LETTER

A Fast Iterative Check Polytope Projection Algorithm for ADMM Decoding of LDPC Codes by Bisection Method

Yan LIN[†], ***Member***, **Qiaoqiao XIA**[†a)], **Wenwu HE**[††], ***and*** **Qinglin ZHANG**[†], ***Nonmembers***

SUMMARY Using linear programming (LP) decoding based on alternating direction method of multipliers (ADMM) for low-density parity-check (LDPC) codes shows lower complexity than the original LP decoding. However, the development of the ADMM-LP decoding algorithm could still be limited by the computational complexity of Euclidean projections onto parity check polytope. In this paper, we proposed a bisection method iterative algorithm (BMIA) for projection onto parity check polytope avoiding sorting operation and the complexity is linear. In addition, the convergence of the proposed algorithm is more than three times as fast as the existing algorithm, which can even be 10 times in the case of high input dimension.

key words: *alternating direction method of multipliers, low-density parity-check codes, check polytope projection, bisection method iterative algorithm*

1. Introduction

In recent years, Barman proposed the linear programming (LP) decoding algorithm based on alternating direction method of multipliers (ADMM) in [1], combining the dual decomposition theory in optimization algorithm and the augmented Lagrange method with constraint optimization as well as LP model to solve LP problem. Although the complexity of the ADMM decoding algorithm is lower than the original LP decoding algorithm, its complexity can be further reduced.

In fact, the projection on parity check polytope is the most computationally intensive part of the whole decoding process. To deal with the problem of high computational complexity of projection, Wei et al. proposed the reduction of useless projection in [2]. Besides, Jiao et al. suggested increasing the projection efficiency using the look-up table to reduce the projection operation [3]. Except to reducing the complexity by decreasing the number of projection, the computational complexity of projection can also be reduced by simplifying the projection algorithm. X. Zhang et al. put forward a novel and efficient projection algorithm based on Cut Search Algorithm (CSA) [4]. Afterwards, G. Zhang proposed a projection method that transformed the projection onto the check polytope to a projection onto probability simplex [5].

However, these algorithms inevitably involve sorting operations, which consume a large amount of computing resources and memory resources. In addition, iterative check polytope projection algorithm proposed by Wei et al. in [6], removed sorting operations to simplify the projection. Nevertheless, the iterative check polytope projection algorithm required a large number of iterations to obtain a valid estimated projection when the dimension of input vector is high. Therefore the current work proposes the bisection method iterative algorithm (BMIA). Compared with algorithm of [6], the BMIA can speed up the projection onto the parity check polytope without increasing computational complexity. According to simulation results, only 1/10 number of iterations for iterative algorithm is required for the proposed algorithm when the dimension of input vector is high.

2. Preliminaries

Consider an LDPC code C of length n and let $m \times n$ parity-check matrix $\mathbf{H}$ correspond to C. Each row of $\mathbf{H}$ corresponds to one check node and each column of $\mathbf{H}$ corresponds to one variable node. Define $\mathcal{I} = \{1, 2, 3, ..., n\}$ and $\mathcal{J} = \{1, 2, 3, ..., m\}$ as the index set of all variable nodes and check nodes, respectively. Let $N_v(i)$ denote the set of check nodes adjacent to variable node vn_i, $i \in \mathcal{I}$. d_i is defined as the degree of variable node vn_i. Similarly, $N_c(j)$ denotes the set of variable nodes adjacent to check node cn_j, $j \in \mathcal{J}$ and d_j is defined as the degree of check node cn_j, i.e., $d_i = |N_v(i)|$ and $d_j = |N_c(j)|$.

Let $\mathbf{x}$ be the transmitted codeword of length n, and $\mathbf{y}$ is the received vector resulted by $\mathbf{x}$ transmitted across a memoryless binary input output-symmetric channel. In [1], the ADMM-LP model was proposed as following formulations to handle the LP decoding problem:

$$\begin{aligned} &\min\ \boldsymbol{\gamma}^T\mathbf{x} \\ &s.t.\ \ \mathbf{T}_j\mathbf{x} = \mathbf{z}_j\ \ \mathbf{z}_j \in \mathcal{P}_{d_j}, \forall j \in \mathcal{J},\ \ \mathbf{x} \in [0,1]^n \end{aligned} \tag{1}$$

where $\boldsymbol{\gamma} \in \mathbb{R}^n$ is the vector of log-likelihood rations (LLRs), and the i-th entry of $\boldsymbol{\gamma}$ can be defined as: $\gamma_i = \log\left(\frac{P_r(y_i/x_i=0)}{P_r(y_i/x_i=1)}\right)$. $\mathbf{T}_j$ is the $d_j \times n$ transfer matrix which selects the d_j components of $\mathbf{x}$ involved in the j-th check node. $\mathbf{z}_j$ is the auxiliary variables of check node cn_j. $\mathcal{P}_{d_j}$ is the parity check polytope, implying the convex hull of all permutations of a length-d_j binary vector with even number of ones.

The augmented Lagrangian function corresponding to formula (1) is presented as follows:

$$L_\mu(\mathbf{x},\mathbf{z},\boldsymbol{\lambda}) = \boldsymbol{\gamma}^T\mathbf{x} + \sum_{j\in\mathcal{J}} \boldsymbol{\lambda}_j^T\left(\mathbf{T}_j\mathbf{x} - \mathbf{z}_j\right) + \frac{\mu}{2}\sum_{j\in\mathcal{J}} \|\mathbf{T}_j\mathbf{x} - \mathbf{z}_j\|_2^2 \tag{2}$$

Manuscript received April 10, 2019.

[†]The authors are with College of Physical Science and Technology, Central China Normal University, Wuhan, 430079, China.

[††]The author is with School of Electronic Information, Wuhan University, Wuhan, 430072, China.

a) E-mail: xiaqq@mail.ccnu.edu.cn (Corresponding author)

DOI: 10.1587/transfun.E102.A.1406

where $\boldsymbol{\lambda}_j \in \mathbb{R}^{d_j} (j \in \mathcal{J})$ is the scaled dual variable, and $\mu > 0$ is penalty parameter which is a constant value.

The ADMM is consistent with the updating rules as follows to handle the optimization problem of (2).

$$x_i^{k+1} = \prod_{[0,1]} \left(\frac{1}{d_i} \left(\sum_{j \in N_v(i)} \left(z_j^k - \frac{1}{\mu} \lambda_j^k \right) - \gamma_i / \mu \right) \right) \quad (3)$$

$$\mathbf{z}_j{}^{k+1} = \prod_{\mathcal{P}_{d_j}} \left(\mathbf{T}_j \mathbf{x}^{k+1} + \boldsymbol{\lambda}_j^k / \mu \right) \quad (4)$$

$$\boldsymbol{\lambda}_j{}^{k+1} = \boldsymbol{\lambda}_j^k + \mu (\mathbf{T}_j \mathbf{x}^{k+1} - \mathbf{z}_j^{k+1}) \quad (5)$$

where $(k+1)$ denotes the $(k+1)$-th iterations of ADMM decoding, and $\prod_{\mathcal{P}_{d_j}}$ is the Euclidean projection of vector onto $\mathcal{P}_{d_j}$.

In [7], Debbabi indicated that the projection onto parity check polytope is the most time-consuming part based on the process of ADMM decoding. As a result, how to simplify the projection operation has become the focus of researches.

2.1 Parity Check Polytope Projection

On the basis of [4], the parity check polytope $\mathcal{P}_{d_j}$ must satisfy following constraints:

$$0 \le u_i \le 1 \quad i \in \left[d_j \right] \quad (6)$$

$$\sum_{i \in \mathcal{V}} u_i - \sum_{i \in \mathcal{V}^c} u_i \le |\mathcal{V}| - 1, \forall \mathcal{V} \subseteq [d_j], |\mathcal{V}| \text{ is odd} \quad (7)$$

where $\left[d_j \right]$ denotes the set of $\left\{ 1, 2, ..., d_j \right\}$. $\mathcal{V}^c = \left[d_j \right] \backslash \mathcal{V}$ is the complement of $\mathcal{V}$.

Regarding the given set $\mathcal{V}$, define its indicator vector $\boldsymbol{\theta}_{\mathcal{V}}$ as follows:

$$\theta_{\mathcal{V},i} = \begin{cases} 1, & \text{if } i \in \mathcal{V} \\ -1, & \text{if } i \in \mathcal{V}^c \end{cases} \quad (8)$$

According to (6), the parity check polytope $\mathcal{P}_{d_j}$ lies inside the $[0,1]^{d_j}$ unit hypercube. Consequently, for a vector $\mathbf{v} \in \mathbb{R}^{d_j}$, let $\mathbf{u} = \prod_{[0,1]^{d_j}} \mathbf{v}$, where $\prod_{[0,1]^{d_j}} (\cdot)$ denotes the projection onto unit hypercube (i.e., if $v_i > 1$, then $u_i = 1$, if $v_i < 0$, then $u_i = 0$, else $u_i = v_i$). If $\mathbf{u}$ satisfies the inequality $\boldsymbol{\theta}_{\mathcal{V}}^T \mathbf{u} \le |\mathcal{V}| - 1$, it can be determined that $\mathbf{u}$ lies inside parity check polytope, and $\mathbf{u}$ is the projection of $\mathbf{v}$ onto $\mathcal{P}_{d_j}$.

Considering this case, $\mathbf{u}$ does not lie inside parity check polytope $\mathcal{P}_{d_j}$. Based on CSA [4], it indicates that there exists the unique inequality $\boldsymbol{\theta}_{\mathcal{V}}^T \mathbf{u} > |\mathcal{V}| - 1$, we call this inequality, the cut at $\mathbf{u}$. The method to find $\mathcal{V}$ is described as follows: firstly, initial $\mathcal{V}$. For $\forall u_i > 0.5$, then $i \in \mathcal{V}$, and for $\forall u_i \le 0.5$, then $i \in \mathcal{V}^c$. Subsequently, we calculate the number of elements in set $\mathcal{V}$, which is denoted as $|\mathcal{V}|$. If $|\mathcal{V}|$ is odd, we flip the index i of u_i which is closest to 0.5, i.e., if $i \in \mathcal{V}$, change i to $i \in \mathcal{V}^c$, if $i \in \mathcal{V}^c$, change i to $i \in \mathcal{V}$, then update $|\mathcal{V}|$. Furthermore, in case of $\mathbf{u} \notin \mathcal{P}_{d_j}$, the CSA suggests that the projection of $\mathbf{v}$ onto $\mathcal{P}_{d_j}$ must be on the facet $\boldsymbol{\theta}_{\mathcal{V}}^T \mathbf{w} = |\mathcal{V}| - 1$. In other words, the projection $\mathbf{z} = \prod_{\mathcal{P}_{d_j}} \mathbf{v}$ satisfies $\boldsymbol{\theta}_{\mathcal{V}}^T \mathbf{z} = |\mathcal{V}| - 1$. According to [4], the point $\mathbf{z} = \prod_{\mathcal{P}_{d_j}} \mathbf{v}$ equal to $\mathbf{z} = \prod_{[0,1]^d} (\mathbf{v} - s^* \boldsymbol{\theta}_{\mathcal{V}})$, where $s^* \ge 0$ is a scalar, called the difference coefficient. Intuitively, if the difference coefficient s^* is determined, the projection of $\mathbf{v}$ onto $\mathcal{P}_{d_j}$ can also be found.

3. Bisection Method Iterative Parity Check Polytope Projection

There exists some algorithms to get the difference coefficient s^* through sorting operation or iterating operation. For the algorithms involving sorting operation, the complexity is high. For the iterative algorithm, the complexity is $O(d_j)$, but the number of iterations is huge when the dimension of input vector is high. Therefore, we proposed the bisection method iterative parity check polytope projection algorithm.

Due to the point $\mathbf{z} = \prod_{[0,1]^{d_j}} (\mathbf{v} - s^* \boldsymbol{\theta}_{\mathcal{V}})$ satisfies $\boldsymbol{\theta}_{\mathcal{V}}^T \mathbf{z} = |\mathcal{V}| - 1$, the following can be concluded:

$$\boldsymbol{\theta}_{\mathcal{V}}^T \prod_{[0,1]^{d_j}} (\mathbf{v} - s^* \boldsymbol{\theta}_{\mathcal{V}}) = |\mathcal{V}| - 1$$

Define the function $f(\beta)$ as follows:

$$f(\beta) = \boldsymbol{\theta}_{\mathcal{V}}^T \prod_{[0,1]^{d_j}} (\mathbf{v} - \beta \boldsymbol{\theta}_{\mathcal{V}}) \quad (9)$$

where $\mathbf{v} \in \mathbb{R}^{d_j}$ is the input vector, β is a scalar and when $\beta = s^*$, it satisfies $f(s^*) = \boldsymbol{\theta}_{\mathcal{V}}^T \prod_{[0,1]^{d_j}} (\mathbf{v} - s^* \boldsymbol{\theta}_{\mathcal{V}}) = |\mathcal{V}| - 1$.

Proposition 1: Given vector $\mathbf{v} \in \mathbb{R}^{d_j}$, $\boldsymbol{\theta}_{\mathcal{V}}$ is the indicator vector of cutting set $\mathcal{V}$. For β, $f(\beta) = \boldsymbol{\theta}_{\mathcal{V}}^T \prod_{[0,1]^{d_j}} (\mathbf{v} - \beta \boldsymbol{\theta}_{\mathcal{V}})$ is a decreasing function.

proof: Define $\mathbf{v}_\beta = \prod_{[0,1]^{d_j}} (\mathbf{v} - \beta \boldsymbol{\theta}_{\mathcal{V}})$, $f(\beta)$ is expressed as formula (9). Taking the directional derivative with respect to β increasing, it's shown as follows:

$$\frac{\partial f(\beta)}{\partial \beta} = \boldsymbol{\theta}_{\mathcal{V}}^T \frac{\partial \mathbf{v}_\beta}{\partial \beta} = - \sum_{0 < v_{\beta,i} < 1, 1 \le i \le d_j} \theta_{\mathcal{V},i}^2 < 0 \quad (10)$$

Therefore $f(\beta)$ is a decreasing function corresponding to β.

Proposition 2: Given a vector $\mathbf{v} \in \mathbb{R}^{d_j}$, let $\mathbf{u} = \prod_{[0,1]^{d_j}} \mathbf{v} \notin \mathcal{P}_{d_j}$, there exists cut $\boldsymbol{\theta}_{\mathcal{V}}^T \mathbf{u} > |\mathcal{V}| - 1$, then the different coefficient s^*, corresponding the equality $\boldsymbol{\theta}_{\mathcal{V}}^T \prod_{[0,1]^{d_j}} (\mathbf{v} - s^* \boldsymbol{\theta}_{\mathcal{V}}) = |\mathcal{V}| - 1$, must satisfy $s^* \in [0, \frac{1}{2}(\min_{i \in \mathcal{V}} v_i - \max_{j \in \mathcal{V}^c} v_j)]$.

Proof: Define vector $\mathbf{z} = \prod_{\mathcal{P}_{d_j}} \mathbf{v}$, $\mathbf{u}' = \prod_{[0,1]^{d_j}} (\mathbf{v} - s^* \boldsymbol{\theta}_{\mathcal{V}})$. We have $\mathbf{z} = \mathbf{u}'$. Assuming that $s^* < 0$, the projection of $\mathbf{v}$ onto unit hypercube satisfies $\prod_{[0,1]^{d_j}} \mathbf{v} \notin \mathcal{P}_{d_j}$ and $\boldsymbol{\theta}_{\mathcal{V}}^T \prod_{[0,1]^{d_j}} \mathbf{v} > |\mathcal{V}| - 1$.

Due to condition $s^* < 0$, we have, for $\forall i \in \mathcal{V}$, $u_i' = \prod_{[0,1]} (v_i + |s^*|) \ge \prod_{[0,1]} v_i$ and for $\forall j \in \mathcal{V}^c$, $u_j' = \prod_{[0,1]} (v_j - |s^*|) \le \prod_{[0,1]} v_j$.

Therefore, we can get inequality as follows:

$$\begin{aligned} \boldsymbol{\theta}_{\mathcal{V}}^T \mathbf{u}' &= \sum_{i \in \mathcal{V}} \prod_{[0,1]} (v_i + |s^*|) - \sum_{j \in \mathcal{V}^c} \prod_{[0,1]} (v_j - |s^*|) \\ &\ge \boldsymbol{\theta}_{\mathcal{V}}^T \prod_{[0,1]^{d_j}} \mathbf{v} \\ &> |\mathcal{V}| - 1 \end{aligned}$$

Obviously, it breaks condition that $f(s^*) = \boldsymbol{\theta}_{\mathcal{V}}^T \prod_{[0,1]^{d_j}} (\mathbf{v} - s^*\boldsymbol{\theta}_{\mathcal{V}}) = |\mathcal{V}| - 1$. Hence, s^* should satisfy that $s^* \geq 0$.

Previously, we have described that the projection $\mathbf{z} = \prod_{\mathcal{P}_{d_j}} \mathbf{v}$ equal to $\mathbf{z} = \prod_{[0,1]^{d_j}} (\mathbf{v} - s^*\boldsymbol{\theta}_{\mathcal{V}})$. From the CSA referred in [4], it can be extended in the conclusion: for all $i \in \mathcal{V}$, $j \in \mathcal{V}^c$, we have $z_i \geq z_j$, in other words, $\min_{i\in\mathcal{V}} z_i \geq \max_{j\in\mathcal{V}^c} z_j$.

Focusing on [8], Proposition 3, since the component-wise ordering of $\prod_{\mathcal{P}_{d_j}} (\mathbf{v} - s^*\boldsymbol{\theta}_{\mathcal{V}})$ is same as that of $(\mathbf{v} - s^*\boldsymbol{\theta}_{\mathcal{V}})$, and we have $\mathbf{z} = \prod_{[0,1]^{d_j}} (\mathbf{v} - s^*\boldsymbol{\theta}_{\mathcal{V}}) \in \mathcal{P}_{d_j}$, that is to say, $\mathbf{z} = \prod_{\mathcal{P}_{d_j}} (\mathbf{v} - s^*\boldsymbol{\theta}_{\mathcal{V}})$. Therefore the component-wise ordering of $\mathbf{z}$ is the same as that of $(\mathbf{v} - s^*\boldsymbol{\theta}_{\mathcal{V}})$.

According to [9], Proposition 6.4, we have that, for $\forall i \in \mathcal{V}$, $z_i \leq v_i - s^*\theta_{\mathcal{V},i}$, and for $\forall j \in \mathcal{V}^c$, $z_j \geq v_j - s^*\theta_{\mathcal{V},j}$. Based on the combination of $\min_{i\in\mathcal{V}} z_i \geq \max_{j\in\mathcal{V}^c} z_j$ for $i \in \mathcal{V}$, $j \in \mathcal{V}^c$, we have $\min_{i\in\mathcal{V}} v_i - s^* \geq \max_{j\in\mathcal{V}^c} v_j + s^*$ (because, for $i \in \mathcal{V}$, $\theta_{\mathcal{V},i} = 1$, for $j \in \mathcal{V}^c$, $\theta_{\mathcal{V},j} = -1$). Therefore, it can be concluded as $s^* \leq \frac{1}{2}(\min_{i\in\mathcal{V}} v_i - \max_{j\in\mathcal{V}^c} v_j)$. The conclusion in Proposition 2 is proved.

Based on the above analysis, it is proper to briefly summarize the BMIA to perform the projection $\mathbf{z} = \prod_{\mathcal{P}_{d_j}} \mathbf{v}$. There are mainly two steps of the projection.

Firstly, we check the projection of $\mathbf{v}$ onto unit hypercube, i.e., $\mathbf{u} = \prod_{[0,1]^{d_j}} \mathbf{v}$, whether or not it is in parity check polytope $\mathcal{P}_{d_j}$ by CSA. If the projection $\mathbf{u}$ satisfies $\boldsymbol{\theta}_{\mathcal{V}}^T\mathbf{u} \leq |\mathcal{V}| - 1$ and then we have $\mathbf{u}$ is the projection of $\mathbf{v}$ onto parity check polytope. If not, we can determine that the projection $z = \prod_{\mathbb{PP}_{d_j}} \mathbf{v}$ lies on the facet $\boldsymbol{\theta}_{\mathcal{V}}^T\mathbf{w} = |\mathcal{V}| - 1$ by CSA.

For the case that $\boldsymbol{\theta}_{\mathcal{V}}^T\mathbf{u} > |\mathcal{V}| - 1$, we perform the BMIA described in Algorithm 1 to find the projection $\mathbf{z}$. Based on the condition that we have determined the indicator vector $\boldsymbol{\theta}_{\mathcal{V}}$ corresponding to facet $\boldsymbol{\theta}_{\mathcal{V}}^T\mathbf{w} = |\mathcal{V}| - 1$, we focus on the equality $f(\beta) = \boldsymbol{\theta}_{\mathcal{V}}^T \prod_{[0,1]^{d_j}} (\mathbf{v} - \beta\boldsymbol{\theta}_{\mathcal{V}})$. The BMIA aims to find a estimated value β of s^* by iterative operation. Proposition 1 and Proposition 2 provide guarantee for β to converge to s^* using bisection method.

As described in Algorithm 1, threshold ϵ (or I_{max}) is the iterative stop condition. Algorithm 1 shows that the ϵ is the length of the current interval and the I_{max} is the maximum number of iterations. Then we need to calculate the upper limit β_{max} of the interval in which the estimated value β located. The next work mainly includes two steps at each iteration, calculating estimated value β and updating the interval. In terms of β, it is the midpoint of the interval. For updating the interval, we perform the method as Algorithm 1, line 7, 8, 9 and 10 to select the subinterval.

Algorithm 1 Bisection Method Iterative Check Polytope Projection Algorithm

Input: Vector $\mathbf{v} \in \mathbb{R}^{d_j}$, indicator vector $\boldsymbol{\theta}_{\mathcal{V}}$ corresponding to the facet $\boldsymbol{\theta}_{\mathcal{V}}^T\mathbf{x} = |\mathcal{V}| - 1$, threshold ϵ (or the max iterations $I_{\max}$)
Output: Projection $\mathbf{z} = \prod_{[0,1]^{d_j}} (\mathbf{v} - \beta\boldsymbol{\theta}_{\mathcal{V}})$

1: $\beta_{max} \leftarrow \frac{1}{2}(\min_{\theta_{\mathcal{V},i}=1} v_i - \max_{\theta_{\mathcal{V},j}=-1} v_j)$ $p \leftarrow |\{i|\theta_{\mathcal{V},i} = 1\}| - 1$
2: initial $\beta_{low} \leftarrow 0$, $\beta_{up} \leftarrow \beta_{max}$, $iter \leftarrow 0$
3: **repeat**
4: $\quad\beta \leftarrow \frac{1}{2}(\beta_{up} + \beta_{low})$
5: $\quad\mathbf{z} \leftarrow \prod_{[0,1]^{d_j}} (\mathbf{v} - \beta\boldsymbol{\theta}_{\mathcal{V}})$
6: $\quad iter \leftarrow iter + 1$
7: $\quad$**if** $\boldsymbol{\theta}_{\mathcal{V}}^T\mathbf{z} < p$ **then**
8: $\quad\quad\beta_{up} \leftarrow \beta$
9: $\quad$**else**
10: $\quad\quad\beta_{low} \leftarrow \beta$
11: $\quad$**end if**
12: **until** $|\beta_{up} - \beta_{low}| < \epsilon$ (or $iter > I_{\max}$)
13: **return** $\mathbf{z}$

4. Simulation Result

Consider sending codeword through additive white Gaussian noise (AWGN) channel with binary phase shift keying (BPSK) modulation. In the simulation experiment, the ADMM-LP decoder with penalty function l_2 [10] is employed. Based on [10], we can obtain a set of parameters (include the penalty parameter μ and the parameter α) for a particular code that achieves the lowest FER by exhaustive search over all possible parameters. In this paper, we mainly consider the influence of different check polytope projection algorithms on the performance of ADMM-LP decoder. Therefore, we select a same set parameters that achieves low FER for simulation codes. The penalty parameter μ is set to 2.2 and the parameter α is set to 3.0. Besides, the maximum number of iterations in ADMM decoding algorithm is set to 300.

To illustrate the performance of BMIA in ADMM-LP decoder, we use the irregular (576, 288) rate 1/2 code C_1 and (576, 192) rate 2/3 code C_2 as well as regular (576, 96) rate 5/6 C_3 from IEEE 802.16e. The check degree of C_1, C_2 and C_3 are $\{6, 7\}$, $\{10, 11\}$ and 20, respectively. In simulation, the frame error rate (FER) curve of the decoder with CSA is used as reference curve to determine whether the FER performance of decoder with BMIA (or Iterative check polytope projection algorithm) is optimal. If the decoder with BMIA (or Iterative check polytope projection algorithm) has a performance practically identical to that of CSA, the FER performance of decoder with BMIA (or Iterative check polytope projection algorithm) is optimal.

Figure 1 shows the effects of I_{max} and ϵ on the FER curves of the BMIA, demonstrating that both the I_{max} and ϵ will affect the performance of the FER curve. We note that the FER curve tends to that of CSA gradually with the increasing of I_{max} ($\epsilon = 0$) (or with the decreasing of ϵ ($I_{max} = \infty$)), especially when the ϵ is 0.01, the FER performance is optimal. Afterwards, we focus on comparing the performance of BMIA with the existing iterative projection algorithm (mainly the algorithm referred in [6]).

Table 1 shows the comparison of the average projection time of 30 iterations between BMIA and Iterative projection algorithm at signal-to-noise rate (SNR)=2.6, 3.2, 4.2 dB for

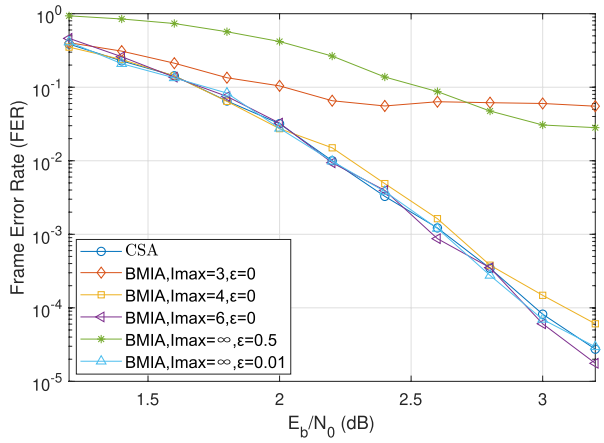

Fig. 1 FER performance of C_1 for ADMM decoder with BMIA projection algorithm for different ϵ and I_{max}.

Table 1 Avg. projection time.

Codeword	Parity Check Polytope Projection Algorithm	
	Iterative	BMIA
C_1	8.8207×10^{-7}	8.6076×10^{-7}
C_2	1.1951×10^{-6}	1.0907×10^{-6}
C_3	2.07×10^{-6}	2.0489×10^{-6}

[1] The simulation used over one million frames at SNR=2.6, 3.2 and 4.2 dB, respectively for C_1, C_2 and C_3 (The FER performance is 10^{-3} for each decoder).

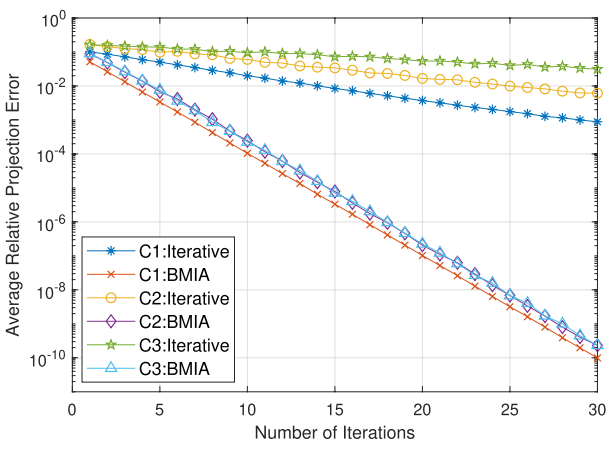

Fig. 2 Relative projection error of C_1, C_2 and C_3 for ADMM decoder with BMIA and Iterative projection algorithm for different iterations.

C_1, C_2 and C_3, respectively. As shown in table, there is almost no difference in the projection time between BMIA and Iterative algorithm under the same conditions ($I_{max} = 100$, $\epsilon = 0$, SNR= 2.6, 3.2, 4.2 dB for C_1, C_2 and C_3, respectively). The result demonstrates that, for each iteration, the computational complexities of BMIA and Iterative algorithm are almost equal.

Figure 2 describes the relationship between the number of iterations and the average relative projection error. The relative error is calculated as $\frac{||\mathbf{z}-\mathbf{z}_{true}||_2}{||\mathbf{z}_{true}||_2}$ ($\mathbf{z}_{true}$ denotes the accurate projection) at each projection. For each simulation point, 35000 frames are obtained. As shown in Fig. 2, for

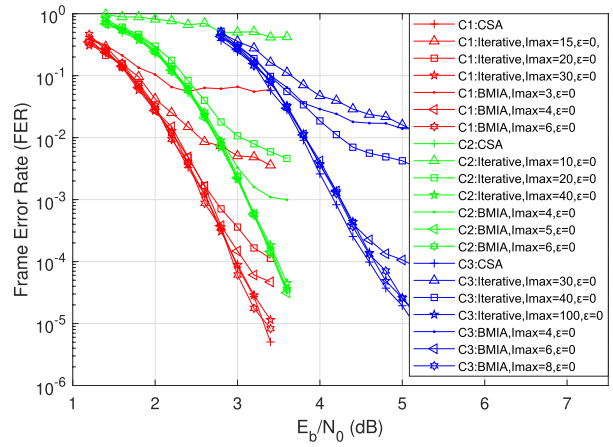

Fig. 3 FER performance of C_1, C_2 and C_3 for ADMM decoder with BMIA and Iterative projection algorithm for different I_{max}.

BMIA and Iterative projection algorithm, the average relative projection errors between the output projection and the accurate projection are decreasing as the number of iteration increase. It is intuitive that the speed of BMIA relative error decreasing is much faster than that of Iterative algorithm. Especially after the iteration number reaches 10 iterations, the average relative error of BMIA is closer to zero.

Figure 3 shows the relationship between the FER performance of the decoders with BMIA and Iterative projection algorithm for different codewords. According to the FER curves of C_1, C_2 and C_3, it's obvious that with the increase of I_{max}, the FER curves of BMIA and Iterative algorithm approach to the CSA FER curve gradually. However, the approaching rate of BMIA and Iterative algorithm are not the same. Especially when the dimension is 20, the FER performance of BMIA has a practically identical to that of CSA after 8 iterations, but FER curve of Iterative algorithm takes 100 iterations to approach the performance of CSA.

Finally, it is worth noting that when the average relative projection errors for BMIA is a little more than that of Iterative projection algorithm, the FER performance for BMIA is intuitively better than that of Iterative algorithm. For example, when the iteration of BMIA is 4 and the iteration of Iterative projection algorithm is 20, the average relative projection errors of BMIA is more than that of Iterative projection algorithm, while the FER performance of C_1 for BMIA is better than that for Iteartive projection algorithm. The projection of $\mathbf{v}$ onto $\mathcal{P}_{d_j}$ are given by $\prod_{[0,1]^{d_j}}(\mathbf{v} - s^*\boldsymbol{\theta}_{\mathcal{V}})$, $\prod_{[0,1]^{d_j}}(\mathbf{v} - \beta_{BMIA}\boldsymbol{\theta}_{\mathcal{V}})$, $\prod_{[0,1]^{d_j}}(\mathbf{v} - \beta_{Iter}\boldsymbol{\theta}_{\mathcal{V}})$ for correct projection, BMIA and Iterative projection algorithm, respectively. Based on the algorithm of BMIA and Iterative projection algorithm respectively, the value of β_{BMIA} could be greater than, less than or equal to s^*, while the value of β_{Iter} is always less than s^* [6]. Therefore, when the average relative projection errors for BMIA and Iterative projection algorithm are identical, the estimate projection calculated by BMIA might lie inside the check polytope, while that of Iterative projection algorithm must be outside of check polytope. To further verification, under the same average relative

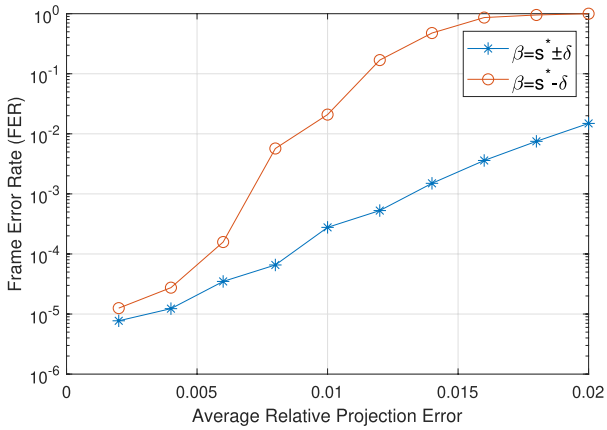

Fig. 4 FER performance of C_1 for ADMM decoder with different β at 3.4 dB

projection errors, the simulation explores the FER performances of code C_1 at 3.4 dB for the case of $\beta = s^* \pm \delta$ and $\beta = s^* - \delta$ ($\delta > 0$), respectively. As shown in Fig. 4, under the same average relative projection errors, the FER performance of decoder satisfying the projection possibly within the check polytope is better than that of decoder satisfying projection being outside of the check polytope.

5. Conclusion

To conclude, we propose a bisection method iterative algorithm in this paper. Compared with the existing projection algorithms, it does not require any sort operation, and the complexity of the algorithm is linear. Additionally, in comparison with the existing iterative projection algorithm, the proposed projection algorithm reduces the number of iterations without affecting the decoding performance. When the dimension of input vector is high, the number of iterations for the performance of the decoder with BMIA reaching optimal is only 1/10 of the iterative projection algorithm.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (61501334), The Fundamental Research Funds for the Central Universities (CCNU16A05028).

References

[1] S. Barman, X. Liu, S.C. Draper, and B. Recht, "Decomposition methods for large scale LP decoding," IEEE Trans. Inf. Theory, vol.59, no.12, pp.7870–7886, Dec. 2013.
[2] H. Wei, X. Jiao, and J. Mu, "Reduced-complexity linear programming decoding based on ADMM for LDPC codes," IEEE Commun. Lett., vol.19, no.6, pp.909–912, June 2015.
[3] X. Jiao, J. Mu, Y.C. He, and C. Chen, "Efficient ADMM decoding of LDPC codes using lookup tables," IEEE Trans. Commun., vol.65, no.4, pp.1425–1437, Jan. 2017.
[4] X. Zhang and P.H. Siegel, "Efficient iterative LP decoding of LDPC codes with alternating direction method of multipliers," Proc. IEEE Int. Symp. Inf. Theory, pp.1501–1505, Istanbul, Turkey, July 2013.
[5] G. Zhang, R. Heusdens, and W.B. Kleijn, "Large scale LP decoding with low complexity," IEEE Commun. Lett., vol.17, no.11, pp.2152–2155, Nov. 2013.
[6] H. Wei and A.H. Banihashemi, "An iterative check polytope projection algorithm for ADMM-based LP decoding of LDPC codes," IEEE Commun. Lett., vol.22, no.1, pp.29–32, Jan. 2018.
[7] I. Debbabi, N. Khouja, F. Tlili, B.L. Gal, and C. Jego, "Multicore implementation of LDPC decoders based on ADMM algorithm," IEEE International Conference on Acoustics, Speech and Signal Processing, pp.971–975, Shanghai, China, March 2016.
[8] X. Liu, ADMM Decoding of LDPC and Multipermutation Codes: From Geometries to Algorithms, Ph.D. thesis, University of Wisconsin, Madison, USA, 2015.
[9] X. Zhang, LDPC Codes–Structural Analysis and Decoding Techniques, Ph.D. thesis, University of California, San Diego, USA, 2012.
[10] X. Jiao, H. Wei, J. Mu, and C. Chen, "Improved ADMM penalized decoder for irregular low-density parity-check codes," IEEE Commun. Lett., vol.19, no.6, pp.913–916, June 2015.