CS Journal Club, July 11, 2013

A sparse signal reconstruction perspective for source localization with sensor arrays

Authors: Dmitry Malioutov, Mujdat Cetin, and Alan S. Willsky
Journal: IEEE Trans. on. Sign. Proc. Aug., 2005.
Presenter: J. Oliver

Abstract

In this paper, the authors present a source localization method based on sparse representation of sensor measurements. In particular, they use SVD of the data matrix obtained from the sensors to summarize the multiple measurements. The SVD summarized data is then sparsely represented in order to detect the sources. The authors also proposed grid refinement in order to mitigate the effects of limiting estimates to a grid of spatial locations. They demonstrate the superior resolution ability with limited time samples of their method over the existing methods via various experiments.

Introduction and Background

- Source localization methods deal with finding the closely spaced sources in presence of considerable noise.
- Many advanced techniques for the localization of sources achieve super-resolution by exploiting the presence of a small number of sources. For example, the key component of the MUSIC method is the assumption of a low-dimensional signal subspace.
- Estimating the spatial locations (or directions) is a well-known problem in array signal processing.
- Three major source estimation techniques are 1. Classical methods (beamformer, MVDR) 2. Subspace methods (MUSIC, ESPRIT) 3. ML-based methods (deterministic and stochastic).
- Beamforming is simple but its resolution is limited. Subspace methods achieve super resolution, provided SNR is moderately high and sources are not strongly correlated and the number of snapshots (measurement vectors) are sufficient. ML techniques are superior than the subspace methods but require accurate initialization for global convergence.
- By turning to the sparse signal representation framework, the authors are able to achieve super-resolution without the need for a good initialization, without a large number of time samples, and with lower sensitivity to SNR and to correlation of the sources.
- The authors have developed the method for narrowband case and discussed in brief how it can be used for wideband source localization.
- Prior research has established sparse signal representation as a valuable tool for signal processing, but its application to source localization has been developed only for very limited scenarios. For example, [1, 2] is concerned with source localization in the beam-space domain, under the assumption that the sources are uncorrelated, and that a large number of time samples is available.

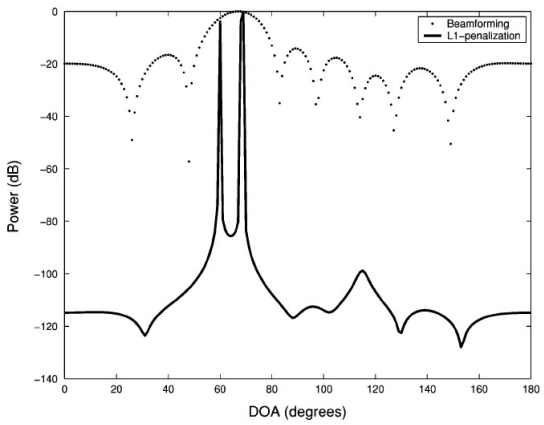

Fig. 1. Single sample source localization with ℓ_1. Spatial spectra of two sources with DOAs of 60° and 70° (SNR = 20 dB).

- In its most basic form, the problem of sparse signal representation in overcomplete bases asks to find the sparsest signal x to satisfy $y = Ax$, where $A \in C^{M\times N}$ is an overcomplete basis, i.e., $M < N$.Without the sparsity prior on x, the problem $y = Ax$ is ill-posed and has infinitely many solutions. Additional information that x should be sufficiently sparse allows one to get rid of the ill-posedness.

Source localization framework

- The goal of the source localization is to find locations of sources of wavefields that impinge on an array of sensors that are seperated by a distance less than or equal to $\lambda/2$
- Consider K narrowband signals $u_k(t), k \in \{1, 2, \cdots, K\}$, arriving at an array of M sensors, after being corrupted by additive noise $n_m(t)$, resulting in sensor outputs $y_m(t), m \in \{1, 2, \cdots, M\}$. After demodulation, the vector form of the received signal is

$$\boldsymbol{y}(t) = \boldsymbol{A}(\boldsymbol{\theta})\boldsymbol{u}(t) + \boldsymbol{n}(t), \quad t \in \{t_1, \cdots, t_T\} \tag{1}$$

- $\boldsymbol{A}(\boldsymbol{\theta})$ is array manifold matrix. The $(m, k)^{th}$ element $\boldsymbol{A}$ contains the delay and gain information from the kth source (at location θ_k) to the mth sensor. The column, $a(\theta_k)$, of $\boldsymbol{A}$ are called steering vectors and is given by $a(\theta_k) = \left[e^{j\frac{2\pi}{\lambda}1\sin\theta_k}, e^{j\frac{2\pi}{\lambda}2\sin\theta_k}, \cdots, e^{j\frac{2\pi}{\lambda}M\sin\theta_k}\right]^T$
- Any source localization method aims to find the unknown locations of the sources $\theta_k, \forall k$, given $\boldsymbol{y}(t)$ and $\boldsymbol{A}$.
- We note that finding $\boldsymbol{\theta}$ is a non-linear estimation problem.

Sparse representation for a single time sample, that is, $T = 1$

- To cast a sparse representation problem, the authors introduce an overcomplete representation of $\boldsymbol{A}$ in terms of all possible source locations.
- Let $\{\tilde{\theta}_1, \tilde{\theta}_2, \cdots, \tilde{\theta}_N\}$ be a sampling grid of all source locations of interest.
- The number of potential sources N will typically be much greater than the number of actual sources K and the number of sensors M.

- A matrix composed of steering vectors corresponding to each potential source location as its columns constitute an over-complete dictionary, that is, $\boldsymbol{A} = \left[a(\tilde{\theta}_1), a(\tilde{\theta}_2), \cdots, a(\tilde{\theta}_N)\right]$. We note that $\boldsymbol{A}$ is known and does not depend on the actual source locations.

- The signal vector is $\boldsymbol{s}(t)$ with the nth element $s_n(t) = u_k(t)$ if the source k comes from θ_n for some k and zero otherwise. For $T = 1$, then the source localization problem reduces to

$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{s} + \boldsymbol{n} \tag{2}$$

- In effect, this overcomplete representation allows us to exchange the problem of parameter estimation of $\boldsymbol{\theta}$ for the problem of sparse spectrum estimation of $\boldsymbol{s}$.

- With the key assumption that the source numbers are less, the underlying spatial spectrum is sparse (i.e., has only a few nonzero elements), and hence we can solve this inverse problem via l_1 methodology, $min\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{s}\|_2^2 + \lambda\|\boldsymbol{s}\|_1$

- The data for the model is complex-valued; hence, neither linear nor quadratic programming can be used for numerical optimization. Instead, the authors adopt an SOC programming framework and find $\boldsymbol{s}$. Once $\boldsymbol{s}$ is found, the estimates of the source locations correspond to the locations of the peaks in $\boldsymbol{s}$.

Source location with multiple time samples and $l_1 - SVD$

- Source localization with multiple snapshots from potentially correlated sources is of greater practical importance.

- When we bring time into the picture, the overcomplete representation is easily extended and it has the following form:

$$\boldsymbol{y}(t) = \boldsymbol{A}\boldsymbol{s}(t) + \boldsymbol{n}(t), \quad t \in \{t_1, t_2, \cdots, t_T\} \tag{3}$$

Single and Joint inverse problem

- The first thought that comes to mind when we switch from one time sample to several time samples is to solve each problem indexed by separately. In that case, we would have a set of solutions $\hat{\boldsymbol{s}}(t)$.

- If the sources are moving fast, then the evolution of the sources is of interest, and the approach is suitable for displaying it.

- When the sources are stationary over several time samples, then it is preferable to combine the independent estimates to get one representative estimate of source locations from them, for example, by averaging or by clustering.

- Now, we consider a simple approach that uses different time samples together. Let $\boldsymbol{Y} = [\boldsymbol{y}(t_1), \boldsymbol{y}(t_2), \cdots, \boldsymbol{y}(t_T)]$, and define $\boldsymbol{S}$ and $\boldsymbol{N}$ similarly. Then, we have

$$\boldsymbol{Y} = \boldsymbol{A}\boldsymbol{S} + \boldsymbol{N} \tag{4}$$

- We note that the matrix $\boldsymbol{S}$ is parametrized temporally and spatially, but sparsity only has to be enforced in time not in space.

- To accommodate this issue in the optimization problem, the authors first compute the l_2 norm of all time-samples of a particular space index of s, that is, $\boldsymbol{s}_i^{l_2} = \|[s_i(t_1), s_i(t_2), \cdots, s_i(t_T)]\|_2$.

- Then the authors minimize the l_1 norm of $\boldsymbol{s}^{l_2} = \left[s_1^{l_2}, s_2^{l_2}, \cdots, s_N^{l_2}\right]$. Now the problem becomes

$$min\|\boldsymbol{Y} - \boldsymbol{AS}\|_f^2 + \lambda\|\boldsymbol{s}^{l_2}\|_1 \tag{5}$$

- Note in Eqn. (5), the optimization is performed over the matrix $\boldsymbol{S}$ and once the estimate of $\boldsymbol{S}$ is computed the peaks of $\boldsymbol{S}$ provide the source locations.
- The main drawback of this technique is its computational cost. The size of the inverse problem increases linearly with T, and the computational effort required to solve it increases superlinearly with T. In order to alleviate this, the authors propose a SVD based solution.

l_1- SVD

- To reduce both the computational complexity and the sensitivity to noise, the authors propose to use the SVD of the $M \times T$data matrix $\boldsymbol{Y}$.
- The idea is to decompose the data matrix into the signal and noise subspaces.
- With the signal subspace, mold the problem as multiple-vector sparse spectrum estimation problem similar to Eqn. (4).
- Without noise on the sensors, the set of vectors of $\boldsymbol{Y}$ would lie in a K-dimensional subspace.
- If we can relate the basis of this K-dimensional subspace (set of K vectors) to the source matrix $\boldsymbol{S}$, then we can just keep K vectors (instead of T) for the estimation problem.
- Take the SVD $\boldsymbol{Y} = \boldsymbol{ULV'}$ and form a $M \times K$ dimensional matrix $\boldsymbol{Y}_{sv}$ as $\boldsymbol{Y}_{sv} = \boldsymbol{YVD}_k$, where $\boldsymbol{D}_k$ is an $T \times K$ matrix given as $\boldsymbol{D}_k = [\boldsymbol{I}_K \boldsymbol{0'}]$
- Now $\boldsymbol{Y}_{sv}$ can be written as

$$\begin{aligned} \boldsymbol{Y}_{sv} &= \boldsymbol{YVD}_k \\ &= (\boldsymbol{AS} + \boldsymbol{N})\boldsymbol{VD}_k \\ &= \boldsymbol{ASVD}_k + \boldsymbol{NVD}_k \\ &= \boldsymbol{AS}_{sv} + \boldsymbol{N}_{sv} \end{aligned} \tag{6}$$

- We note that the sparsity structure of $\boldsymbol{S}$ is retained in $\boldsymbol{S}_{sv}$.
- Considering the k-th column of Eqn. (6) we have

$$\boldsymbol{y}^{sv}(k) = \boldsymbol{A}\boldsymbol{s}^{sv}(k) + \boldsymbol{n}^{sv}(k), \quad k = 1, 2, \cdots, K \tag{7}$$

  This is exactly the same form as multiple-vector model in Eqn. (3), expect that indexing is by singular vector, k.
- By bringing SVD, the problem size is reduced from T to K. This reduction is substantial, because in typical situations $K \ll T$.
- Now in the matrix $\boldsymbol{S}_{sv}$, the sparsity is along the spatial domain and not in the singular vector domain.
- To accommodate the true sparsity in the minimization problem, the authors define $\tilde{s}_i^{l_2} = \|[s_i^{sv}(1), s_i^{sv}(2), \cdots, s_i^{sv}(K)\|_2$. The sparsity of the $N \times 1$ vector $\tilde{s}_i^{l_2}$ is the sparsity of the spatial spectrum, which can be found by minimizing

$$\|\boldsymbol{Y}_{sv} - \boldsymbol{AS}_{sv}\|_f^2 + \lambda\|\tilde{\boldsymbol{s}}^{l_2}\|_1 \tag{8}$$

- In this paper, the authors have solved the above problem using SOC programming (see paper for details)

Multi-resolution grid refinement

- Thus far, in this paper, the estimates of the source locations are confined to a grid.
- We cannot make the grid very fine uniformly since this would increase the computational complexity and also the columns of $\boldsymbol{A}$ becomes more linearly dependent.
- Hence, the authors explore the idea of adaptively refining the grid in order to achieve better precision
- Instead of having a universally fine grid, we make the grid fine only around the regions where sources are present.
- This requires an approximate knowledge of the locations of the sources, which can be obtained by using a coarse grid first.

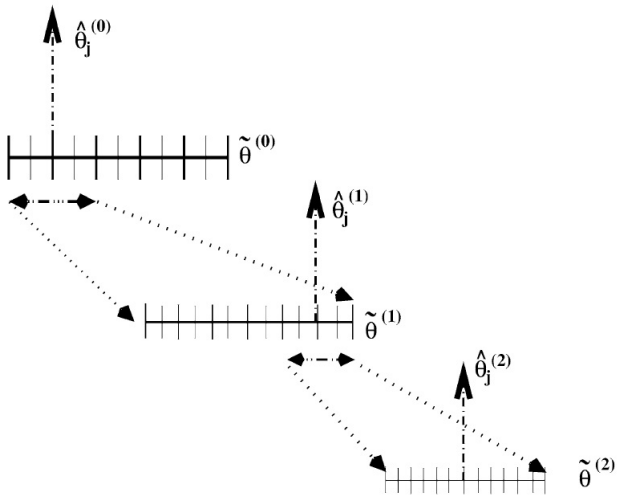

Fig. 3. Illustration of grid refinement.

- The grid refinement algorithm goes like this

1. Create a rough grid of potential source locations $\tilde{\theta}^{(0)}$, for $i = 1, 2, \cdots, N$. Set $r = 0$.
2. Form $\boldsymbol{A}_r = \boldsymbol{A}(\tilde{\boldsymbol{\theta}}^{(r)})$, where $\tilde{\boldsymbol{\theta}}^{(r)} = \left[\tilde{\theta}_1^{(r)}, \tilde{\theta}_2^{(r)}, \cdots, \tilde{\theta}_N^{(r)}\right]$. Use the SOC minimization to find the estimates of the source locations and set $r = r + 1$.
3. Get a refined grid $\tilde{\boldsymbol{\theta}}^{(r)}$ around the locations of the peak, $\hat{\theta}_j^{(r-1)}$ (explained below).
4. Return to step 2, until the grid is fine enough.

- There are many ways of refining the grid; the authors have chosen a simple equispaced grid refinement.
- Suppose at step r, we have a uniform grid with spacing δ_r. Also, we have an estimate $\hat{\theta}_j^{(r)}$

- Pick an interval around the jth detected source with two grid spacing on either side, that is, $[\hat{\theta}_j^{(r)} - 2\delta_r, \hat{\theta}_j^{(r)} + 2\delta_r]$, for $j = 1, 2, \cdots, K$.
- In the intervals around the peak, select a new grid whose spacing is a fraction of the old one $\delta_{r+1} = \delta_r/\gamma$

Simulation results

- The authors consider $M = 8$ sensors separated by half a wavelength. $K = 2 \quad (62°, 65°), T = 200, N = 180$.

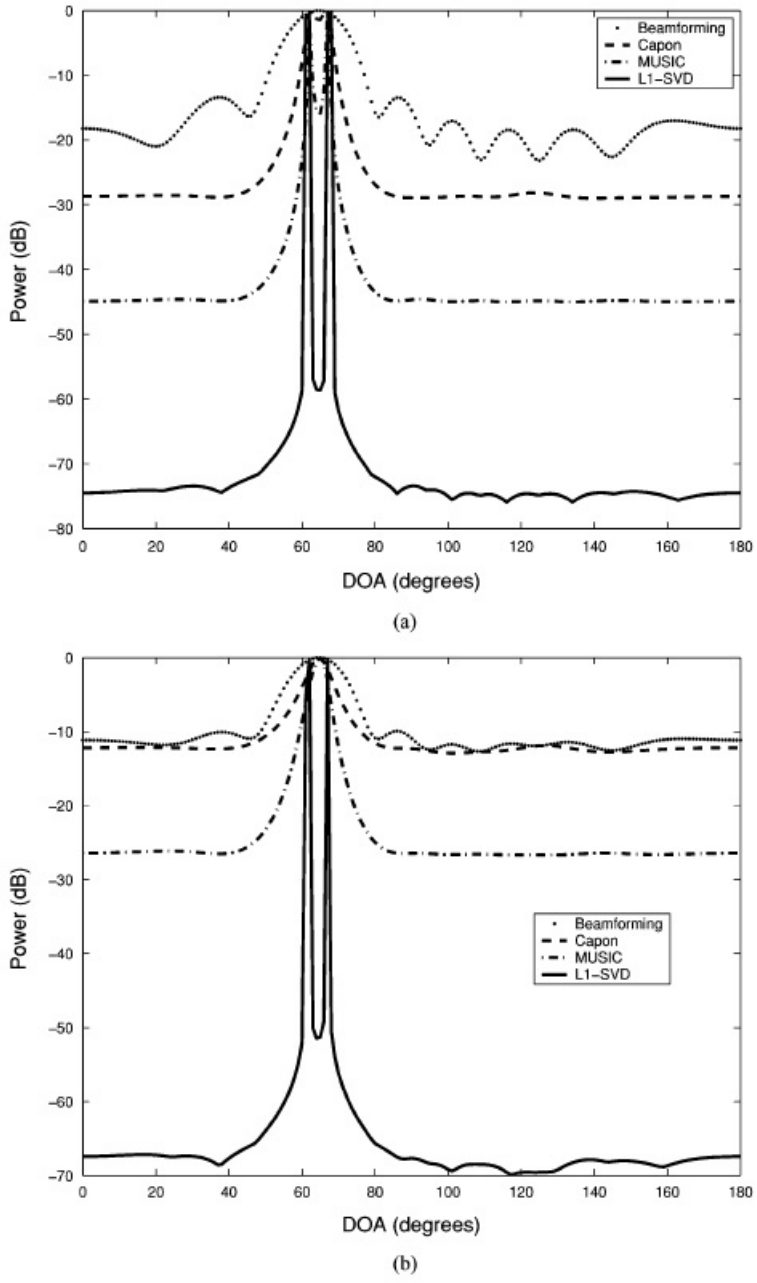

Fig. 4. (a) and (b). Spatial spectra for beamforming, Capon's method, MUSIC, and the proposed method (ℓ_1-SVD) for uncorrelated sources. DOAs: 62° and 67°. Top: SNR = 10 dB. Bottom: SNR = 0 dB.

- For correlated sources, the result is as follows

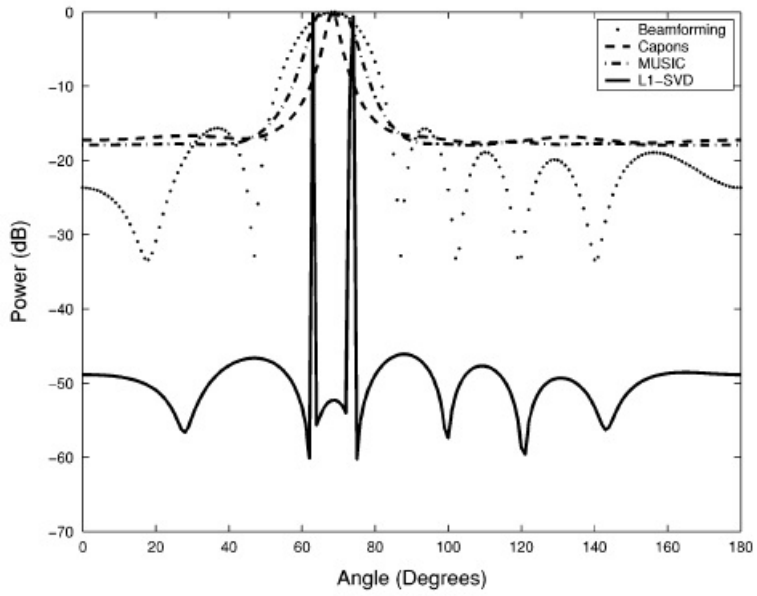

Fig. 5. Spectra for correlated sources. SNR = 20 dB. DOAs: 63° and 73°.

summary

In this paper, the authors have proposed a source location estimation based on sparse representation. The SVD of the sensor measurements summarizes the large chunk of data which is then used as a model for identifying the sources. This method is applicable for both narrow and wideband beamforming. The authors have also presented a grid refinement method in order to obtain fine estimates. The advantages of the proposed method include superior resolution ability with limited time samples for both correlated and uncorrelated sources.

References

[1] J. J. Fuchs, "Linear programming in spectral estimation. Application to array processing," in: Proc. IEEE Int. Conf. Acoust., Speech, Signal Process., vol. 6, 1996, pp. 3161-3164.

[2] J. J. Fuchs, "On the application of the global matched filter to DOA estimation with uniform circular arrays," IEEE Trans. Signal Process., vol. 49, no. 4, pp. 702-709, Apr. 2001.

Enhancing Iterative Decoding of Cyclic LDPC Codes Using Their Automorphism Groups

Authors: Chao Chen, Baoming Bai, Xinquan Yang, Li Li, Yang Yang
Publication: IEEE T. Comm, June 2013
Speaker: Jeong-Min Ryu

Short summary: For cyclic LDPC codes, they propose to use their automorphism groups to improve the iterative decoding performance. Three types of iterative decoders are devised to take advantage of the code's automor-phism group. Towards exploiting the automorphism group of a code, they propose a new class of cyclic LDPC codes based on pseudo-cyclic MDS codes with two information symbols, for which nonequivalent parity-check matrices are obtained. Simulation results show that for their constructed codes of short lengths, the automorphism group can significantly enhance the iterative decoding performance.

I. INTRODUCTION

- **The use of automorphism group for classical codes**

Most classical codes are defined by high-density parity-check (HDPC) matrices, whose Tanner graphs have a large number of short cycles.

→ Iterative decoding performs rather poorly for these codes.

→To mitigate the deleterious effect of short cycles, Jiang and Narayanan [3] and Kothiyal et al. [4] proposed adaptive versions of iterative decoding, respectively.

→As a result, the performance was greatly improved. However, a significant increase in decoding complexity was incurred.

Classical codes are known to have a very rich algebraic structure.

→ To overcome the adverse effect of short cycles while maintaining a reasonable complexity, the automorphism group, as a code structure, was exploited for iterative decoding

→ For HDPC and moderate-density parity-check (MDPC) codes, the automorphism group aided iterative decoding techniques are applied.

- **In this paper,**

1) they apply automorphism group aided iterative decoding techniques to cyclic LDPC codes.

2) For a cyclic code, two particular subgroups of the automorphism group are well known. They show that for a large class of cyclic LDPC codes [15]-[18], [20], the two subgroups of the automorphism group belong to the same equivalence class and thus cannot be harnessed for iterative decoding.

3) They present a class of cyclic LDPC codes for which the automorphism group can be exploited for iterative decoding.

II. How to Use the Automorphism Group of a Code in Iterative Decoding

A. The Automorphism Group of a Code

Definition: Let C be a binary linear block code of length N. The set of coordinate permutations that map C to itself forms a group under the composition operation. This group is called the automorphism group of C, denoted by Aut(C)

For a permutation $\pi \in \mathrm{Aut}(C)$, let π^{-1} denote its inverse. From the definition we know that for any $c = (c_0, c_1, ..., c_{N-1}) \in C$, $\pi c = \left(c_{\pi^{-1}(0)}, c_{\pi^{-1}(1)}, ..., c_{\pi^{-1}(N-1)}\right) \in C$.

Let $C^{\perp}$ denote the dual code of C, then the following property holds.

Property 1: $\mathrm{Aut}(C) = \mathrm{Aut}(C^{\perp})$.

Property 2: For any $\pi \in \mathrm{Aut}(C)$ and a parity check matrix H of C, πH also forms a parity check matrix of C.

Property 3: For a binary cyclic code with odd length N, the automorphism group contains the following two subgroups:

S_0: The set of permutations $\tau^0, \tau^1, ..., \tau^{N-1}$, where $\tau^k : j \to (j+k) \mod N$.

S_1: The set of permutations $\zeta^0, \zeta^1, ..., \zeta^{N-1}$, where $\zeta^k : j \to (2^k \cdot j) \mod N$ and m_1 is the smallest positive integer such that $2^{m_1} \equiv 1 \mod N$.

B. Two Perspectives and Their Equivalence

Using the automorphism group of a code for decoding has a long history. In the early 1960s, MacWilliams devised a hard-decision decoding procedure, called the *permutation decoding* [14]. Recently, the code's automorphism group was brought into use in the soft-decision iterative decoding of HDPC codes [5]−[9] and MDPC codes [10]. Here, they review two possible perspectives involved and show their equivalence.

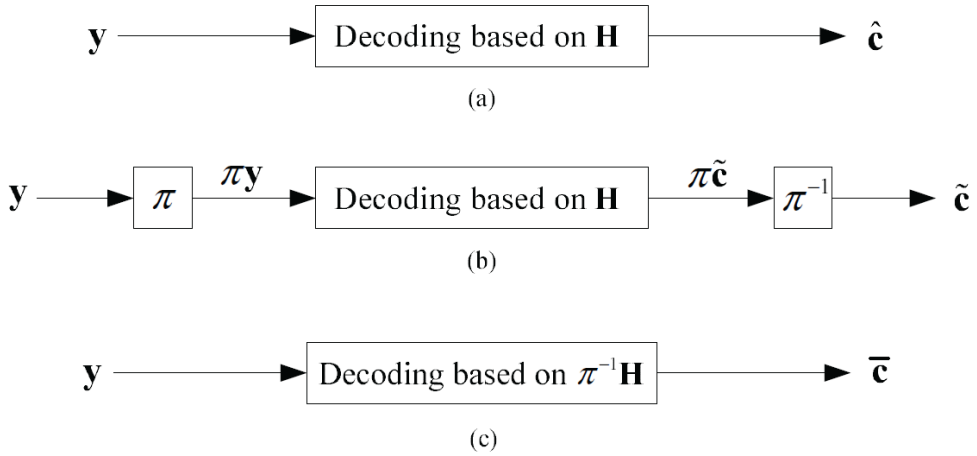

Fig. 1. Iterative decodings of the received sequence based on the SPA with flooding schedule.

(a) Assume that the BPSK signaling is used over the AWGN channel. Let $c \in C$ be the transmitted codeword and $x \in \{\pm 1\}^N$ the corresponding modulated sequence. Then the received signal sequence is given by

$$y = x + n,$$

where n contains N i.i.d. Gaussian noise samples with zero mean and variance σ^2.

(b) Applying a permuation $\pi \in \mathrm{Aut}(C)$, we have

$$\pi y = \pi x + \pi n.$$

Fact 1: In Fig. 1(a) and Fig. 1(b), the outputs $\hat{c}$ and $\tilde{c}$ are not necessarily equal. It is possible that only one of $\hat{c}$ and $\tilde{c}$ is the transmitted codeword.

Fact 2: In Fig. 1(a) and Fig. 1(c), the outputs $\hat{c}$ and $\bar{c}$ are not necessarily equal. It is possible that only one of $\hat{c}$ and $\bar{c}$ is the transmitted codeword.

Fact 3: In Fig. 1(b) and Fig. 1(c), the outputs $\tilde{c}$ and $\bar{c}$ are equal.

C. A partition of the Automorphism Group

We call two parity-check matrices *equivalent* if they can be obtained from each other through row permutations; otherwise, we call them *nonequivalent*. Since the flooding schedule is assumed, we further have

Fact 4: In Fig. 1(a) and Fig. 1(c), if H and πH are equivalent, then the outputs $\hat{c}$ and $\bar{c}$ are equal.

Let $\pi_1, \pi_2 \in Aut(C)$, then π_1 and π_2 belong to the same equivalence class if and only if $\pi_1 H$ and $\pi_2 H$ are equivalent. Note that the partition depends on the selection of H. For a given H, we can construct the same number of nonequivalent parity-check matrices as that of equivalence classes.

D. Design of Three Types of Iterative Decoders

Definition: Let the automorphism group of a code be partitioned based on a given H, a d-order diversity set is a set of d permutations that belong to different equivalence classes.

We choose a d-order diversity set $\{\pi_l : \pi_l \in Aut(C), l = 0,1,...,d-1\}$. Denote by L_y the log-likelihood ratio vector (LLRV) computed from y. Below, they present three types of iterative decoders that use the diversity set in different manners.

- **Decoder-1**: the diversity set is used in a *serial* manner. The decoding procedure is shown in Algorithm 1

Algorithm 1 The Decoding Procedure of Decoder-1

1: **for** $0 \le l \le d-1$ **do**
2: Perform at most I iterations of $\pi_l \mathbf{L_y}$ based on $\mathbf{H}$: if a codeword $\mathbf{c}$ is obtained, stop decoding and output $\pi_l^{-1}\mathbf{c}$.
3: **if** $l = d-1$ and no codeword has been obtained **then**
4: Output the hard-decision of $\mathbf{L_y}$.
5: **end if**
6: **end for**

- **Decoder-2**: the diversity set is used in a *periodic* manner. The decoding procedure is shown in Algorithm 2. In line 5, an inner iteration refers to one time updating of all check nodes and variable nodes of H. For $d = 2$, the decoder works in a Turbo manner [1]. But there are two main differences: 1) The message passing out of a component decoder in the preceding iteration is not subtracted from the a priori information passing to this component decoder in the current iteration; 2) Soft information exchanged between the two component decoders is not limited to information bits.

Algorithm 2 The Decoding Procedure of Decoder-2

1: $\mathbf{s} \leftarrow \mathbf{L_y}$.
2: **for** $0 \le i \le I-1$ **do**
3: **for** $0 \le l \le d-1$ **do**
4: $\mathbf{s} \leftarrow \pi_l \mathbf{s}$.
5: Perform one inner iteration of $\mathbf{s}$ based on $\mathbf{H}$ and update $\mathbf{s}$ as the *a posteriori* LLRV.
6: $\mathbf{s} \leftarrow \pi_l^{-1} \mathbf{s}$.
7: **end for**
8: Use $\mathbf{s}$ to make a hard-decision: if a codeword $\mathbf{c}$ is obtained, stop decoding and output $\mathbf{c}$.
9: **if** $i = I-1$ and no codeword has been obtained **then**
10: Output the hard-decision of $\mathbf{L_y}$.
11: **end if**
12: **end for**

Decoder-3: the diversity set is used in a *parallel* manner. Define $H_l \triangleq \pi_l^{-1} H$. Then by concatenating H_l, we form an augmented parity-check matrix

$$H_{aug} = \begin{bmatrix} H_0 \\ H_1 \\ \vdots \\ H_{d-1} \end{bmatrix}.$$

The decoder performs the SPA with flooding schedule on this highly redundant parity-check matrix, with the maximum number of iterations I.

III. A NEW CONSTRUCTION

We define an $(l\cdot c)\times(l\cdot c)$ binary matrix as

$$B=\begin{bmatrix} A_0 & A_1 & \cdots & A_{c-2} & A_{c-1} \\ A_{c-1}^{(1)} & A_0 & \cdots & A_{c-3} & A_{c-2} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ A_2^{(1)} & A_3^{(1)} & \cdots & A_0 & A_1 \\ A_1^{(1)} & A_2^{(1)} & \cdots & A_{c-1}^{(1)} & A_0 \end{bmatrix},$$

where each submatrix is an $l\times l$ circulant and the zeroth row of $A_i^{(1)}$ is the first row of A_i. Define a permutation π as

$$\pi: j\to(j \bmod l)\cdot c+\lfloor j/l\rfloor,\ j=0,1,...,l\cdot c-1. \tag{1}$$

Theorem 4: If we perform row and column permutations on *B*, both using the permutation π given in (1), then we obtain a circulant matrix.

They summarize the construction procedure as follows.

Step 1: Choose a nonzero codeword from an (*n*, 2) pseudo-cyclic MDS code with $a=\alpha$.

Step 2: Use the codeword and its pseudo-cyclic shifts to construct the base matrix W'.

Step 3: Use matrix dispersion on W' to obtain the QC matrix $H_{disp}(W')$.

Step 4: Apply Theorem 4 to $H_{disp}(W')$ to obtain a circulant as the parity-check matrix *H*.

For **Step 1** and **Step 2**, the form of base matrix W' is given by

$$W'=\begin{bmatrix} w_0 & w_1 & \cdots & w_{n-2} & w_{n-1} \\ \alpha w_{n-1} & w_0 & \cdots & w_{n-3} & w_{n-2} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \alpha w_2 & \alpha w_3 & \cdots & w_0 & w_1 \\ \alpha w_1 & \alpha w_2 & \cdots & \alpha w_{n-1} & w_0 \end{bmatrix}. \tag{2}$$

To construct the base matrix of the form (2), they consider using pseudo-cyclic MDS codes with two information symbols. A pseudo-cyclic code with parameter $a \in GF(q)$ has the property that for any codeword $(c_0, c_1, ..., c_{n-1})$, its pseudo-cyclic shift $(ac_{n-1}, c_0, ..., c_{n-2})$ is also a codeword. If $a = 1$, the pseudo-cyclic code reduces to a cyclic code.

For **Step3**, the Tanner graph corresponding to the matrix $H_{disp}(W)$ has no cycles of length 4 and hence has a girth of at least 6. So the matrix $H_{disp}(W)$ can serve as the parity-check matrix and gives a QC-LDPC code of length $n(q-1)$.

$$H_{disp}(W) = \begin{bmatrix} A(w_{0,0}) & A(w_{0,1}) & \cdots & A(w_{0,n-1}) \\ A(w_{1,0}) & A(w_{1,1}) & \cdots & A(w_{1,n-1}) \\ \vdots & \vdots & \ddots & \vdots \\ A(w_{m-1,0}) & A(w_{m-1,1}) & \cdots & A(w_{m-1,n-1}) \end{bmatrix}.$$

The way to construct the matrix A is given as follow:

Let GF(q) be a finite field with q elements and a be a primitive element of GF(q). Then $\alpha^{-\infty} \triangleq 0, \alpha, \cdots, \alpha^{q-2}$ give all the elements of GF(q). For each non-zero element $\alpha^i, (0 \le i \le q-2)$, define a $(q-1)\times(q-1)$ matrix $A(\alpha^i)$ over GF(2): it is a circulant permutation matrix; the zeroth row is a $(q-1)$-tuple with weight one where the ith component is equal to one and all the other $q-2$ components are equal to zero. The matrix $A(\alpha^i)$ is referred to as the $(q-1)$-fold matrix dispersion of element ai over GF(2). The $(q-1)$-fold matrix dispersion of zero element of GF(q), $A(0)$, is defined as the $(q-1)\times(q-1)$ all-zero matrix.

IV. Simulation Results

They present the simulation results for our constructed cyclic LDPC codes. The BPSK modulated AWGN channel is assumed. In addition to the three decoders presented in Section II, they also simulated a decoder that is not assisted by the automorphism group. The decoder is called Decoder-0, which performs the SPA with flooding schedule on the defining parity-check matrix. For all these decoders, the maximum number of iterations is set to be 100.

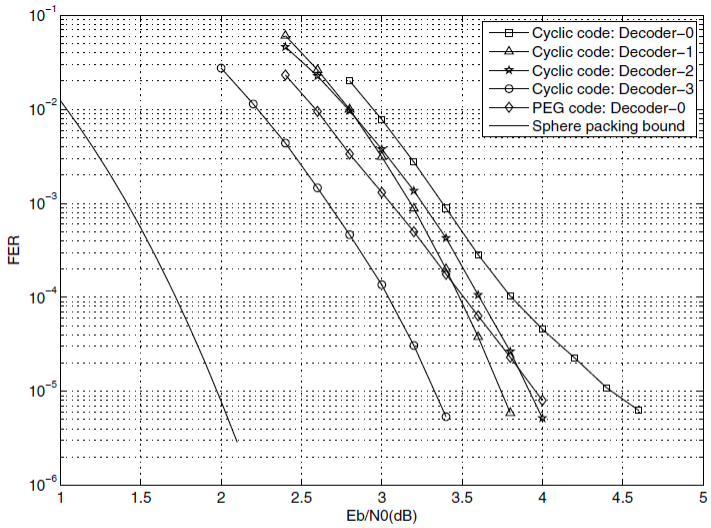

Fig. 2. Performance of the $(341, 160)$ cyclic LDPC code and the $(341, 160)$ PEG-based LDPC code.

Fig. 2 shows the FER performance of the code. The 2-order diversity set $\{\zeta^0, \zeta^1\}$ is used. For comparison, they further simulated a (341,160) LDPC code constructed using the progressive-edge-growth (PEG) algorithm [29]. The parity-check matrix of the code has column weight 3 and row weights 5 and 6. The sphere-packing bound [30] for this length and rate is also included in the figure.

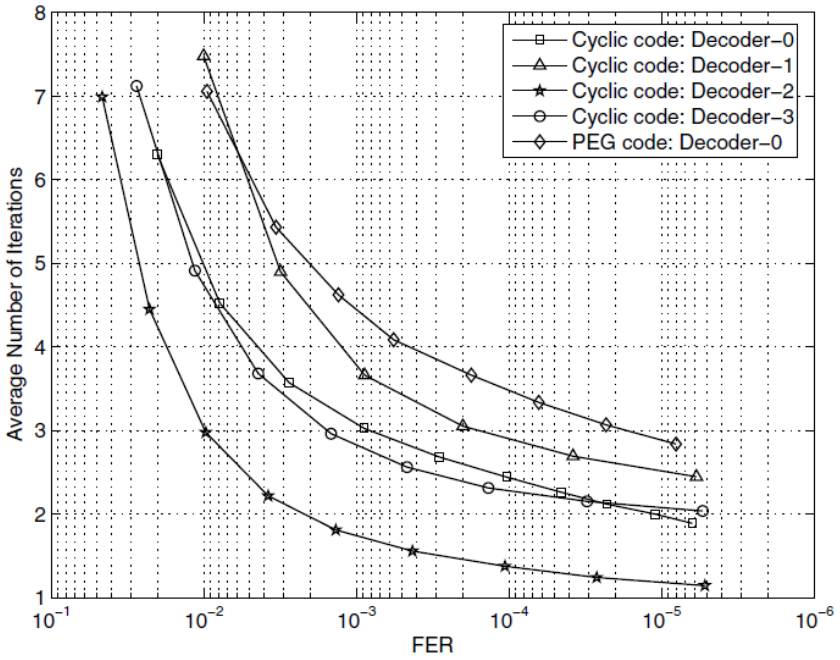

Fig. 3. Average number of iterations for all decoders.

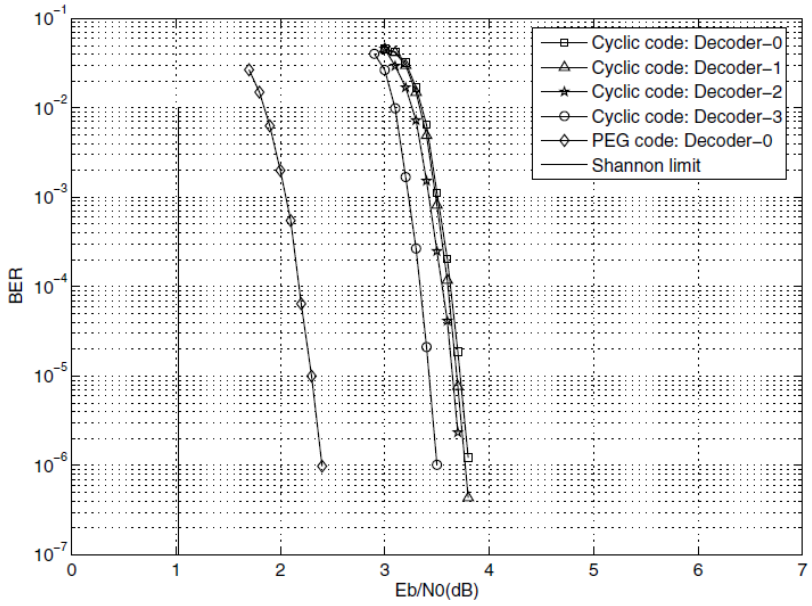

Fig. 4. Performance of the $(5461, 3612)$ cyclic LDPC code and the $(5461, 3612)$ PEG-based code.

- Like HDPC codes, the performance gain for LDPC codes seems more significant for short code lengths. This can be seen by comparing Fig. 2 and Fig. 4 (note that the diversity sets for the two codes have the same order).
- For both HDPC and LDPC with long code lengths, the automorphism group aided iterative decoding does not perform well. In fact, for long HPDC codes, it performs even worse than the hard-decision decoding.
- To obtain a noticeable performance gain, LDPC codes require a smaller diversity set than HDPC codes. This is because that LDPC codes are inherently more suited to iterative decoding than HDPC codes.

Multiuser Detection of Sparsely Spread CDMA

Main ref.: D. Guo, and C.-C. Wnag, "Multiuser detection of sparsely spread CDMA," *IEEE Journal on selected areas in communi.*, Apr, 2008 [1]

Presenter: Jaewook Kang

In Proceeding of CS Journal Club in GIST

Date: July. 2013

I. Introduction

This paper has discussed about design and analysis of multiuser detection (MUD) using sparsely spread CDMA systems. The objective of the MUD problem is how to detect multiple user signals simultaneously at the low computational cost. The main obstacle is multiple-access interference (MAI). These multiple user signals are interference for each user detection one another. The MAI problem arise in most CDMA systems, and optimal detection in such systems requires exponentially growing computation as the number of user increases. This paper investigates a suboptimal MUD detection using sparse CDMA systems. The key idea of the proposed system is to encode the transmitted waveforms using sparsely spread CDMA codes and detect the signal using a linear-complexity belief propagation (BP) algorithm. We summarize the contributions of this work is following:

1) Description of the sparse CDMA system
2) Ensemble of the sparsely spread CDMA codes
3) Design of the BP algorithm for the MUD problem
4) Asymptotic analysis of performance of the BP algorithm based MUD detection

In this report, we aim to sketch the key point of each contribution of this paper.

II. Description of the sparse CDMA system

We consider a fully-synchronous CDMA system which is able to simultaneously transmit K user signals. As shown in Fig.1, symbols X_k from the k-th user is multiplied by the spreading code $\{S_{lk}\}_{l=1}^{L}$ having code length L, being transmitted to the receiver with gain $\frac{A_k}{\sqrt{\Lambda_k}}$ for the transmit power regulation. Then, the receive observes the L channel outputs per a symbol transmission from K users, given by

$$Y_l = \sum_{k=1}^{K} S_{lk} \frac{A_k}{\sqrt{\Lambda_k}} X_k + N_l \quad \text{for } l = 1 \text{ to } L, \tag{1}$$

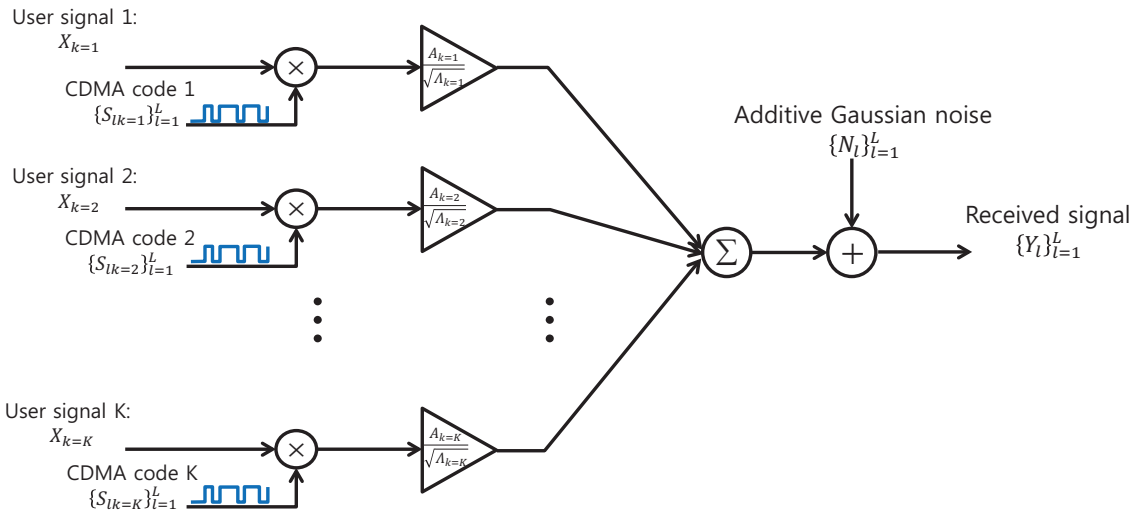

Fig. 1. System model

where we consider additive noise following the zero-mean and unit variance Gaussian distribution, i.e., $N_l \sim \mathcal{N}(0,1)$. In vector form, the expression in (1) can be represented as

$$\underline{Y} = \mathbf{S}\mathbf{A}\underline{X} + \underline{N}, \tag{2}$$

where $\underline{Y} = [Y_1, ..., Y_L] \in \mathbb{R}^L$ denotes the channel output vector, $\underline{X} = [X_1, ..., X_K] \in \mathcal{X}^K \subset \mathbb{R}^K$ is the input symbol vector, $\mathbf{S} \in \mathbb{R}^{L\times K}$ is the sparse spreading matrix, and $\mathbf{A} = diag(\frac{A_1}{\sqrt{\Lambda_1}}, \frac{A_2}{\sqrt{\Lambda_2}}, ..., \frac{A_K}{\sqrt{\Lambda_K}})$ is the gain matrix which has a diagonal form. In the system model, we additionally assume that input symbols X_k, elements of the spreading codes S_{lk} and the transmit gain A_k are i.i.d. drawn from P_X,P_S,P_A respectively. In the receiver side, the goal of the multiuser detector is to estimate the input vector $\underline{X}$ from the channel output vector $\underline{Y}$ given $\mathbf{S}, \mathbf{A}, P_X$.

III. Ensemble of the sparsely spread CDMA codes

Let $\mathbf{H} \in \{0,1\}^{L\times K}$ denote an incidence matrix of the spreading codes $\mathbf{S}$ which indicates nonzero position of the matrix $\mathbf{S}$. The authors also defined that two notation from the incidence matrix, which are

$$\text{The k-th symbol degree: } \Lambda_k = \sum_{l=1}^{L} H_{lk} \tag{3}$$

$$\text{The l-th chip degree: } \Gamma_k = \sum_{k=1}^{K} H_{lk} \tag{4}$$

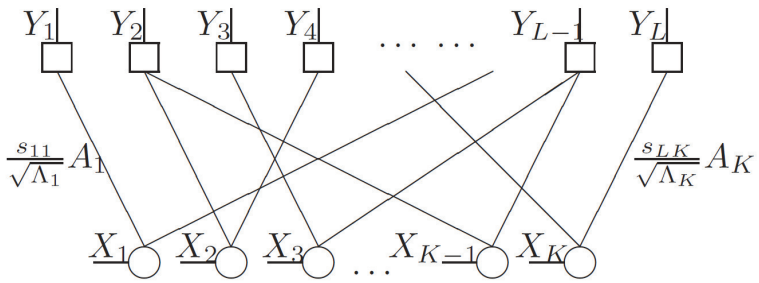

Fig. 2. Factor graphical representation of the CDMA system

Similarly, the average of the symbol and chip degree is defined as $\overline{\Lambda} = \frac{1}{K}\sum_{k=1}^{K}\Lambda_k$ and, $\overline{\Gamma} = \frac{1}{L}\sum_{l=1}^{L}\Gamma_l$, respectively. Then, the factor graph representation of the CDMA system is given in Fig.2. The authors of this paper have tried to analyze the performance of this CDMA system by assuming the following:

1) *Large-system limit*: The system size is very large, i.e., $K, L \to \infty$, and its system load remains a constant, i.e., $\beta \to K/L$.
2) *No-short-cycle*: Under the large-system-limit, the factor graph of the CDMA system does not include cycles shorter than the number of the BP iterations denoted by t.
3) *Chip-semi-regular*: Under the large-system-limit, the chip degree concentrate around their average, i.e., for every l and very small constant $\epsilon > 0$, $\lim_{K,L\to\infty} \Pr\{|\Gamma_l - \mathbf{E}\overline{\Gamma}| > \varepsilon \mathbf{E}\overline{\Gamma}\} = 0$.

Throughout this report, such CDMA system satisfying above assumption is referred to *large-sparse-system* (LSS).

IV. Design of the BP algorithm for the MUD problem

Before discussing the BP detection algorithm for the MUD system, let us summarize the important known facts the BP algorithms

1) BP basically aims to find marginal posterior PDF of each element X_k.
2) In order to reduce the complexity, BP removes the duplicated calculation with message exchanging over the graph connection.
3) *Optimality of BP*: BP provides exact inference (optimal) of the marginal PDFs if the corresponding factor graph is perfectly tree-structured.
4) *Loopy BP*: BP is well applied to graphs with cycles and provides good approximation of the marginal PDFs in practice even through the performance is suboptimal .

For the description of the BP algorithm, we define two notation for the message: *symbol-to-chip* (StC) messages denoted by $V_{k\to l}^{(t)}(x)$ and *chip-to-symbol* (CtS) messages denoted by $U_{l\to k}^{(t)}(x)$ where t is the

number of iterations. In addition, $V_k(x)$ denotes the marginal PDF of X_k. For convenience, we define a set of edge representing statistical connection over the factor graph as $\mathcal{E} := \{(l,k)|S_{lk} \neq 0\}$. Also we define ∂l (resp. ∂k) as the subset of symbols (resp. ships) which have the statistical connection to chip l (resp. symbol k), called its neighborhood. Then, the iterative BP algorithm for computing the marginal PDF of all symbols is shown in Algo.1. This iterative BP algorithm performs exact marginalization of each symbol X_k given the entire observation $\underline{Y}$ if the factor graph is cycle-free. In practical CDMA systems, however, the average node degree is always greater than 2 such that cycles are inevitable. Thus, the BP algorithm performs approximate inference by assuming that all nodes, $\{X_k\}$ and $\{Y_l\}$, are i.i.d. each other.

Algorithm 1 Iterative BP

Inputs: Channel output $\underline{Y}$, Spreading matrix $\mathbf{\Phi}$, Gain matrix $\mathbf{A}$, Prior knowledge $p_X(x)$

Outputs: Marginal PDFs $V_k(x)$ for every k

1)Initialization:

set $U^0_{l\to k}(x) = 1 \ \forall x \in \mathcal{X}$ for every $(l,k) \in \mathcal{E}$

2)Iterations:

for $t = 1$ **to** T **do**

set $V^{(t)}_{k\to l}(x) \propto p_X(x) \times \prod_{j\in\partial k\backslash l} U^{(t-1)}_{j\to k}(x)$ for every $(l,k) \in \mathcal{E}$

set $U^{(t)}_{l\to k}(x) \propto \mathbf{E}\left\{p_{Y_l|\underline{X}}(y|\underline{X})|X_k = x, V^{(t)}_{k\to l}\right\}$

$$:= \sum_{(x_i)_{\partial l\backslash k}} \exp\left[-\frac{1}{2}\left(y_l - \frac{s_{lk}a_k}{\sqrt{\Lambda_k}}x - \sum_{i\in\partial l\backslash k}\frac{s_{li}a_i}{\sqrt{\Lambda_i}}x_i\right)^2\right] \times \prod_{i\in\partial l\backslash k} V^{(t)}_{i\to l}(x_i) \text{ for every } (l,k) \in \mathcal{E}$$

end for

3)Marginal PDFs calculation:

set $V_k(x) \propto p_X(x) \prod_{j\in\partial k} U^{(T)}_{j\to k}(x)$ for every k

The LLR form of BP algorithm is simply obtained by fixing a reference point $x_0 \in \mathcal{X}$ and then defining LLR messages as

$$\text{LLR CtS message: } L^{(t)}_{l\to k}(x) := \log \frac{\mathbf{E}\left\{p_{Y_l|\underline{X}}(y|\underline{X})|X_k = x, L^{(t)}_{k\to l}\right\}}{\mathbf{E}\left\{p_{Y_l|\underline{X}}(y|\underline{X})|X_k = x_0, L^{(t)}_{k\to l}\right\}} \tag{5}$$

$$\text{LLR StC message: } L_{k\to l}^{(t)}(x) := \log p_X(x) + \sum_{j\in\partial k\setminus l} L_{j\to k}^{(t-1)}(x) \tag{6}$$

V. Asymptotic analysis of performance of BP

The key result of this paper states that

The marginal posterior computed for each symbol X_k using BP after t itertions essentially converges to the marginal posterior of a scalar Gaussian channel as the system size increases.

Now, we provide mathematical support for the statement above by stage. Let $P_{X_k}^{bp}(\cdot|\underline{Y}_k^{(t)}, \mathbf{S}, \mathbf{A})$ denote the output CDF from BP, which is approximate posterior of X_k given $\underline{Y}_k^{(t)}$. Here, $\underline{Y}_k^{(t)}$ is all observations within distance $2t-1$ to X_k on the factor graph. If X_k and Y_l is directly connected, the distance will be 1. In addition, let us introduce the canonical scalar Gaussian channel, given as

$$Z = \sqrt{g}X + N, \tag{7}$$

where $X \sim P_X$ and $N \sim \mathcal{N}(0,1)$ are independent, and g denotes the channel gain. For remainder derivation, we use $P_{X|Z;g}(\cdot|z;g)$ to denote the CDF of the posterior distribution of X given Z, according to the Gaussian channel model in (7).

***Theorem** 1 (Gaussian convergence of Marginal posterior):* Given fixed iterations t, the marginal posterior of X_k converges to that of the Gaussian channel, i.e. for every k

$$P_{X_k}^{bp}(x|\underline{Y}_k^{(t)}, \mathbf{S}, \mathbf{A}) \to P_{X|Z;g}(x|h(\underline{Y}_k^{(t)}, \mathbf{S}, \mathbf{A}); \eta^{(t)}A_k^2), \tag{8}$$

in probability under the LSS setup, where the Gaussian channel output is given as $Z = h(\underline{Y}_k^{(t)}, \mathbf{S}, \mathbf{A}) \sim \mathcal{N}(\sqrt{\eta^{(t)}}ax, 1)$, the channel gain $A_k\sqrt{\eta^{(t)}}$ is determined by the following recursion:

$$\frac{1}{\eta^{(t)}} = 1 + \beta\overline{\text{var}}\left\{AX|\sqrt{\eta^{(t-1)}}AX + N\right\}, \tag{9}$$

and

$$\overline{\text{var}}\left\{U|V\right\} := \mathbf{E}\left\{\left(U - \mathbf{E}\left\{U|V\right\}\right)^2\right\}. \tag{10}$$

Proof: The authors proved Theorem 1 by considering messages of the LLR form given in (5) and (6). Proving Theorem 1 is equivalent to showing the LLR StC message is Gaussian distributed with $X_k \sim \mathcal{N}(A_k^2\eta^{(t)}X_k, 1)$. We summarize this proof in four steps.

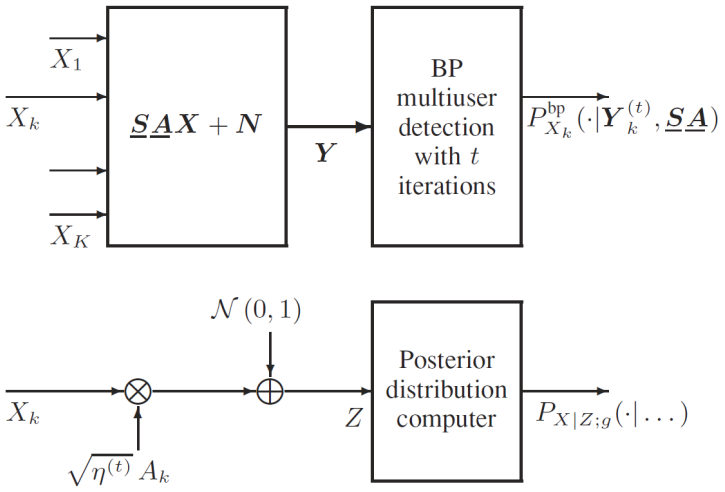

Fig. 3. Upper diagram: Multiuser channel and BP detection. Lower diagram: The asymptotically equivalent scalar Gaussian channel

Step I: The StC message is Gaussian RV by the central limit theorem (CLT): Under the no-short-cycle assumption, all CtS messages $L_{l\to k}^{(t)}$ are i.i.d. conditioned on $X_k = x_k$. From (6), by CLT, the message is a Gaussian random vector.

Step II: LLR obtained from a scalar Gaussian channel is also Gaussian distributed. Namely, for $Y = \sqrt{\gamma}X + N$, its LLR is a Gaussian RV, i.e.,

$$\log \frac{p_{Y|X}(Y|x_1)}{p_{Y|X}(Y|x_0)} = \sqrt{\gamma}(x_1 - x_0)Y - \gamma(x_1^2 - x_0^2)/2 \tag{11}$$

Step III: Calculation of mean and covariance of the StC messages given as

$$\mathbf{E}[L_{k\to l}^{(t)}(x)] = \log p_X(x) + \sum_{j\in\partial k\setminus l} \mathbf{E}[L_{j\to k}^{(t-1)}(x)]. \tag{12}$$

We first consider the mean of the CtS messages. To this end, we have

$$\begin{aligned} f(y,x) &:= \mathbf{E}\left\{p_{Y_l|\underline{X}}(y|\underline{X})|X_k = x, L_{k\to l}^{(t)}\right\} \\ &= \sum_{(x_i)_{\partial l\setminus k}} \left(\frac{1}{\sqrt{2\pi}} \exp\left[-\frac{1}{2}(y - \sum_{i\in\partial l\setminus k} \frac{s_{li}A_i}{\sqrt{\Lambda_i}} x_i - c_k x)^2 + \underbrace{\sum_{i\in\partial l\setminus k} L_{i\to l}^{(t)}(x_i)}_{=B_{x_k}} \right] \right), \end{aligned} \tag{13}$$

where we use $c_k = \frac{s_{lk}a_k}{\sqrt{\Lambda_k}}$. Then, we apply the 2nd order Taylor approximation with respect to $x = 0$ as

$$\begin{aligned} f(y,x) &\approx f(y,x=0) + f\prime(y,x=0)x + \frac{1}{2}f\prime\prime(y,x=0)x^2 \\ &= g_0(y) + g_1(y)c_k x + \frac{1}{2}g_2(y)c_k^2 x^2, \end{aligned} \tag{14}$$

where we define

$$g_0(y) := \frac{1}{\sqrt{2\pi}} \sum_{(x_i)_{\partial l \setminus k}} \exp\left[-\frac{1}{2}(y - \sum_{i \in \partial l \setminus k} \frac{s_{li}A_i}{\sqrt{\Lambda_i}} x_i)^2 + B_{x_k} \right] \tag{15}$$

$$g_1(y) := \frac{1}{\sqrt{2\pi}} \sum_{(x_i)_{\partial l \setminus k}} (y - \sum_{i \in \partial l \setminus k} \frac{s_{li}A_i}{\sqrt{\Lambda_i}} x_i) \exp\left[-\frac{1}{2}(y - \sum_{i \in \partial l \setminus k} \frac{s_{li}A_i}{\sqrt{\Lambda_i}} x_i)^2 + B_{x_k} \right] \tag{16}$$

$$g_2(y) := \frac{1}{\sqrt{2\pi}} \sum_{(x_i)_{\partial l \setminus k}} ((y - \sum_{i \in \partial l \setminus k} \frac{s_{li}A_i}{\sqrt{\Lambda_i}} x_i)^2 - 1) \exp\left[-\frac{1}{2}(y - \sum_{i \in \partial l \setminus k} \frac{s_{li}A_i}{\sqrt{\Lambda_i}} x_i)^2 + B_{x_k} \right]. \tag{17}$$

Then, from (5), the CtS LLR message is given as

$$\begin{aligned} L_{l\to k}^{(t)} &= \log \frac{g_0(y) + g_1(y)c_k x + \frac{1}{2}g_2(y)c_k^2 x^2}{g_0(y) + g_1(y)c_k x_0 + \frac{1}{2}g_2(y)c_k^2 x_0^2} \\ &\approx \frac{g_1(y)}{g_0(y)} c_k (x - x_0) + \frac{g_2(y)}{g_0(y)} c_k^2 (x^2 - x_0^2) - \frac{1}{2}\frac{g_1^2(y)}{g_0^2(y)} c_k^2 (x^2 - x_0^2), \end{aligned} \tag{18}$$

where we further apply the 2nd order Taylor approximation of $\log(x)$. The mean of the CtS message $\mathbf{E}[L_{j\to k}^{(t)}(x)]$ can be obtained by taking integration to (18) with respect to y. Then, using (12), the mean of the LLR StC message is obtained as

$$\mathbf{E}\left[L_{k\to l}^{(t)}(x) \right] = \Theta(x_k(x - x_0) - (x^2 - x_0^2)/2) \tag{19}$$

where

$$\Theta = A_k^2 \int \frac{g_1^2(y)}{g_0(y)} dy \frac{\sum_{j \in \partial k \setminus l} S_{jk}^2}{\Lambda_k}. \tag{20}$$

In (20), by law of large number, $\frac{\sum_{j \in \partial k \setminus l} S_{jk}^2}{\Lambda_k} \to 1$. Here, importantly note that the result in (19) is exactly equivalent to mean of LLR in a scalar Gaussian channel

$$\begin{aligned} X_k &= \sqrt{\Theta} X + N \\ &= A_k \sqrt{\int \frac{g_1^2(y)}{g_0(y)} dy} X + N \end{aligned} \tag{21}$$

where X_k here is a symbol obtained from the BP iteration given the observation $\underline{Y}$ (which is equivalent to Z in Th.1) and $N \sim \mathcal{N}(0,1)$ is additive noise with unit variance. Although the derivation of covariance is omitted here, they are also equivalent.

Now, we summarize the proof as following

1) LLR of StC message is asymptotically a Gaussian RV by CLT from Step I.
2) LLR of a scalar Gaussian channel is a Gaussian RV from Step II.
3) The mean and covariance of LLR StC message have the exactly same form as the LLR of the scalar Gaussian channel in (21) from Step III.
4) From (12), each individual symbol X_k via BP is also Gaussian distributed with $N \sim \mathcal{N}(\sqrt{\Theta}, 1)$.

The last piece of the proof of Theorem 1 is to quantify the corresponding SNR Θ with respect to the number of BP iterations t by showing that

$$\lim_{\substack{L,K\to\infty \\ \overline{\Gamma}\to\infty}} \int \frac{g_1^2(y)}{g_0(y)} dy = \eta^{(t)}. \tag{22}$$

But, the proof of (22) was not well explained in the paper. One thing is that one can derive the recursion in (9) by showing (22).

REFERENCES

[1] D. Guo and C. C. Wang, "Multiuser detection of sparsely spread CDMA," *IEEE J. Sel. Areas Comm.*, vol. 26, no. 3, pp. 421-431, Mar. 2008.