEMENTATION COMPLEXITY

In this section, we analyze the implementation complexity of the proposed MST algorithm, which is compared with that of the MST algorithm in [5].

Let us assume $N = D$. In MST algorithm in [5], (10) requires about N^2 complex multiplications and additions, (11) requires about $N^2/2$ complex multiplications and additions, and (12) requires about N^3 complex multiplications and additions. As a comparison, about $N^2/2$ and N^3 complex multiplications and additions are required in (14) and (23), respectively. Specifically, only about $N^3/2 + 2N^2$ complex multiplications and additions are required in (29), where it should be noticed that $\tilde{\mathbf{R}}^{[k]}(l-1)$ is always triangular, and $\mathbf{y}^{[k]}(l)^H \mathbf{U}^{[k]}(l-2)$ is computed firstly.

It can be easily seen that with respect to (10), (11) and (12) of the MST algorithm in [5], (14) and (23) of the proposed MST algorithm requires less computational complexity. Moreover, compared to (14) and (23), (29) requires less computational complexity.

For different N , we carried out some numerical experiments to count the average floating-point operations (flops) of the MST algorithm in [5] and the proposed MST algorithms, i.e., the algorithm by (14) and (23) and that by (29). The results are shown in Fig. 1. From Fig. 1, it can also be seen that the complexity of the proposed MST algorithms is less than that of the MST algorithm in [5].

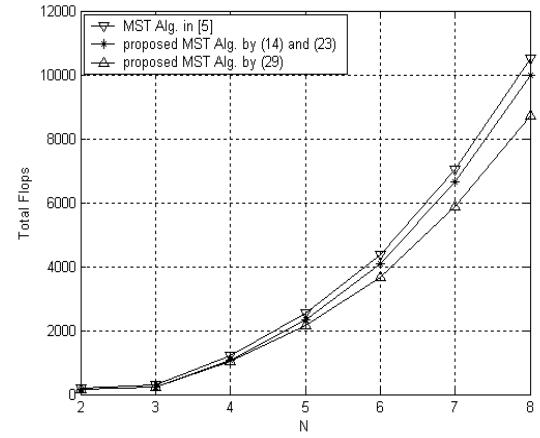


Fig.1 Computational Complexities of the presented MST algorithms.

VI. CONCLUSION

We propose efficient MST algorithms to implement Distributed IA based on CR that employs IST. In the proposed MST algorithms, modified QR decomposition is utilized which decomposes a matrix into a unitary matrix and a lower triangular matrix. With respect to the MST algorithm in [5], the proposed MST algorithms require less computational complexity.

ACKNOWLEDGMENT

This work is supported by Major National S&T Project #2009ZX03003-002 and Major National S&T Project #2010ZX03005-002-01 of China.

REFERENCES

- [1] V.R. Cadambe, S.A. Jafar, "Interference Alignment and Degrees of Freedom of the K-User Interference Channel," IEEE Transactions on Information Theory, vol 54, pp. 3425 – 3441, Aug. 2008.
- [2] K. Gomadam, V.R. Cadambe, S.A. Jafar, "Approaching the Capacity of Wireless Networks through Distributed Interference Alignment," IEEE Globecom 2008.
- [3] S. Srinivasa and S. Jafar, "The throughput potential of cognitive radio - a theoretical perspective," IEEE Communications Magazine, vol. 45, no. 5, 2007.
- [4] Bo Niu, Alexander M. Haimovich, "Interference Subspace Tracking for Network Interference Alignment in Cellular Systems," IEEE Globecom 2009.
- [5] H. Zhu, B. Li and W. Chen, "A Practical and Efficient Algorithm for Distributed Interference Alignment Based on Cognitive Radio," to appear in the 6th International Conference on Wireless Communications , Networking and Mobile Computing (WiCOM) 2010, Sept. 23-25, 2010, Chengdu, China.
- [6] E. J. Baranowski, "Triangular factorization of inverse data covariance matrices", International Conference on Acoustics, Speech, and Signal Processing, 1991 (ICASSP-91), 14-17 Apr 1991, pp. 2245 - 2247, vol.3.
- [7] B. Song, R. Cruz and B. Rao, "Network duality and its application to multi-user mimo wireless networks with sinr constraints," IEEE ICC 2005, vol. 4, pp. 2684–2689, 16-20 May 2005.

- [8] H. Lütkepohl, *Handbook of Matrices*. New York: John Wiley & Sons, 1996.
- [9] H. Zhu, W. Chen, and F. She, “Improved Fast Recursive Algorithms for V-BLAST and G-STBC with Novel Efficient Matrix Inversion,” IEEE ICC 2009.