Robust Wirtinger Flow Algorithm for Channel Coded Blind Demixing

Amin Jalali
ECE Department
Univeristy of California, Davis
Davis, California
amjal@ucdavis.edu

Yuanming Shi
School of Inf. Science and Tech.
ShanghaiTech University
Shanghai, China
shiym@shanghaitech.edu.cn

Zhi Ding
ECE Department
Univeristy of California, Davis
Davis, California
zding@ucdavis.edu

***Abstract*—As applications of Internet-of-things (IoT) rapidly expand, unscheduled multiple user access with low latency and low cost communication is attracting growing more interests. To recover the multiple uplink signals without strict access control under dynamic co-channel interference environment, the problem of blind demixing emerges as an important obstacle for us to overcome. Without channel state information, successful blind demixing can recover multiple user signals more effectively by leveraging prior information on signal characteristics such as constellations and distribution. This work studies how forward error correction (FEC) codes in Galois Field can generate more effective blind demixing algorithms. We propose a constrained *Wirtinger flow algorithm* by defining a *valid signal set* based on FEC codewords. Specifically, targeting the popular polar codes for FEC of short IoT packets, we introduce signal projections within iterations of Wirtinger Flow based on FEC code information. Simulation results demonstrate stronger robustness of the proposed algorithm against noise and practical obstacles and also faster convergence rate compared to regular Wirtinger flow algorithm.**

***Index Terms*—Blind demixing, forward error correction (FEC) constraints, IoT, signal recovery, Wirtinger flow.**

I. Introduction

Blind demixing, as a general form of blind deconvolution, is a fundamental problem that arises in different fields such as wireless receptions, speech processing, image processing, and geophysical signal processing, among others. In blind deconvolution, we observe the convolution of two unknown sequences among noise and would like to recover one or both sequences based on their statistical or other characteristics. More generally, blind demixing considers a sequence $\mathbf{y}$ which is the noisy superposition of multiple unknown convolved sequences $\{\mathbf{s}_i\}$ and $\{\mathbf{w}_i\}$ such that $\mathbf{y} = \sum_{i=1}^{S} \mathbf{s}_i * \mathbf{w}_i + \mathbf{n}$, where $\{\mathbf{n}\}$ is the observation noise. Without further knowledge on the characteristic properties of $\{\mathbf{s}_i\}_1^S$ and $\{\mathbf{w}_i\}_1^S$, it is impossible to untangle the mixed signals. However, under certain reasonable and practical assumptions [1], [2], signal recovery becomes possible. The goal of blind demixing is to recover $\{\mathbf{s}_i\}_1^S$ and $\{\mathbf{w}_i\}_1^S$ based on their various characteristics known a priori.

Blind demixing can be a practical signal recovery solution to support unscheduled channel access by reducing the scheduling and channel estimation overheads. In particular, the massive number of low energy IoT devices in deployment makes unscheduled device access a very attractive low overhead protocol. To accommodate more spontaneous IoT transmissions without centralized scheduling, successful signal reception among co-channel interferences is essential. Blind demixing presents an effective, low cost, and simpler receiver solution by minimizing delays and power/bandwidth costs associated with the transmission of reception of scheduling information and pilots for channel state information.

There exist a number of different approaches to solve the blind deconvolution and the blind demixing problem. One approach is to transform the original bilinear problem into a convex optimization problem by lifting the unknown sequences into unknown rank-one matrices. Through convex relaxations, the transformed problems can be solved via semidefinite programming and nuclear norm minimization [2], [3]. Although such approach shows attractive statistical guarantee in terms of convergence, the much larger solution search space due to lifting does not scale to large problem size involving massive user deployment.

To avoid the scalability issue caused by convex lifting, it is more advantageous to remain within the lower dimensional parameter space. Without relying on convex optimization, recent development has considered non-convex problems by exploiting the manifold geometry of fixed-rank matrices [4] via Riemannian optimization. However, the complex iterative strategies in the Riemannian optimizarion algorithms raise challenges to statistical analysis. On the other hand, the Wirtinger flow (WF) algorithm originally proposed in [5] to solve the phase retrieval is simpler to implement and represents a good candidate for high dimensional statistical problems. As shown in [4], the WF algorithm provides statistically optimal solution to the blind demixing problem.

Generally, blind demixing can benefit from advanced knowledge and characteristics of the underlying unknown sequences. Particularly, in most modern communication systems, FEC codes play a key role to overcome errors cased by noises, interferences, and other practical obstacles in almost all signal transmissions. Existing blind demixing studies have not considered their integration into the problem formulation, primarily because FEC codes are often defined in the finite Galois

This material is based upon work supported by the National Science Foundation under Grant No. 2009001.

field that is incompatible with the blind demixing models typically defined over complex field. We have successfully developed joint receivers that can effectively incorporate the FEC code information as polytope signal constraints in the complex signal field to achieve effective and robust wireless signal reception [6], [7]. In this work, we propose effective algorithms of blind demixing by integrating signal characteristics that are evoked by the underlying FEC codes in wireless transmissions. We focus specifically on the use of polar codes that are practically suitable as FEC codes for short IoT transmissions.

Organization: present the notations of the blind demixing system model in Section II. We formulate the exact codeword constrained blind demixing problem as a non-convex optimization problem. Next, we review the Wirtinger flow algorithm for blind demixing. In Section III, we present the relaxed complex field signal constraints induced by the polar codewords and incorporate the relaxed codeword constraints into a joint blind demixing Wirtinger flow algorithm. We present the simulation results in Section —V before concluding in Section V.

II. System Model

Let $\{.\}^*$, $\{.\}^T$, and $\{.\}^H$ denote conjugate, transpose, and conjugate transpose, respectively. We show scalars with either small or large cap but non-bold letters such as v or V, vectors with bold small cap such as $\mathbf{v}$, matrices with bold large cap such as $\mathbf{V}$ and finally sets with $\mathcal{V}$.

A. Signal Model

Consider an IoT network containing one base or access station and S distributed devices. The received signal vector is the mixture of actively transmitted signals from as many as S devices. Each signal must pass through its own channel modeled by a linear time invariant causal system. Neither the source signal nor the user channel response is known. The objective of the access station is to simultaneously decode user data for each source node. A potential by-product of this blind demixing may also be the identification of the user channel.

Let $\mathbf{x}_i = [x_{i,1} \cdots, x_{i,K}]^T$ be the QAM complex signal vector transmitted by the i-th source node, where $1 \leq i \leq S$. Each signal vector $\mathbf{x}_i$ is multiplied by a known linear precoding matrix $\mathbf{A}_i \in \mathbb{C}^{N\times K}$ consisting of zero mean Gaussian i.i.d. random variables with variance 1. Without loss of generality, we focus on OFDM transmissions by defining $N \times N$ matrix $\mathbf{F}$ and $\mathbf{F}^H$ be the N-point FFT and IFFT matrix respectively where $\mathbf{F}(a,b) = N^{-1/2}e^{-j2\pi(a-1)(b-1)/N}, 1 \leq a, b \leq N$ and $\mathbf{F}\mathbf{F}^H = \mathbf{I}$.

After linear precoding $\mathbf{A}_i\mathbf{x}_i$, we apply IFFT to generate signal vector in the time domain, $\tilde{\mathbf{x}}_i = \mathbf{F}^H\mathbf{A}_i\mathbf{x}_i$. The time domain signal vector $\tilde{\mathbf{x}}_i$ will further append a cyclic prefix before being transmitted to the baseband channel. At the receiver, by removing the cyclic prefix from the received signal, the relationship between input and output signal vectors can be written in cicular convolution form as

$$\tilde{\mathbf{y}} = \sum_{i=1}^{S} \tilde{\mathbf{x}}_i \circledast \mathbf{w}_i + \tilde{\mathbf{n}}, \tag{1}$$

where $\tilde{\mathbf{n}}$ is complex AWGN channel noise vector and $\mathbf{w}_i \in \mathbb{C}^N$ is the unknown channel impulse response vector from the i-th source to the receiver with the maximum delay spread L. Assuming a slotted random access scheme, we can write, without loss of generality

$$\mathbf{w}_i = \mathbf{D}\mathbf{h}_i, \quad \text{where } \mathbf{h}_i \in \mathbb{C}^L \quad \mathbf{D} = \begin{bmatrix} \mathbf{I}_L \\ \mathbf{0} \end{bmatrix} \tag{2}$$

Taking FFT of the received signal $\tilde{\mathbf{y}}$ at the receiver generates the equivalent frequency domain relationship between input and output as

$$\begin{aligned} \mathbf{y} = \mathbf{F}\tilde{\mathbf{y}} &= \sum_{i=1}^{S} \mathbf{F}\tilde{\mathbf{x}}_i \odot \mathbf{F}\mathbf{w}_i + \mathbf{F}\tilde{\mathbf{n}} \\ &= \sum_{i=1}^{S} \mathbf{F}\mathbf{F}^H\mathbf{A}_i\mathbf{x}_i \odot \mathbf{F}\mathbf{D}\mathbf{h}_i + \mathbf{F}\tilde{\mathbf{n}} \\ &= \sum_{i=1}^{S} \mathbf{A}_i\mathbf{x}_i \odot \mathbf{C}\mathbf{h}_i + \mathbf{n}, \end{aligned} \tag{3}$$

where $\odot$ denotes element-wise product. In addition, we also write $\mathbf{C} = \mathbf{F}\mathbf{D}$ and $\mathbf{n} = \mathbf{F}\tilde{\mathbf{n}}$. Let $\mathbf{c}_j^T$ and $\mathbf{a}_{ij}^T$ be the j-th row of $\mathbf{C}$ and $\mathbf{A}_i$, respectively. As a result, y_j as the $j-$th element of $\mathbf{y}$ becomes

$$\begin{aligned} y_j &= \sum_{i=1}^{S} \mathbf{a}_{ij}^T\ \mathbf{x}_i{\mathbf{c}_j}^T\mathbf{h}_i + n_j, \quad 1 \leq j \leq N \\ &= \sum_{i=1}^{S} \mathbf{x}_i^T\mathbf{a}_{ij}{\mathbf{c}_j}^T\mathbf{h}_i + n_j \\ &= \sum_{i=1}^{S} {\mathbf{c}_j}^T\mathbf{h}_i\mathbf{x}_i^T\mathbf{a}_{ij} + n_j, \end{aligned} \tag{4}$$

where n_j is j-th element of white Gaussian noise vector $\mathbf{n}$.

B. Forward Error Correction (FEC) Codes

Note that source signal sequence $\mathbf{x}_i$ originates from the data bits of the $i-$th source. In most communication signal transmissions, data bits are first encoded into FEC codewords before being mapped into QAM symbols of constellation $\mathcal{Q}$. Without loss of generality, we focus our discussion on polar codes.

Briefly, a polar code of rate $r = k/n$ is specified by $(n, k, \mathcal{I}^c)$, where $n = 2^m$ is the codeword length and k is the number of information bits in a codeword. Let $\mathcal{I} \subseteq \{1, \ldots, n\}$ denote the set of indices of the information bits whose compliment set $\mathcal{I}^c$ denotes the set of frozen (non-information bearing) bits. Let $\mathbf{u} = [u_1, u_2, \cdots, u_n]$ denote the binary information vector and let $\mathbf{b} = [b_1, b_2, \cdots, b_n]$ be the binary codeword vector. There is a 1-1 mapping $\mathbf{b} = \mathbf{u}\mathbf{G}_\mathbf{n}$ between $\mathbf{u}$ and $\mathbf{b}$ in which $\mathbf{G}_\mathbf{n}$ is the generator matrix of the polar code. Note that $\mathbf{G}_\mathbf{n}$ is defined through $\mathbf{G}_\mathbf{n} = \mathbf{B}_\mathbf{n}\mathbf{R}^{\otimes n}$, where $\mathbf{B}_\mathbf{n}$ is a bit reversal operator defined in [8], and $\mathbf{R}^{\otimes n}$ denotes $n-$fold Kronecker power of polarization kernel $\mathbf{R} = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$. Using the concept of channel polarization, n identical realization of

the channel can be transformed into n parallel virtual bit-channels, which become polarized asymptotically to either extremely noisy or error-free as n tends to infinity. Consequently, the crucial step in constructing polar codes is to sort the virtual bit-channels based on their capacity and to select the k most reliable ones out of n bit-channels for carrying the k information bits in each codeword of n bits. The remaining $n-k$ bit-channels in the $n-$bit codeword will contain frozen bits that are set to known values without bearing information.

C. Code Constrained Blind Demixing

Consider a linear block code of rate $r = k/n$ whose 2^k codewords form a set $\mathcal{C}$ in which each codeword contains k information bits. Let the ground-truth codeword vector of the i-th source be $\mathbf{b}_i^T = [b_{i,1} \cdots, b_{i,n}]^T$. The codeword $\mathbf{b}_i$ is then mapped to ground-truth signal vector $\mathbf{x}_i$. We label this mapping by $\mathbf{x}_i = \tilde{\mathcal{M}}(\mathbf{b}_i)$. In order to successfully detect the ground-truth signal vector $\{\mathbf{x}_i\}_1^S$, without prior knowledge about channel state information (CSI) $\{\mathbf{h}_i\}_1^S$ under AWGN, we write least square estimation problem under codeword constraints. That is, we require the solution for the signal vector to be a mapping from a valid polar codeword belonging to $\mathcal{C}$.

$$\begin{aligned} \min_{\{x_i\},\{h_i\}} \quad & \sum_{j=1}^{N} \mid y_j - \sum_{i=1}^{S} \mathbf{c}_j{}^T \mathbf{h}_i \mathbf{x}_i^T \mathbf{a}_{ij} \mid^2 \\ \text{s. t.} \quad & \mathbf{x}_i \in \mathcal{F}, \end{aligned} \tag{5}$$

where $\mathcal{F}$ is defined as a set containing all valid signal vectors, which we call *valid set*.

$$\mathcal{F} = \{\mathbf{x} \mid \tilde{\mathcal{M}}(\mathbf{b}) = \mathbf{x}, \mathbf{b} \in \mathcal{C}\} \tag{6}$$

The codeword constrained blind demixing problem in (5), is a non-convex mixed-integer problem that is difficult to be solved and inherently admits multiple local minima. The existence of local minima is evident from at least the inherent scalar ambiguity γ in each $\mathbf{h}_i \mathbf{x}_i^T = \gamma \mathbf{h}_i (\gamma^{-1} \mathbf{x}_i)^T$. To the best of our knowledge, there has been no prior attempt to take advantage of the vital FEC codeword information in the blind demixing problem thus far. We shall first review the effective Wirtinger flow algorithm for solving the optimization of (5), without the FEC codeword constraints. Based on the Wirtinger flow solution, we shall later propose a joint relaxed optimization for blind demixng by incorporating codeword information efficiently to develop an efficient and faster converging demixing receiver that is robust against noises and interferences.

D. Wirtinger Flow for Blind Demixing

Wirtinger flow algorithm is a two stage, iterative algorithm consisting of spectral initialization and standard gradient decent update procedure without regularization that can be used in blind demixing [4]. Specifically, the gradient of Wirtinger flow is represented by Wirtinger derivatives.

1) Spectral Initialization: Define matrix $\mathbf{M}_i \triangleq \sum_{j=1}^{N} y_j \mathbf{c}_j^T \mathbf{a}_{ij}$, for $i = 1 \cdots S$. Let $\sigma_1(\mathbf{M}_i)$, $\mathbf{u}_i$ and $\mathbf{v}_i$ be the leading singular value, left singular vector and right singular vector of matrix $\mathbf{M}_i$, respectively. We initialize

$$\begin{aligned} \mathbf{h}_i^0 &= \sqrt{\sigma_1(\mathbf{M_i})}\mathbf{u}_i \\ \mathbf{x}_i^0 &= \sqrt{\sigma_1(\mathbf{M_i})}\mathbf{v}_i \end{aligned} \tag{7}$$

2) Update rule: For $i = 1, \cdots, S$, $\nabla_{\mathbf{h}_i} G$ and $\nabla_{\mathbf{x}_i} G$ denote the Wirtinger gradient of the error function $G(.)$ with respect to $\mathbf{h}_i$ and $\mathbf{x}_i$, repectively:

$$\nabla_{\mathbf{h}_i} G = \sum_{j=1}^{N} \left(\sum_{i=1}^{S} \mathbf{c}_j{}^T \mathbf{h}_i \mathbf{x}_i^T \mathbf{a}_{ij} - y_j \right) \mathbf{c}_j^* \mathbf{a}_{ij}^H \mathbf{x}_i^* \tag{8a}$$

$$\nabla_{\mathbf{x}_i} G = \sum_{j=1}^{N} \left(\sum_{i=1}^{S} \mathbf{c}_j{}^T \mathbf{h}_i \mathbf{x}_i^T \mathbf{a}_{ij} - y_j \right) \mathbf{a}_{ij}^* \mathbf{c}_j^H \mathbf{h}_i^* \tag{8b}$$

For each of the signal source sequence $\mathbf{x}_i$ and the corresponding CSI $\mathbf{h}_i$, $i = 1, \cdots, S$, the Wirtinger flow algorithm updates their $t+1$-th iteration using a stepsize $\eta > 0$ via

$$\mathbf{h}_i^{t+1} = \mathbf{h}_i^t - \eta \frac{1}{\| \mathbf{x}_i^t \|_2^2} \nabla_{\mathbf{h}_i} G(\mathbf{h}_1^t, \cdots, \mathbf{h}_S^t, \mathbf{x}_1^t, \cdots, \mathbf{x}_S^t) \tag{9a}$$

$$\mathbf{x}_i^{t+1} = \mathbf{x}_i^t - \eta \frac{1}{\| \mathbf{h}_i^t \|_2^2} \nabla_{\mathbf{x}_i} G(\mathbf{h}_1^t, \cdots, \mathbf{h}_S^t, \mathbf{x}_1^t, \cdots, \mathbf{x}_S^t). \tag{9b}$$

E. Direct Mapping WF Outputs to Valid Codewords

In order to incorporate codeword information into the Wirtinger flow algorithm, our solution is to project the output $\mathbf{x}_i^t$ of the WF algorithm at iteration t for the i-th source to the valid set $\mathcal{F}$ defined in (6). Since the set $\mathcal{F}$ contains all the signal vectors that are mapped from valid bit vectors, there can be one member of $\mathcal{F}$ that is closest to the iterative output signal $\mathbf{x}_i^t$. To obtain this member, we solve the following optimization problem.

$$\mathbf{x}_i^{t,p} = \arg\min_{\mathbf{x} \in \mathcal{F}} \| \mathbf{x} - \mathbf{x}_i^t \|_2^2 \tag{10}$$

Please note that in a practical communication system, one symbol of vector $\mathbf{x}_i$ should be known to the receiver to overcome the ambiguity problem that was explained earlier. We can use that known symbol to de-rotate the vector $\mathbf{x}_i^{t,p}$ and then calculate the corresponding $\mathbf{h}_i^{t,p}$ and finally continue with WF algortihm update rule in (11). By incorporating this projection, we derive a WF algorithm based on direct codeword projection (WF-DCP)

The projection of (10) requires 2^k vector-by-vector comparison. Its computation complexity can be quite high for each iteration. As a matter of tradeoff, this projection may be executed once every α iterations. The major issue is that the constraint in (10) of WF-DCP is non-convex that makes this part NP-hard and its complexity grows exponentially with the dimension of codeword size. Therefore, it would be computationally very costly to solve (10) in WF-DCP. To address this issue, we propose to use the relaxed version of codeword constraints that was introduced in [9] for polar codes, originally to decode such codes with linear programming. We have shown in [6], [7] that

such relaxed codeword constraints can be well incorporated in real/complex field to design a more robust MIMO detector for improved signal detection in scenarios that channel noise and interferences are high.

III. Projection to Relaxed Codeword Constrains

To overcome the high computational complexity of exhaustive codeword enumeration as described in WF-DCP, one approach is to incorporate the codeword constraint into the Euclidean signal space of the WF optimization. Recall that codewords $\mathbf{b}$ of linear block codes are those binary sequences that satisfy the parity check condition $\mathbf{P} \cdot \mathbf{b} = \mathbf{0}$ that is linear in the finite binary Galois field GF(2). We propose to directly incorporate this linear GF(2) constraints into the original optimization (5) in Euclidean space.

Goela *et. al* [9] utilized the recursive structure of polar codes that leads to a sparse graph representation with $O(n \log n)$ auxiliary variables, where n is the block length. Fig. 1 shows such a factor graph of a polar code with block length $n = 2^3$. Taking advantage of this factor graph, a polytope can be defined in a space of dimension $O(n \log n)$ [9]. We shall exploit this polytope to generate a set of linear coding constraints that can be incorporated into (5).

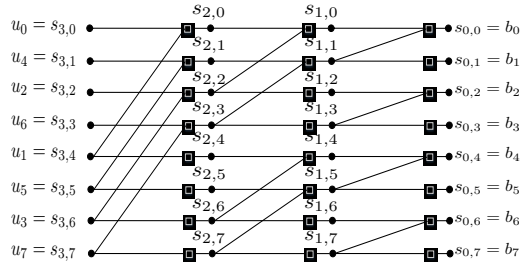


Fig. 1: factor graph representation of a polar code with block length $n = 2^3$

We denote the corresponding polytope as $\mathcal{P}$. The graph of Fig. 1 shows how a polar codeword $\mathbf{b}$ can be constructed from binary vector $\mathbf{u}$ by a 1-1 mapping through the generator matrix $\mathbf{G_n}$, $\mathbf{b} = \mathbf{u}\mathbf{G_n}$. The circle nodes on the graph represent a total of $n(1+\log n)$ binary variables and the square nodes represent the check nodes. If all the check nodes are satisfied, then $\mathbf{b}$ is a valid codeword.

An example of a check node constraint in Fig. 1 is $u_0 \oplus u_1 \oplus s_{2,0} = 0$, where $\oplus$ denotes modulo-2 addition. To define the relaxed polytope $\mathcal{P}$, we let the variables in the graph be real variables instead of binary. Note that each constraint involves only either 3 or 2 variables. Therefore, for each check node $j \in \mathcal{J}$ with 3 neighbors $\mathcal{N}(j) = \{a_1, a_2, a_3\}$, the local minimal convex polytope of check node j is $\mathcal{P}_j$, which can be very simply defined by the following linear inequalities

$$\begin{aligned} 0 &\le a_1 \le a_2 + a_3 \\ 0 &\le a_2 \le a_3 + a_1 \\ 0 &\le a_3 \le a_1 + a_2 \\ a_1 &+ a_2 + a_3 \le 2 \end{aligned} \tag{11}$$

For each check node $j \in \mathcal{J}$ with only two neighbors $\mathcal{N}(j) = \{a_1, a_2\}$, the local polytope $\mathcal{P}_j$ is defined by

$$\begin{aligned} a_1 &= a_2 \\ 0 &\le a_1 \le 1 \\ 0 &\le a_2 \le 1 \end{aligned} \tag{12}$$

Moreover, we denote the cutting plane $\mathcal{T}$ as defined by setting all frozen variables with indices belonging to $\mathcal{I}^c$ to zero. In summary, the polytope $\mathcal{P}$ is the intersection of all local polytopes plus the cutting plane $\mathcal{T}$ via

$$\mathcal{P} = \left(\bigcap_j \mathcal{P}_j \right) \cap \mathcal{T} \tag{13}$$

Therefore, we can write down the linear coding constraints by enforcing all the variables of the factor graph to comply with the polytope $\mathcal{P}$, i.e. $\mathbf{s} \in \mathcal{P}$ where $\mathbf{s}$ denotes all the variables of the factor graph. These constraints can be incorporated into (10) as relaxed version of $\mathbf{x} \in \mathcal{F}$. Therefore, we can define a new polytope $\mathcal{X}$ that we call it *relaxed valid set* based on polytope $\mathcal{P}$ as follows

$$\mathcal{X} = \{\mathbf{x} \mid \mathbf{s} \in \mathcal{P}, \tilde{\mathcal{M}}(\mathbf{b}) = \mathbf{x}\}. \tag{14}$$

Finally, we can relax the codeword constraints using this polytope and write a quadratic programming (QP) optimization problem with linear constraints to project to the relaxed valid set.

$$\begin{aligned} \min_{\mathbf{x}} \quad & \| \mathbf{x} - \mathbf{x}_i^t \|_2^2 \\ \text{s.t.} \quad & \mathbf{x} \in \mathcal{X} \end{aligned} \tag{15}$$

Therefore, we propose to use projection defined in (15) to be done at each α iteration of WF algorithm instead of (10) to take advantage of codeword information.

A. Projection Takes the Iterate Closer to the Ground-Truth

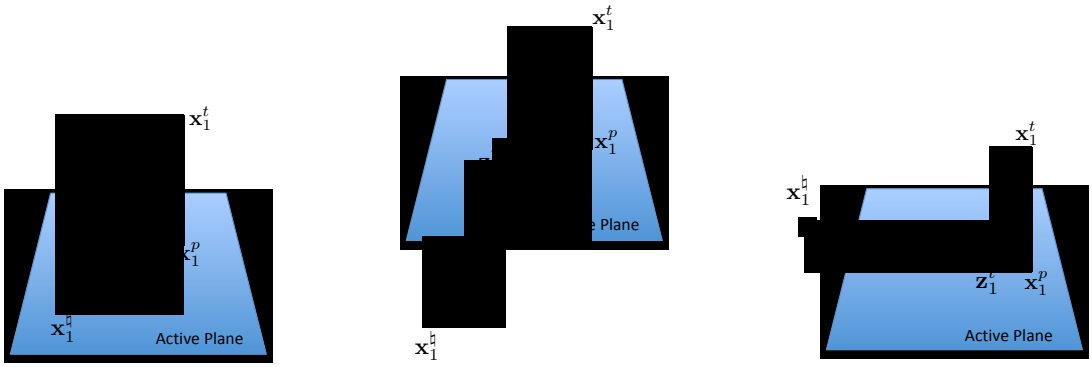
The convergence of Wirtinger flow algorithm was proved in [4] by showing that iterates of Wirtinger flow, stay in the region of incoherence and contraction by exploiting the local geometry of blind demixing problem.

In this section, we consider the noiseless case, and we prove after each projection, the l_2 norm distance of iterates of WF algorithm to the ground-truth becomes smaller. From (13), we know that, $\mathcal{X}$ is a polytope consisting of multiple planes and also the ground-truth signal vectors $\{\mathbf{x}_i^\natural\}_1^S$ is one of the vertices of this polytope. Without loss generality, we prove for $i = 1$ case. Let's denote the projection of $\mathbf{x}_1^t$ to $\mathcal{X}$ by $\mathbf{x}_1^p$, i.e., $\mathbf{x}_1^p$ is the solution of (10). Our goal is to prove

$$\| \mathbf{x}_1^p - \mathbf{x}_1^\natural \|_2 \le \| \mathbf{x}_1^t - \mathbf{x}_1^\natural \|_2 \tag{16}$$

If the current iteration $\mathbf{x}_1^t$ is inside $\mathcal{X}$, then $\mathbf{x}_1^t$ already satisfies the code constraints and projection will not change it, i.e.,

$$\| \mathbf{x}_1^p - \mathbf{x}_1^\natural \|_2 = \| \mathbf{x}_1^t - \mathbf{x}_1^\natural \|_2$$



(a) Case 1 (b) Case 2 (c) Case 3

Fig. 2: Three cases with regard to the position of $\mathbf{x}_1^p$ and the active plane

We are interested in the case when $\mathbf{x}_1^t$ is outside polytope $\mathcal{X}$. We know $\mathbf{x}_1^p$ lies on one of the planes of $\mathcal{X}$ that we call it the *active plane*. There are only three possible cases with regard to the position of $\mathbf{x}_1^p$ and the active plane. We consider each of them.

1) $\mathbf{x}_1^p$ lies on the active plane as depicted in Fig. 2a. In this case, since $(\mathbf{x}_1^t - \mathbf{x}_1^p)$ is orthogonal to the active plane, we can use Pythogoras theorem to write:

$$\| \mathbf{x}_1^t - \mathbf{x}_1^\natural \|_2^2 = \| \mathbf{x}_1^t - \mathbf{x}_1^p \|_2^2 + \| \mathbf{x}_1^p - \mathbf{x}_1^\natural \|_2^2$$

and therefore (16) holds in this case.

2) $\mathbf{x}_1^t$ and $\mathbf{x}_1^\natural$ are on two different sides of the active plane as depicted in Fig. 2b. Let us draw a distinction between $\mathbf{x}_1^\natural$ and $\mathbf{x}_1^t$, and call its intersection with the active plane $\mathbf{z}_1^t$. From triangle inequality, we know

$$\| \mathbf{x}_1^\natural - \mathbf{x}_1^p \|_2 \leq \| \mathbf{z}_1^t - \mathbf{x}_1^p \|_2 + \| \mathbf{z}_1^t - \mathbf{x}_1^\natural \|_2 \tag{17}$$

Similar to case 1, we can use Pythogoras to show $\| \mathbf{z}_1^t - \mathbf{x}_1^p \|_2 \leq \| \mathbf{z}_1^t - \mathbf{x}_1^t \|_2$. Therefore, by replacing it into the right hand-side of (18), we get

$$\begin{aligned} \| \mathbf{x}_1^\natural - \mathbf{x}_1^p \|_2 &\leq \| \mathbf{z}_1^t - \mathbf{x}_1^t \|_2 + \| \mathbf{z}_1^t - \mathbf{x}_1^\natural \|_2 \\ &= \| \mathbf{x}_1^\natural - \mathbf{x}_1^t \|_2 \end{aligned} \tag{18}$$

Hence, (16) holds in case 2 as well.

3) $\mathbf{x}_1^t$ and $\mathbf{x}_1^\natural$ are on the same side of the active plane as depicted in Fig. 2c. We prove that this case cannot happen due to convexity of $\mathcal{X}$ by contradiction. We assume $\mathbf{x}_1^t$ and $\mathbf{x}_1^\natural$ are on the same side of the active plane. We label the projection of $\mathbf{x}_1^t$ to the line that passes through $\mathbf{x}_1^\natural$ and $\mathbf{x}_1^p$ as $\mathbf{z}_1^t$. Since both $\mathbf{x}_1^\natural$ and $\mathbf{x}_1^p$ belong to the convex set $\mathcal{X}$, any point on the line segment between the two should also be in the set $\mathcal{X}$, therefore, $\mathbf{z}_1^t \in \mathcal{X}$. However, we know that $\mathbf{x}_1^p$ is the closest point on $\mathcal{X}$ to $\mathbf{x}_1^t$. Therefore, $\mathbf{x}_1^p$ and $\mathbf{z}_1^t$ have to be the same, i.e., $(\mathbf{x}_1^t - \mathbf{x}_1^p)$ must be orthogonal to $(\mathbf{x}_1^\natural - \mathbf{x}_1^p)$. However, this results in the fact that $\mathbf{x}_1^\natural$ lies on the active plane such as case 1. Therefore, case 3 cannot happen and (16) is proven. In other words, by doing the projection of (15) on iterations of Wirtinger flow $\mathbf{x}_1^t$, they only get closer to the ground-truth. Therefore, as we will also confirm by simulations in the next section, Wirtinger flow will converge faster.

IV. Simulation Results

We first present a set of simulation tests and results to test our proposed algorithms to demonstrate its capability to improve Wirtinger flow algorithm. Throughout this section, we utilized the MOSEK solver [10] to solve the QP in our simulations. We compare the convergence rate and also robustness against noise of regular Wirtinger flow with the proposed algorithm. To be exact in our comparison, same scenario is applied to all algorithms including same white noise and same channel realizations.

We assumed $S = 4$, i.e., 4 sources are sending signals at the same time. In particular, we look at the average error of our source signals with respect to their estimates, defined by

$$e = \sum_{i=1}^{S} \| \mathbf{x}_i^\natural - \hat{\mathbf{x}}_i \|_2$$

where $\mathbf{x}_i^\natural$ is the ground-truth signal and $\hat{\mathbf{x}}_i$ is its estimate. Channel maximum delay tap is assumed to be $L = 16$. We chose a polar code of length $n = 32$ with rate $= \frac{1}{2}$. Therefore, using a QPSK symbol mapping, signal vector would have dimesion $K = 16$. The number of observation is selected to be $N = 300$ to ensure that the WF algorithm is able to converge and find a solution as discussed in more details in [4]. The stepsize is set to $\eta = 0.0004$ to guarantee that the algorithm converges and does not go to infinity.

To see the effectiveness of using codeword constraints in the Wirtinger flow algorithm, we have plotted the estimation error e versus iteration count in a noiseless scenario. In Fig. 3, QPWF with α equal to 20 and 5 is compared against WF without getting any help from channel decoder. As we can see the algorithm with more projections to the valid codeword set converges faster.

In Fig. 4 we compare the average error of all 4 signals for two algorithms in different SNRs. We assume polar code rate of 0.5 and 0.25. We can see that adding codeword constraints helps WF to save more than 5dB in code rate of 0.5 compared

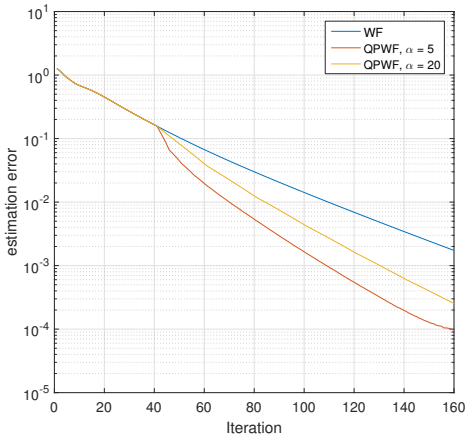


Fig. 3: Convergence rate comparison for QPWF for α values in a polar FEC scenario

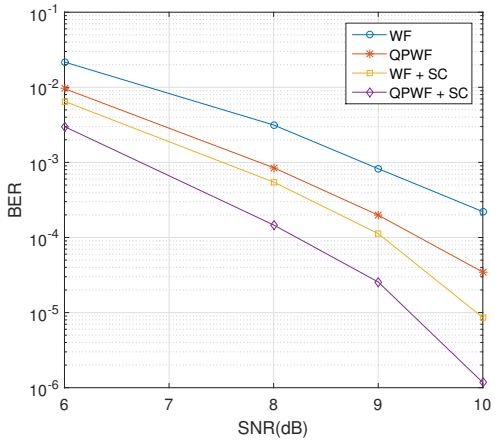


Fig. 5: Bit error rate comparison for QPWF versus WF algorithms before and after polar SC decoder

to regular WF. This gap goes up to 9dB in code rate 0.25 case. This was expected due to more information that code constraints will provide in case of lower code rates.

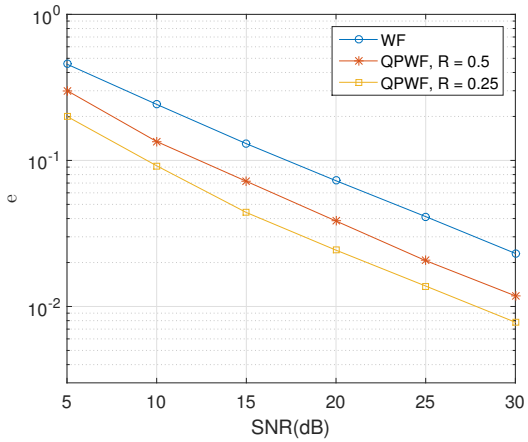


Fig. 4: Average signal error for QPWF versus Regular Wirtinger flow algorithm for code rate 0.5 and 0.25

To further complete the simulations, signal estimates are converted to log likelihood ratios and then fed to a successive cancellation decoder. We shall evaluate how much we improve the bit error rate at the decoder output, using our proposed method. Fig. 5 shows that we can save 1.2 dB of power by incorporating code constraints into WF algorithm at BER 10^{-5}.

V. Conclusion

This work considered the problem of blind demixing for a wireless communication system in which signals are being detected without any knowledge of channel. To make detection more robust against noise and practical obstacles of the channel, we proposed to incorporate useful codeword information into well-known Wirtinger flow algorithm. Considering polar codes as FEC for short data packages, our method incorporated codeword information by projecting iterations of Wirtinger flow to a valid codeword set. Our simulation results indicates clearly improved performance of this method compared to simple Wirtinger flow in noisy channels, both in terms of bit-error-rate and signal detection error. Moreover, we theoretically proved that by imposing codeword constraints, the iterations of Wirtinger flow get closer to the ground-truth and therefore, speed up overall convergence rate of the algorithm. Future works might focus on other methods of incorporating codeword information for Wirtinger flow algorithm or other well-known blind demixing algorithms.

References

[1] Z. Ding and Y. Li, *Blind equalization and identification*. CRC press, 2018.

[2] S. Ling and T. Strohmer, "Blind deconvolution meets blind demixing: Algorithms and performance bounds," *IEEE Trans. on Inf. Theory*, vol. 63, no. 7, pp. 4497–4520, 2017.

[3] A. Ahmed, B. Recht, and J. Romberg, "Blind deconvolution using convex programming," *IEEE Trans. on Inf. Theory*, vol. 60, no. 3, pp. 1711–1732, 2014.

[4] J. Dong and Y. Shi, "Nonconvex demixing from bilinear measurements," *IEEE Transactions on Signal Processing*, vol. 66, no. 19, pp. 5152–5166, 2018.

[5] E. J. Candes, X. Li, and M. Soltanolkotabi, "Phase retrieval via wirtinger flow: Theory and algorithms," *IEEE Trans. on Inf. Theory*, vol. 61, no. 4, pp. 1985–2007, 2015.

[6] A. Jalali and Z. Ding, "A joint detection and decoding receiver design for polar coded MIMO wireless transmissions," in *IEEE Int. Symp. Inf. Theory (ISIT)*. IEEE, 2018, pp. 1011–1015.

[7] ——, "Joint detection and decoding of polar coded 5G control channels," *IEEE Trans. on Wireless Commun.*, 2020.

[8] E. Arıkan, "Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels," *IEEE Trans. Inf. Theory*, vol. 55, no. 7, pp. 3051–3073, 2009.

[9] N. Goela, S. B. Korada, and M. Gastpar, "On LP decoding of polar codes," in *Inf. Theory Workshop (ITW), 2010 IEEE*. IEEE, 2010, pp. 1–5.

[10] E. Andersen and K. Andersen, "The MOSEK optimization toolbox for MATLAB manual. version 7.0, 2013."