Software Defined Radio Implementation of the Dual Link Algorithm in TDD Mode using USRP E310

Zhe Feng, Xing Li, Victor Palacios, Peter Mathys, Youjian Liu FIRSTNAME.LASTNAME@COLORADO.EDU
Department of Electrical, Computer, and Energy Engineering, University of Colorado at Boulder

Mingda Zhou, Xinming Huang XHUANG@WPI.EDU
Department of Electrical and Computer Engineering, Worcester Polytechnic Institute

Xin Cai XCAI@STU.XIDIAN.EDU.CN
ISN Lab, Xidian University, Xi'an, China

Abstract

This paper presents challenges and the progress of implementing the Dual Link Algorithm in the fairly new USRP E310 software defined radio (SDR) in time division duplex (TDD) mode. The dual link algorithm that we designed previously aims to solve the interference problem in future dense cellular networks. The algorithm performs joint beam-forming matrix design for transmit signals of multiple transmitters, which are equipped with multiple antennas. The receivers can also be equipped with multiple antennas. The algorithm is the most efficient one we know to find a locally optimal solution to the weighted sum rate maximization problem. Our experiment uses four E310s to model two pairs of interfering users with two antennas at each transmitter and each receiver. In TDD mode, the distributed version of the dual link algorithm is an iterative algorithm with iterations between forward link and reverse link transmissions. The algorithm takes advantage of channel reciprocity to reduce complexity. Therefore, it requires that the same antennas are used for both transmission and reception, as well as a compensation for the differences in transmit and receive RF chains. This paper shares our experience with GNU Radio on using the two TRX ports in each E310 to transmit and then receive signals in network mode, where the signal processing is performed on computers[1].

[1]This work was supported in part by NSF Grants ECCS-1408604 and IIP-1414250.

Proceedings of the 6th GNU Radio Conference, Copyright 2016 by the author(s).

1. Introduction

One of the main approaches to accommodating the explosive growth in mobile data is to reduce the cell size and increase the base station density, while all cells reuse the same frequency spectrum. However, the inter-cell interference becomes severe because the probability of line of sight increases as cell size shrinks. On the other hand, the situation is not hopeless. As promised by interference alignment through joint transmit signal design, every user can have half of the bandwidth at infinite SNR Cadambe & Jafar (2008) no matter how many users are there. Consequently, joint transmit signal design algorithms are expected to be employed to manage interference, or equivalently, maximize data rate at practical SNR, and asymptotically achieve interference alignment.

The weighted sum-rate maximization problem in MIMO interference channels is a classic problem. It can be used for other utility optimization by finding appropriate weights. Various algorithms have been proposed for different scenarios, e.g., Huh et al. (2009; 2010); Jindal et al. (2005); Kim & Giannakis (2011); Liu et al. (2010); Shi et al. (2009; 2011); Yu et al. (2004); Yu (2006; 29 2007-feb. 2); Zhang et al. (2009a;b). Due to the non-convex nature of the problem, convergence property of such algorithms is always a key factor. The majority of the weighted sum-rate maximization algorithms that are convergent in MIMO interference channels take the mean square error (MSE) approach, which transforms the weighted sum-rate maximization problem into an equivalent weighted sum mean square error minimization problem. In Shi et al. (2011), the excellent WMMSE algorithm was proposed for beamforming matrix design for the MIMO interfering broadcast channels and can be applied to the interference channels. With the block coordinate optimization technique, the WMMSE algorithm is guaranteed to converge to a stationary point.

In our previous paper Li et al. (2015), we proposed a faster and convergent algorithm named Dual Link algorithm that solves the weighted sum-rate maximization problem from a different perspective. It exploits the forward-reverse link rate duality in a unique way, equivalently splits the non-convex problem into two sub convex problems and monotonically increases the weighted sum-rate over iterations. The dual link algorithm is highly scalable and suitable for distributed implementation because, for each data link, only its own channel state and the aggregated interference plus noise covariance need to be estimated no matter how many interferers are there. The algorithm works for the MIMO B-MAC networks and assumes Gaussian transmit signal. The MIMO B-MAC network model Liu et al. (2010) includes broadcast channel (BC), multiaccess channel (MAC), interference channels, X networks, and many practical wireless networks as special cases. The dual link algorithm is especially suited for distributed implementation in a time division duplex (TDD) system, where each data link only needs to estimate its own channel and the total received signal covariance matrix, regardless of the number of interfering users. Thus, the algorithm is fit for distributed implementation and is scalable to large systems.

In this paper, we presents the challenges and progress of implementing the distributed dual link algorithm using software defined radio (SDR) USRP E310. The block diagram of the simulation of the algorithm using the GNU Radio Companion and the simulation results are presented. For over-the-air transmission, the main challenge is to implement the TDD mode, where we did not find any precedent to follow. The newly released USRP E310 is ideal for the implementation of the algorithm because it has two TRX antenna ports for transmission and reception. However, the TDD mode using the same antennas is not suitable to be implemented in GNU Radio Companion as of now, because in a TDD mode flow graph, there are both UHD source block, for receiving, and UHD sink block, for transmitting. GNU Radio Companion will keep both running and will have conflict if both blocks are set to use the same antennas. This may be solved by future UHD driver change that enables turning on and off the transmission and reception inside GNU Radio Companion. For now, our solution is to design separate flow graphs for transmission and reception, export the flow graphs' codes, call them one by one in another program. In this paper, we present the flow graphs and the codes of the implementation and share how we set E310 to use the same TRX ports to transmit and then receive. As of now, we have solved the problems encountered in TDD mode, the implementation of the whole system is still in progress.

The rest of this paper is organized as follows. Section 2 presents the system model and the Dual Link algorithm that we want to implement using USRP E310. Section 3 presents the challenge and progress of the implementation. Section 4 shows simulation results. Section 5 concludes with a discussion of future work.

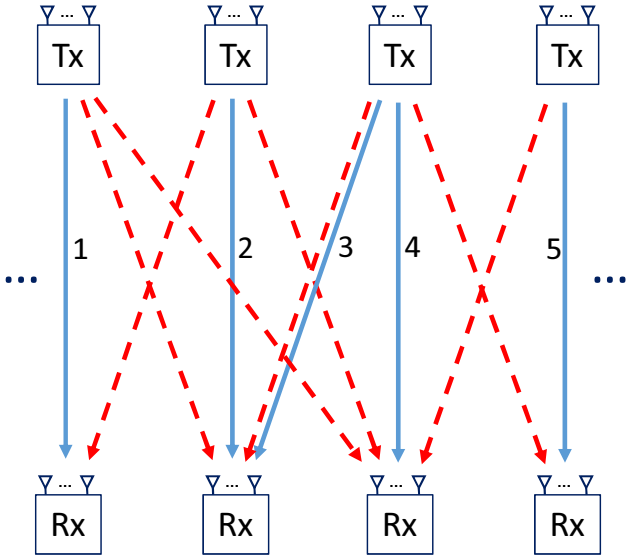

Figure 1. An example of B-MAC network. The solid lines represent data links and the dash lines represent interference.

2. System Model

In this section, we describe the system model and the optimization problem, followed by a brief review the Dual Link algorithm.

2.1. The MIMO B-MAC Network

We consider a general interference network named MIMO B-MAC network with multiple transmitters and receiversLiu et al. (2010; 2011). A transmitter in the MIMO B-MAC network may send independent data to different receivers, like in BC, and a receiver may receive independent data from different transmitters, like in MAC. Each receiver may receive interference from any transmitter in the network. The MIMO B-MAC network model includes broadcast channel (BC), multiaccess channel (MAC), interference channels, X networks, and many practical wireless networks as special cases.

Assume there are totally L mutually interfering data links in a B-MAC network. Link l's transmitter is denoted by T_l, which has L_{T_l} many antennas. Its receiver is denoted by R_l, which has L_{R_l} many antennas. Figure 1 shows an example of B-MAC networks with five data links. Link 2 and 3 have the same physical receiver. Link 3 and 4 have the same physical transmitter. When multiple data links have the same receiver or the same transmitter, interference cancellation techniques such as successive decoding

and cancellation or dirty paper coding can be applied Liu et al. (2010).

The received signal at R_l is

$$\mathbf{y}_l = \sum_{k=1}^{L} \mathbf{H}_{l,k}\mathbf{x}_k + \mathbf{n}_l, \quad (1)$$

where $\mathbf{x}_k \in \mathbb{C}^{L_{T_k}\times 1}$ is the transmit signal of link k and is modeled as a circularly symmetric complex Gaussian vector; $\mathbf{H}_{l,k} \in \mathbb{C}^{L_{R_l}\times L_{T_k}}$ is the channel state information (CSI) matrix between T_k and R_l; and $\mathbf{n}_l \in \mathbb{C}^{L_{R_l}\times 1}$ is a circularly symmetric complex Gaussian noise vector with identity covariance matrix. The circularly symmetric assumption of the transmit signal can be dropped easily by applying the proposed algorithm to real Gaussian signals with twice the dimension. Multiple channel uses can be combined into a larger B-MAC networks with parallel channels, like in interference alignment Cadambe & Jafar (2008).

2.2. The Weighted Sum-rate Maximization Problem

Assuming the channels are known at both the transmitters and receivers (CSITR), an achievable rate of link l is

$$\mathcal{I}_l\left(\mathbf{\Sigma}_{1:L}\right) = \log\left|\mathbf{I} + \mathbf{H}_{l,l}\mathbf{\Sigma}_l\mathbf{H}_{l,l}^{\dagger}\mathbf{\Omega}_l^{-1}\right| \quad (2)$$

where $\mathbf{\Sigma}_l$ is the covariance matrix of $\mathbf{x}_l$; and $\mathbf{\Omega}_l$ is the interference-plus-noise covariance matrix of the l^{th} link,

$$\mathbf{\Omega}_l = \mathbf{I} + \sum_{k=1,k\neq l}^{L} \mathbf{H}_{l,k}\mathbf{\Sigma}_k\mathbf{H}_{l,k}^{\dagger}. \quad (3)$$

If the interference from link k to link l is completely canceled using successive decoding and cancellation or dirty paper coding, we can simply set $\mathbf{H}_{l,k} = \mathbf{0}$ in (3). Otherwise, the interference is treated as noise.

Our goal is to jointly design the signal covariance $\mathbf{\Sigma}_l$'s for all the transmit signals in order to maximize the weighted sum-rate $\sum_{l=1}^{L} w_l\mathcal{I}_l\left(\mathbf{\Sigma}_{1:L}\right)$ of the whole network. The optimization problem that we want to solve is the weighted sum-rate maximization under a total power constraint:

$$\textbf{WSRM\_TP}: \quad \max_{\mathbf{\Sigma}_{1:L}} \quad \sum_{l=1}^{L} w_l\mathcal{I}_l\left(\mathbf{\Sigma}_{1:L}\right) \quad (4)$$
$$\text{s.t.} \quad \mathbf{\Sigma}_l \succeq 0, \forall l,$$
$$\sum_{l=1}^{L} \text{Tr}\left(\mathbf{\Sigma}_l\right) \leq P_{\text{T}},$$

where $w_l > 0$ is the weight for link l.

2.3. The Dual Link Algorithm

The Dual Link Algorithm in Table Algorithm 1 is an iterative algorithm we proposed in Li et al. (2015) to obtain the optimal $\mathbf{\Sigma}_l$'s for the weighted sum-rate maximization problem (4). It has monotonic convergence and is ideally suited for distributed implementation.

Algorithm 1 Dual Link Algorithm

1. Initialize $\mathbf{\Sigma}_l$'s, s.t. $\sum_{l=1}^{L}\text{Tr}\left(\mathbf{\Sigma}_l\right) = P_{\text{T}}$
2. $R \Leftarrow \sum_{l=1}^{L} w_l\mathcal{I}_l\left(\mathbf{\Sigma}_{1:L}\right)$
3. Repeat
4. $R^{'} \Leftarrow R$
5. $\mathbf{\Omega}_l \Leftarrow \mathbf{I} + \sum_{k\neq l}\mathbf{H}_{l,k}\mathbf{\Sigma}_k\mathbf{H}_{l,k}^{\dagger}$
6. $\hat{\mathbf{\Sigma}}_l \Leftarrow \frac{P_{\text{T}}w_l\left(\mathbf{\Omega}_l^{-1} - \left(\mathbf{\Omega}_l + \mathbf{H}_{l,l}\mathbf{\Sigma}_l\mathbf{H}_{l,l}^{\dagger}\right)^{-1}\right)}{\sum_{l=1}^{L} w_l\text{tr}\left(\mathbf{\Omega}_l^{-1} - \left(\mathbf{\Omega}_l + \mathbf{H}_{l,l}\mathbf{\Sigma}_l\mathbf{H}_{l,l}^{\dagger}\right)^{-1}\right)}$
7. $\hat{\mathbf{\Omega}}_l \Leftarrow \mathbf{I} + \sum_{k\neq l}\mathbf{H}_{k,l}^{\dagger}\hat{\mathbf{\Sigma}}_k\mathbf{H}_{k,l}$
8. $\mathbf{\Sigma}_l = \frac{P_{\text{T}}w_l\left(\hat{\mathbf{\Omega}}_l^{-1} - \left(\hat{\mathbf{\Omega}}_l + \mathbf{H}_{l,l}^{\dagger}\hat{\mathbf{\Sigma}}_l\mathbf{H}_{l,l}\right)^{-1}\right)}{\sum_{l=1}^{L} w_l\text{tr}\left(\hat{\mathbf{\Omega}}_l^{-1} - \left(\hat{\mathbf{\Omega}}_l + \mathbf{H}_{l,l}^{\dagger}\hat{\mathbf{\Sigma}}_l\mathbf{H}_{l,l}\right)^{-1}\right)}$
9. $R \Leftarrow \sum_{l=1}^{L} w_l\mathcal{I}_l\left(\mathbf{\Sigma}_{1:L}\right)$
10. until $\left|R - R^{'}\right| \leq \epsilon$ or a fixed number of iterations are reached.

A virtual dual network can be created from the original B-MAC network by reversing the roles of all transmitters and receivers and replacing the channel matrices with their conjugate transpose. The data links in the original networks are denoted as *forward links* while those in the dual network are denoted as *reverse links*. We use $\hat{}$ to denote the corresponding terms in the reverse links. The interference-plus-noise covariance matrix of reverse link l is

$$\hat{\mathbf{\Omega}}_l = \mathbf{I} + \sum_{k=1,k\neq l}^{L} \mathbf{H}_{k,l}^{\dagger}\hat{\mathbf{\Sigma}}_k\mathbf{H}_{k,l}, \quad (5)$$

where $\hat{\mathbf{\Sigma}}_k$ is the transmit signal covariance matrix of reverse link k.

In each iteration, first, the forward link input covariance matrix $\mathbf{\Sigma}_l$ and interference-plus-noise covariance matrix $\mathbf{\Omega}_l$ are used to update the corresponding reverse link input covariance matrix $\hat{\mathbf{\Sigma}}_l$ as shown in Step 6 of Algorithm 1. Then updated$\hat{\mathbf{\Sigma}}_l$ and the corresponding $\hat{\mathbf{\Omega}}_l$ are used to update the forward link input covariance matrix $\mathbf{\Sigma}_l$ as shown in Step 8 of Algorithm 1. As proved in Li et al. (2015), the weighted sum-rate $\sum_{l=1}^{L} w_l\mathcal{I}_l\left(\mathbf{\Sigma}_{1:L}\right)$ will keep increasing over the iterations and converge to a stationary point of the problem (4).

In practical applications, distributed algorithm with low coordination overhead is highly desirable. It turns out that the dual link algorithm is ideally suited for distributed implementation. A centralized implementation of the dual

link incurs overhead. The algorithm needs global channel state information $\mathbf{H}_{k,l},\ \forall k,l$, as do other algorithms. The collection of global channel state information wastes bandwidth and incur large delays for large networks. If the delay is too long, there won't be enough time left for actual data transmission before the channels change. Fortunately, a distributed implementation of the dual link algorithm needs minimal local channel state information compared to other algorithms, especially in TDD networks. A node only needs to estimate total received signal covariance and desired link channel state information. No channel state information of interfering links is needed. And the nodes do not need to exchange channel state information because non-local CSI is not needed. In a TDD network, assuming channel reciprocity, the virtual reverse link iteration, Step 8 of Algorithm 1, can be carried out in physical reverse links. Assume reverse link l transmits pilot signals using beamforming matrix $\hat{\mathbf{V}}_l$, where $\hat{\mathbf{V}}_l\hat{\mathbf{V}}_l^\dagger = \hat{\mathbf{\Sigma}}_l$. To calculate $\mathbf{\Sigma}_l$, we need $\hat{\mathbf{\Omega}}_l$ and reverse link total received signal covariance $\hat{\mathbf{\Omega}}_l + \mathbf{H}_{l,l}^\dagger\hat{\mathbf{\Sigma}}_l\mathbf{H}_{l,l}$, which can be estimated at node T_l because the channel has done the summation of signal, interference, and noise for us for free, no matter how large the network is. Using the pilot signal, $\mathbf{H}_{l,l}^\dagger\hat{\mathbf{V}}_l$ can be estimated and $\mathbf{H}_{l,l}^\dagger\hat{\mathbf{\Sigma}}_l\mathbf{H}_{l,l} = \mathbf{H}_{l,l}^\dagger\hat{\mathbf{V}}_l\hat{\mathbf{V}}_l^\dagger\mathbf{H}_{l,l}$ can be calculated and subtracted from $\hat{\mathbf{\Omega}}_l + \mathbf{H}_{l,l}^\dagger\hat{\mathbf{\Sigma}}_l\mathbf{H}_{l,l}$ to obtain $\hat{\mathbf{\Omega}}_l$. All reverse links only need to share a scalar variable to adjust the total power. The forward link calculation can be done similarly. By avoiding global CSI collection, the distributed dual link algorithm has significant lower signaling overhead compared to other methods.

Because we rely on the reciprocity of the TDD channels to carry out the virtual dual network iterations in the physical reverse channels, the same antennas have to be used for both transmission and reception. This is one of the main challenges in implementing the Dual Link algorithm in the E310 since we couldn't find any prior SDR GNU Radio solution on using the same antennas in TDD mode.

3. Software Defined Radio Implementation

To test the dual link algorithm's performance in over-the-air channels, we implement the algorithm using software defined radios. There are several challenges in the implementation. First, the algorithm assumes reciprocity in the physical forward and reverse links. Consequently, the SDR devices must transmit and receive through the same antennas and the difference in the transmit and receive RF chains of each transceiver must be compensated. Second, the algorithm is iterative, calling for the SDR devices to switch between the transmit mode and the receive mode rapidly. To support multiple antennas, the SDR devices also need to be equipped with multiple antennas which are able to transmit and receive. The newly released USRP E310 is ideal for the implementation of the algorithm because it has two TRX antenna ports for transmission and reception. The radio frequency network on chip (RFNoC) RFNoc capability of E310 also has the potential to speed up the communication by conveniently utilizing the field-programmable gate array (FPGA) circuits inside the E310s. As the name suggests, RFNoC's primary use is to create applications as low-level embedded algorithms, which are faster than their non-embedded counterparts due to their sheer computational power. RFNoC also allows for modular FPGA development, in the same manner that GNU Radio flowcharts use interconnected blocks, which will enable the embedded implementation of the Dual Link Algorithm to be easily tailored to a plethora of software-defined radio (SDR) applications since it will be composed of individual customizable blocks rather than a single algorithm.

We plan to implement the Dual Link Algorithm in three steps: 1) The algorithm is prototyped in GNU Radio Companion with simulated channel; 2) To test the synchronization and channel estimation components, the case of a single pair of users will be implemented in network mode of E310s using over-the-air channel; 3) The case of two pair of interfering users will be implemented using four E310s in network mode. We are currently at step 2), in the process of smoothing the transmit and receive mode switching using the TRX ports of the E310s.

3.1. GNU Radio Simulation

In the first step, we built an Out-of-Tree (OOT) module Feng and designed the flow graph in Figure 2 in GNU Radio Companion. In this system, there are two pairs of users equipped with two antennas each. The upper stream flow is the forward link while the lower one is the reverse link. A centralized processor calculates the weighted sum rate after a full forward and reverse loop.

One problem is that GNU Radio Companion does not support loop in simulation. Our solution is to pass the data by using UDP Source and Sink. In the forward link, at the transmitter side, The "UDP source" block outputs a vector of length 8 including a pair of 2 by 2 covariance matrices Σ_1 and Σ_2. The vector is divided into two vectors which contain one covariance matrix each. The two "pilot_gen" blocks actually act as transmitters of the users "1-1" and "1-2". Each "pilot_gen" block generates 2 orthogonal pilot sequences, which is multiplied by a precoding matrix to make the transmit signal have covariance matrix Σ_1 and Σ_2. The "channel" block models a B-MAC network with a pair of direct channels and a pair of interference channels. The "Interference gain" parameter determines the ratio of trace of interference channels to that of direct channels. At the receiver side, the two

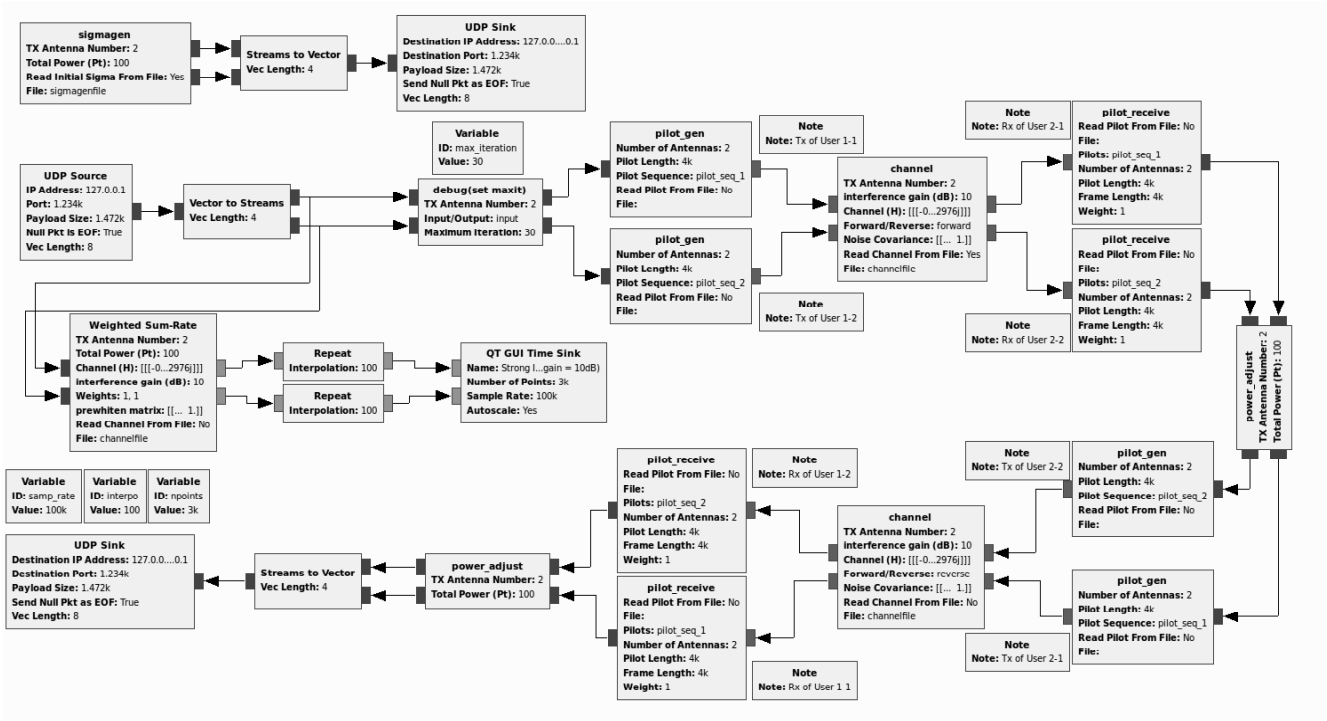

Figure 2. GNU Radio Simulation of two pair of users

"pilot_receive" blocks actually act as the receivers of the users "2-1" and "2-2". Each "pilot_receive" block first estimates the channel matrix by correlating the received data with known pilot sequences. Then it calculates the initial $\hat{\Sigma}_l$ matrix $\mathbf{\Omega}_l^{-1} - \left(\mathbf{\Omega}_l + \mathbf{H}_{l,l}\mathbf{\Sigma}_l\mathbf{H}_{l,l}^{\dagger}\right)^{-1}$ from the received samples. The "power adjust" block takes in all the initial $\hat{\mathbf{\Sigma}}$ matrices and reallocates the power such that the sum of their traces equals to the total power constraint. The "power adjust" block outputs the adjusted $\hat{\mathbf{\Sigma}}$ as the input of the "pilot gen" at the reverse link. It is noted that user "2-1" and "2-2" are changed from receive to transmit mode while user "1-1" and "1-2" are switched to receive from transmit mode. Similarly, the Σ is calculated from the reverse link and passed to the "UDP Sink" which is connected to to "UDP Source" to create a loop. Two blocks are connected to the "UDP Sink", the "UDP Source" and "power_adjust" of the reverse link. Because the "sigmagen" block only generate the initial Σ_1 and Σ_2 before the first iteration and then stays idle. The "power_adjust" block starts to transmit data since the end of the first iteration. This time is perfectly aligned without conflict. Thus we creates a virtual loop in the simulation which allows multiple iterations. A "debug" block is created to control the number of iterations by counting the number of times data pass through the block. Furthermore, the "Weighted Sum Rate" block calculates the weighed sum rate R_l after the l-th iteration. The rates are displayed through "Time Sink" to indicate convergence.

This design distinguishes it from most systems with several features. Firstly, most of the blocks in this OOT module are written in Python to better deal with the matrix operations resulted from multi-user MIMO communication. Additionally, because most of the data streaming between blocks are matrices, we reshape the matrices into long vectors and pass them. Moreover, iterations are enabled by the virtual loop and controlled by the iterative counter. This method might inspire the implementations of some other iterative algorithms.

3.2. USRP Implementations

To test the algorithm in over-the-air channels, there are several more issues to consider. For example, if the system is moved to RF with the channel effects like time delay and frequency offset, synchronization and phase compensation algorithm should be implemented. Furthermore, when we want to make iterative transmission and reception, we have to write a master program to call corresponding transmitter and receiver accordingly, because inside the GNU Radio Companion, a UHD source block and a UHD sink block will be kept on all the time and thus, they cannot use the same antennas. Some problems are not fully solved yet. We would like to share our progress and welcome any suggestions.

We start from a simple case that contains only one way communication between a pair of E310 with two antennas each to illustrate the design of synchronization and frequency offset. Both E310s are used in network mode. As in the previous example, at the transmitter side, the "pilot_gen" block generates two orthogonal pilot sequences. Each sequence is pulse shaped by a interpolating root raised cosine filter (RRC) to reduce inter-symbol interference (ISI). Then both RRC filters are connected to a USRP Sink with two input channels to modulate the base band signals to RF.

At the receiver side, a "USRP Source" block is used to demodulate the RF signal back to base band. Its two output ports are connected to two decimating RRC filters. Then each output is connected to a "phase_corrector" hierarchical block. Inside the hierarchical block, the decimated samples are first multiplied by the conjugate of the pilot sequence, and then the results are connected to a phase lock loop (PLL) to generate a clean reference carrier. The conjugate of the reference carrier is then multiplied to the decimated samples to recover their phase. Then we consider the frame synchronization. Since the two antennas transmit orthogonal pilot sequences, their synchronizations can be done separately. For each antenna link, the phase corrected output is connected to a filter matched to the corresponding pilot sequence. The magnitude of the output is expected to have a peak, which is located at the end of the pilot sequence. So a peak detector is used to detect the peak. When a peak is detected, the peak detector outputs a flag '1'. In order to use the flags to synchronize the received data, we designed a "rx_frame_sync" block with two input ports "in_data" and "in_flag". When the "in_flag" inputs 0, this block does nothing. When the "in_flag" inputs 1, this block starts to output the "pilot_length" number of samples prior to this flag. With this design, we can "chop" the desired frame at the appropriate location and pass the frame for further processing.

As mentioned before, it is impossible to implement TDD mode using the same antennas in GNU Radio Companion. But it is possible in Python. For the two-user case, the two users are denoted as A and B. We built the transmitter and receiver flow graphs of A and B into python objects. The transmitter object of A and receiver object of A share the same address and antennas. The same is true for B. By deploying the start(), stop(), wait() methods of gr.top_block, we are able to build an iterative system by calling different objects accordingly. For example, a simple piece of python codes which does the iterative transmission is shown below. The tb_at and tb_ar denote the transmitter and receiver objects of user A while tb_bt and tb_br denote the transmitter objects of user B. The data from a "File Source" is first transmitted from user A and received by user B. Then user B stores the data in a file and transmits it back to A. Then user A stores the data in a file and back and forth. The program starts forward transmission by calling the start() method of tb_br and tb_at. The tb_at and tb_br can be called to stop if we hit the "Enter" button. To start the reverse link, we hit the "Enter" button again to call the start() method of tb_bt and tb_ar. This is how the iterative transmission can be implemented.

```
qa = Qt.QApplication(sys.argv)
#define quitting:
def qt_tb_at():
    tb_at.wait()
    tb_at.stop()
def qt_tb_ar():
    tb_ar.wait()
    tb_ar.stop()
def qt_tb_br():
    tb_br.wait()
    tb_br.stop()
def qt_tb_bt():
    tb_bt.wait()
    tb_bt.stop()
#activate the rx of B and tx of A
tb_br.start()
tb_br.show()
tb_at.start()
tb_at.show()
try:
raw_input('Press Enter to stop forward link: ')
except EOFError:
 pass
#stop the tx of A and rx of B
qa.connect(qa,Qt.SIGNAL("aboutToQuit()"),qt_tb_at)
qa.connect(qa,Qt.SIGNAL("aboutToQuit()"),qt_tb_br)
try:
raw_input('Press Enter to start reverse link: ')
except EOFError:
pass
#activate the rx of A and tx of B
tb_bt.start()
tb_ar.start()
tb_ar.show()
tb_bt.show()
```

This simple example code illustrates how to make iterative transmission and receptions between a pair of users. We found that the "File Source" was not working after the second iteration. It is because the read pointer stayed at the end of the file after the first call and was not moved to the start of the file at the second call. We fixed this issue by calling the file_source.open() method every time we finished a call. There are some issues with this simple example which we are still working on to resolve. One of the main issue is how to stop more elegantly without hitting the "Enter" button to make it run automatically. If we can resolve all

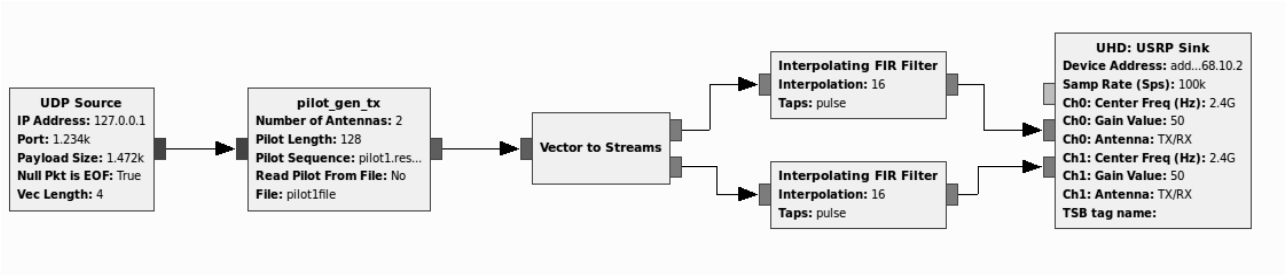

Figure 3. Tx of the E310 network mode

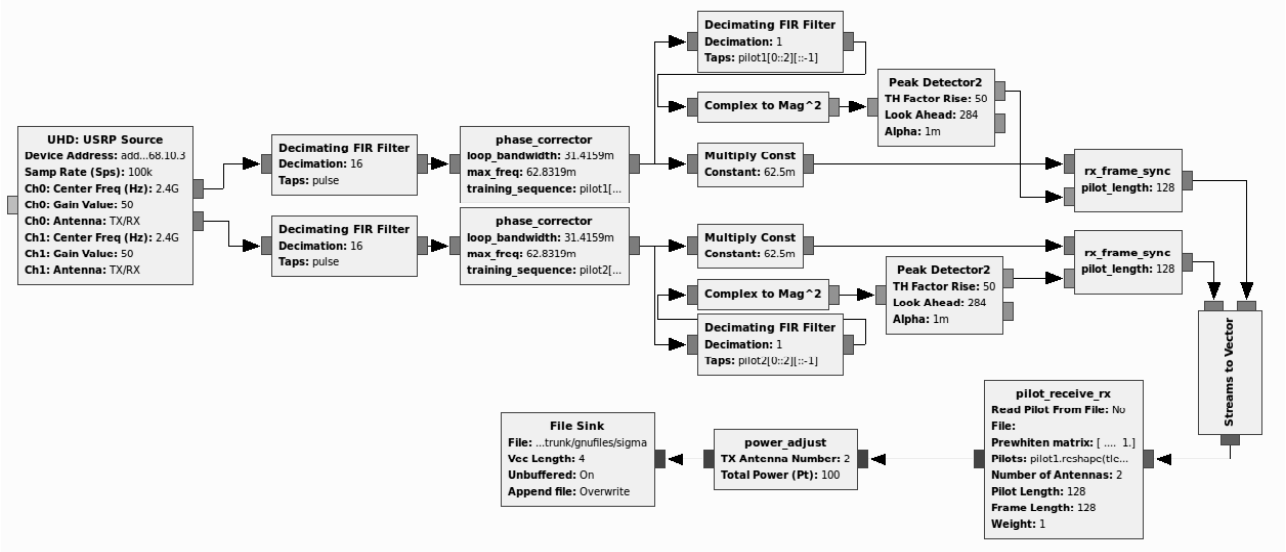

Figure 4. Rx of the E310 network mode

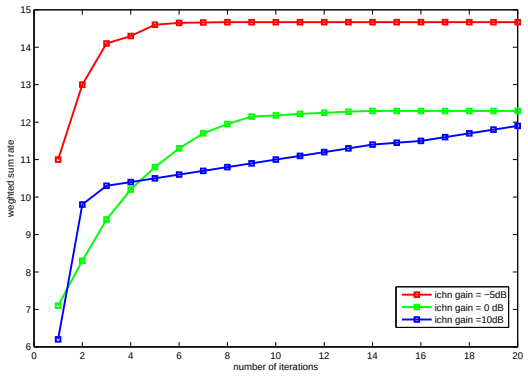

Figure 5. Weighted Sum Rate vs number of iterations

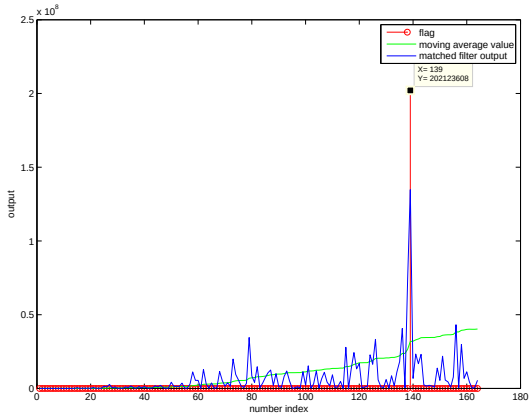

Figure 6. synchronization

the issues and replace the simple transmitter and receiver with the dual link algorithm, we should be able to test it on USRP. Additionally, in this design, the E310s can transmit and receive at the same antennas, so the channel reciprocity is preserved after implementation of a RF chain calibration.

4. Result and Discussion

4.1. GNU Radio Simulation

An experiment is conducted with the GNU Radio System in Figure 2. The SNR is set to 20 dB and each user is equipped with 2 antennas. The "channel" block is set with interference gain equals to -5dB, 0 dB and 10dB which correspond to weak, moderate and strong interference. The number of iterations is set to 20. The weights of all users are set to 1. The weighted sum rate vs number of iterations figure is generated and plotted in Figure 5. It can be seen the algorithm converges quickly to a high rate for weak interference and converges slowly to a lower rate for strong interference as expected.

4.2. E310 Synchronization

For the one way transmission system, we conducted an experiment to show the synchronization capability. In this experiment, we use one E310 as transmitter with the flow graph in Figure 3 and another E310 as receiver with the flow graph in Figure 4. At the transmitter, the pilot sequences are orthogonal with length 128. The shaping pulse is a standard RRC filter of length 16. The TRX-A and TRX-B antennas of the E310 are used to transmit at center frequency 2.4GHz. At the receiver, the TRX-A and TRX-B antennas of another E310 are used to receive at center frequency 2.4GHz. We used "File Sinks" to store the power of the output signal of the matched filter, the output flags and the moving average value of the "peak detector". The three streams are plotted in Figure 6.

It can be seen from Figure 6 that the magnitude of the matched filter output has a peak, which is the end of the pilot sequence frame. The peak detector is able to capture this peak which leads to a correct frame synchronization. The output $\hat{\Sigma}$ also looks reasonable after further processing. So this synchronization results make it possible to the transmit and receive via E310.

5. Conclusion and Future Work

We have presented challenges and progress in the implementation of the dual link algorithm to solve the weighted sum-rate maximization problem. We showed how to model and simulate the algorithm in GNU Radio Companion. We also provided a method to implement the TDD communication mode using E310s. The remaining work includes fully automated TDD communication mode implementation, RF chain calibration, and synchronization algorithm implementation.

References

Cadambe, V.R. and Jafar, S.A. Interference Alignment and Degrees of Freedom of the K-User Interference Channel. *IEEE Transactions on Information Theory*, 54(8):3425–3441, August 2008. ISSN 0018-9448. doi: 10.1109/TIT.2008.926344. 1, 2.1

Feng, Zhe. OOT Module. URL `https://github.com/fengzhe29888/gr-PWF`. 3.1

Huh, H., Papadopoulos, H., and Caire, G. MIMO broadcast channel optimization under general linear constraints. In *Proc. IEEE Int. Symp. on Info. Theory (ISIT)*, 2009. 1

Huh, H., Papadopoulos, H. C., and Caire, G. Multiuser MISO transmitter optimization for intercell interference mitigation. *IEEE Transactions on Signal Processing*, 58

(8):4272 –4285, August 2010. ISSN 1053-587X. doi: 10.1109/TSP.2010.2048564. 1

Jindal, N., Rhee, W., Vishwanath, S., Jafar, S. A., and Goldsmith, A. Sum power iterative water-filling for multi-antenna Gaussian broadcast channels. *IEEE Trans. Info. Theory*, 51(4):1570–1580, April 2005. 1

Kim, Seung-Jun and Giannakis, G.B. Optimal resource allocation for MIMO ad hoc cognitive radio networks. *IEEE Trans. Info. Theory*, 57(5):3117 – 3131, May 2011. ISSN 0018-9448. doi: 10.1109/TIT.2011.2120270. 1

Li, Xing, You, Seungil, Chen, Lijun, Liu, An, and Liu, Y.E. A new algorithm for the weighted sum rate maximization in MIMO interference networks. In *IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 147–152, March 2015. doi: 10.1109/WCNC.2015.7127460. 1, 2.3, 2.3

Liu, An, Liu, Youjian, Xiang, Haige, and Luo, Wu. Duality, polite water-filling, and optimization for MIMO B-MAC interference networks and iTree networks. *IEEE Trans. Info. Theory, in revision, submitted*, Apr. 2010. 1, 2.1

Liu, An, Liu, Youjian, Xiang, Haige, and Luo, Wu. MIMO B-MAC interference network optimization under rate constraints by polite water-filling and duality. *IEEE Trans. Signal Processing*, 59(1):263 –276, January 2011. ISSN 1053-587X. doi: 10.1109/TSP.2010.2088394. 2.1

RFNoc. RFNoC: Getting Started. URL https://github.com/EttusResearch/uhd/wiki/RFNoC:-Getting-Started. 3

Shi, C., Berry, R. A., and Honig, M. L. Monotonic convergence of distributed interference pricing in wireless networks. *in Proc. IEEE ISIT, Seoul, Korea*, June 2009. 1

Shi, Qingjiang, Razaviyayn, M., Luo, Zhi-Quan, and He, Chen. An iteratively weighted mmse approach to distributed sum-utility maximization for a mimo interfering broadcast channel. *IEEE Trans. Signal Processing*, 59 (9):4331 –4340, September 2011. ISSN 1053-587X. 1

Yu, W., Rhee, W., Boyd, S., and Cioffi, J.M. Iterative water-filling for Gaussian vector multiple-access channels. *IEEE Trans. Info. Theory*, 50(1):145–152, 2004. 1

Yu, Wei. Sum-capacity computation for the gaussian vector broadcast channel via dual decomposition. *IEEE Trans. Inform. Theory*, 52(2):754–759, February 2006. ISSN 0018-9448. doi: 10.1109/TIT.2005.862106. 1

Yu, Wei. Multiuser water-filling in the presence of crosstalk. *Information Theory and Applications Workshop, San Diego, CA, U.S.A*, pp. 414 –420, 29 2007-feb. 2. doi: 10.1109/ITA.2007.4357612. 1

Zhang, Lan, Liang, Ying-Chang, Xin, Yan, Zhang, Rui, and Poor, H.V. On gaussian MIMO BC-MAC duality with multiple transmit covariance constraints. In *Proc. IEEE Int. Symp. on Info. Theory (ISIT)*, pp. 2502 –2506, June 2009a. 1

Zhang, Lan, Xin, Yan, and Liang, Ying-chang. Weighted sum rate optimization for cognitive radio MIMO broadcast channels. *IEEE Trans. Wireless Commun.*, 8(6): 2950 –2959, June 2009b. ISSN 1536-1276. doi: 10.1109/TWC.2009.080621. 1