l_2-Box ADMM Decoding for LDPC Codes Over ISI Channels

Xiaopeng Jiao, Haiyang Liu, Jianjun Mu, and Yu-Cheng He

***Abstract*—The alternating direction method of multipliers (ADMM) has been adapted for decoding low-density parity-check (LDPC) codes over intersymbol interference (ISI) channels in order to reduce the complexity of the traditional Turbo equalization (TE) method. However, the error rate performance of ADMM decoding may be degraded in certain ISI channels when compared with the TE at high signal-to-noise ratio regions. In order to address the problem, we derive a novel ADMM-based decoding method, called l_2-box ADMM decoding, for LDPC codes over ISI channels. By introducing suitable penalty functions to l_2-box ADMM decoding, we present an enhanced method, called l_2-box ADMM penalized decoding, to further improve the performance. Simulation results on several ISI channels indicate that the proposed l_2-box ADMM penalized decoding can achieve better error rate performances compared with the TE and the existing ADMM penalized decoding methods, especially for ISI channels with long memory, while maintaining the low-complexity advantage of the ADMM penalized decoding.**

***Index Terms*—l_2-box ADMM, ADMM decoding, ISI channel, LDPC codes, turbo equalization.**

I. Introduction

Recently, the alternating direction method of multipliers (ADMM) decoding of low-density parity-check (LDPC) codes has received a lot of interest [1]. Compared with the traditional LDPC decoding methods, the ADMM decoding can achieve better performances, especially in the error floor regions [2]. Besides, the ADMM decoding has several variants, which allow us to balance a better tradeoff between the decoding performance and complexity, see [3]–[6], [8]–[12] and the references therein.

The intersymbol interference (ISI) channel is a classical model for channels with memory, which is widely used in the communication and magnetic recording systems. Traditionally, the decoding of LDPC codes over ISI channels can be performed using the Turbo equalization (TE) [13] or the optimization-based methods [14]–[16]. A major drawback of these methods is that their complexities are exponential in the length of the channel memory. In order to reduce the decoding complexity, the ADMM-based decoding of LDPC codes over ISI channels has been proposed recently [17]. Simulation results indicate that the ADMM penalized decoding can outperform the traditional decoding methods over the ISI channel in many simulation cases. Moreover, the complexity of the ADMM penalized decoding is linear with the channel memory length [17], which is particularly suitable for ISI channels with long memory. It is worth mentioning that a special class of ISI channels with long memory, called sparse ISI channels, can be used to model high-rate mobile radio systems and aeronautical communication systems [18], [19].

While these results suggest that the ADMM decoding is promising, the decoding performance may be degraded for certain ISI channels (e.g., PR16 channel in [17]) compared with that of the TE at high signal-to-noise ratio (SNR) regions. Hence, it is deserved to develop more powerful ADMM-based decoding algorithm for LDPC codes over ISI channels, which motivates this work. It is known that the decoding of LDPC codes over ISI channels can be formulated as a quadratic programming (QP) problem [20]. From this QP problem, we derive a novel ADMM-based decoding method, called l_2-box ADMM decoding, for LDPC codes over ISI channels. Indeed, the l_p-box ADMM is a newly-developed framework that has been successfully applied to the optimization problems in several areas (e.g., machine learning) [21]. The l_p-box ADMM is also used for decoding LDPC codes over memoryless AWGN channels, which has some advantages over the ADMM penalized decoding [6]. This further motivates the research work in this paper. We choose $p = 2$ in this work since one of the key subproblems involved in the ADMM update steps has a closed-form solution in this case, which is beneficial from the implementation point of view.

In order to further improve the decoding performance, we introduce suitable penalty functions to l_2-box ADMM decoding, which leads to l_2-box ADMM penalized decoding. Simulation results reveal that the proposed l_2-box ADMM penalized decoding can achieve better error rate performances compared with the TE and the ADMM penalized decoding, especially for ISI channels with long memory, while maintaining the low-complexity merits of the ADMM decoding.

II. Background

For a communication system with LDPC codes over ISI channels, an information vector $\boldsymbol{u} \in \{0,1\}^K$ of length K is fed into an LDPC encoder, and $\boldsymbol{x} \in \{0,1\}^N$ represents the generated codeword of length N. After this, every element x_i of $\boldsymbol{x}$ is mapped to $1 - 2x_i$ using the binary phase shift keying (BPSK) modulation, where $i \in \mathcal{N} \triangleq \{1, 2, \ldots, N\}$. The modulated vector is then transmitted over an ISI channel with coefficients $(h_0, h_1, \ldots, h_D) \in \mathbb{R}^{D+1}$, where D is called the memory length of the channel. The received vector $\boldsymbol{r} \in \mathbb{R}^N$ has elements $r_i = \sum_{t=0}^{D} h_t(1 - 2x_{i-t}) + n_i$, where $i \in \mathcal{N}$ and n_i is an independent and identically distributed (i.i.d.) Gaussian random noise with mean zero and variance σ^2.

Let $\boldsymbol{H} = (H_{ji})_{M\times N}$ be the parity-check matrix of an LDPC code. Let $\mathcal{N}$ be the index set for the variable nodes in $G_{\boldsymbol{H}}$, the Tanner graph of $\boldsymbol{H}$, and $\mathcal{M} \triangleq \{1, 2, \ldots, M\}$ be the index set for the check nodes in $G_{\boldsymbol{H}}$. There exists an edge (v_i, c_j) in $G_{\boldsymbol{H}}$ that connects the variable node v_i and the check node c_j if and only if $H_{ji} = 1$. Let $N_v(i) = \{j \in \mathcal{M} : H_{ji} = 1\}$ be the index set for the neighbors of v_i, and $N_c(j) = \{i \in \mathcal{N} : H_{ji} = 1\}$ be the index set for the neighbors of c_j. The degrees of nodes v_i and c_j are given by $d_{v,i} = |N_v(i)|$ and $d_{c,j} = |N_c(j)|$, respectively.

Manuscript received June 10, 2020; revised December 29, 2020 and March 2, 2021; accepted March 21, 2021. Date of publication March 23, 2021; date of current version May 5, 2021. This work was supported in part by the National Natural Science Foundation of China under Grants 61977051, 61871376, and 61971322, in part by the Natural Science Foundation of Fujian Province under Grant 2018J01096, in part by the China Postdoctoral Science Foundation under Grant 2020M683427, in part by the Fundamental Research Funds for the Central Universities under Grant XJS200301, and in part by the Opening Project of Key Laboratory of Microelectronic Devices & Integrated Technology, Institute of Microelectronics, Chinese Academy of Sciences. The review of this article was coordinated by Prof. Ha H. Nguyen. *(Corresponding author: Jianjun Mu.)*

Xiaopeng Jiao and Jianjun Mu are with the School of Computer Science and Technology, Xidian University, Xi'an 710071, China (e-mail: jiaozi1216@126.com; jjmu@xidian.edu.cn).

Haiyang Liu is with the Institute of Microelectronics of Chinese Academy of Sciences, Beijing 100029, China (e-mail: liuhaiyang@ime.ac.cn).

Yu-Cheng He is with the School of Information Science and Engineering, Xiamen Key Lab of Mobile Multimedia Communications, Huaqiao University, Xiamen 361021, China, and also with the State Key Lab of Integrated Services Networks, Xidian University, Xi'an 710071, China (e-mail: yucheng.he@hqu.edu.cn).

Digital Object Identifier 10.1109/TVT.2021.3068398

The maximum likelihood decoding of LDPC codes over an ISI channel can be formulated as [17]

$$\begin{aligned}\min_{\boldsymbol{x}} \quad & f(\boldsymbol{x}) = \sum_{i=1}^{N}\left(r_i - \sum_{t=0}^{D} h_t\left(1-2x_{i-t}\right)\right)^2, \\ \text{s.t.} \quad & \boldsymbol{P}_j\boldsymbol{x} \in \mathbb{PP}_{d_{c,j}},\ j \in \mathcal{M},\ \boldsymbol{x} \in \{0,1\}^N, \end{aligned} \tag{1}$$

where $\boldsymbol{P}_j$ is the $d_{c,j} \times N$ binary matrix that selects the $d_{c,j}$ elements of $\boldsymbol{x}$ whose indices are in $N_c(j)$, and $\mathbb{PP}_{d_{c,j}}$ is the check polytope defined as the convex hull of all the length-$d_{c,j}$ binary vectors with an even number of 1's.

The problem given in (1) is a nonlinear integer programming problem, which is intractable to solve in general. In order to tackle the problem, we can relax problem (1) to a quadratic programming (QP) problem by replacing the constraint $\boldsymbol{x} \in \{0,1\}^N$ with $\boldsymbol{x} \in [0,1]^N$. Consequently, two ADMM penalized decoding algorithms for LDPC codes over ISI channels have been proposed [17]. These methods have some advantages over the conventional methods. However, the decoding performance of these methods may be degraded for certain ISI channels (e.g., PR16 channel in [17]) compared with that of the TE method at high SNR regions.

III. l_2-Box ADMM Decoding

It is known from [6] that the constraint $\boldsymbol{x} \in \{0,1\}^N$ in problem (1) can be equivalently written as[1]

$$\boldsymbol{x} \in [0,1]^N \cap \left\{\boldsymbol{x} : \left\|\boldsymbol{x} - \frac{1}{2}\mathbf{1}_N\right\|_2^2 = \frac{N}{4}\right\}, \tag{2}$$

where $\mathbf{1}_N$ denotes the all-one vector of length N. Then problem (1) can be reformulated as the following optimization problem

$$\begin{aligned}\min_{\boldsymbol{x}} \quad & f(\boldsymbol{x}) = \sum_{i=1}^{N}\left(r_i - \sum_{t=0}^{D} h_t\left(1-2x_{i-t}\right)\right)^2, \\ \text{s.t.} \quad & \boldsymbol{P}_j\boldsymbol{x} \in \mathbb{PP}_{d_{c,j}},\ j \in \mathcal{M}, \\ & \boldsymbol{x} \in [0,1]^N,\ \left\|\boldsymbol{x} - \frac{1}{2}\mathbf{1}_N\right\|_2^2 = \frac{N}{4}. \end{aligned} \tag{3}$$

In the following subsection, the ADMM method will be used to solve the problem in (3), and we call this method *the l_2-box ADMM decoding for LDPC codes over ISI channels.*

It should be noted that the l_p-box ADMM decoding problem for AWGN channels with $p \in (0, +\infty)$ has been provided in [6, Eq. (3)]. By setting $p = 2$, the l_2-box ADMM decoding problem for AWGN channels can be obtained. Through comparison, we know that the objective function of the l_2-box ADMM decoding problem is a linear function for AWGN channels [6], whereas a quadratic function for ISI channels. Thus, the ADMM update steps for ISI channels will be more complicated than those for AWGN channels.

[1]There exist other methods to replace the binary constraint $\boldsymbol{x} \in \{0,1\}^N$ by the equivalent constraints with continuous variables. For example, the variable $x_i \in \{0,1\}$ $(1 \le i \le N)$ can be replaced by the following set of equivalent constraints in [7]:

$$x_i(\hat{x}_i - 1) = 0, x_i = \hat{x}_i, 0 \le x_i \le 1,$$

where $\hat{x}_i$ is an auxiliary variable. Based on these constraints, a penalty dual decomposition (PDD) method has been developed in [7] for decoding LDPC codes over AWGN channels.

A. Derivation of Update Steps for l_2-Box ADMM Decoding

In order to make the problem in (3) fit for the ADMM template and decouple the two kinds of constraints in the problem, we introduce two auxiliary vectors $\boldsymbol{y} = (y_1, y_2, \ldots, y_N) \in \mathbb{R}^N$ and $\boldsymbol{z} = (\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_M) \in \mathcal{Z} \triangleq \mathbb{PP}_{d_{c,1}} \times \mathbb{PP}_{d_{c,2}} \times \cdots \times \mathbb{PP}_{d_{c,M}}$. Then the problem in (3) can be rewritten as

$$\begin{aligned}\min_{\boldsymbol{x}} \quad & f(\boldsymbol{x}) = \sum_{i=1}^{N}\left(r_i - \sum_{t=0}^{D} h_t\left(1-2x_{i-t}\right)\right)^2, \\ \text{s.t.} \quad & \boldsymbol{P}_j\boldsymbol{x} = \boldsymbol{z}_j,\ \boldsymbol{z}_j \in \mathbb{PP}_{d_{c,j}},\ j \in \mathcal{M}, \\ & \boldsymbol{x} = \boldsymbol{y},\ \boldsymbol{x} \in [0,1]^N,\ \left\|\boldsymbol{y} - \frac{1}{2}\mathbf{1}_N\right\|_2^2 = \frac{N}{4}. \end{aligned} \tag{4}$$

The augmented Lagrangian function of (4) is

$$\begin{aligned} L_{\mu_1,\mu_2}(\boldsymbol{x},\boldsymbol{y},\boldsymbol{z},\boldsymbol{\lambda}_1,\boldsymbol{\lambda}_2) = f(\boldsymbol{x}) + \frac{\mu_1}{2}\sum_{j\in\mathcal{M}}\|\boldsymbol{P}_j\boldsymbol{x} - \boldsymbol{z}_j + \boldsymbol{\lambda}_{1,j}\|_2^2 \\ -\frac{\mu_1}{2}\sum_{j\in\mathcal{M}}\|\boldsymbol{\lambda}_{1,j}\|_2^2 + \frac{\mu_2}{2}\|\boldsymbol{x} - \boldsymbol{y} + \boldsymbol{\lambda}_2\|_2^2 - \frac{\mu_2}{2}\|\boldsymbol{\lambda}_2\|_2^2, \end{aligned} \tag{5}$$

where $\mu_1, \mu_2 \in \mathbb{R}^+$ are penalty parameters, $\boldsymbol{\lambda}_1 = \{\boldsymbol{\lambda}_{1,j} : j \in \mathcal{M}\}$ with $\boldsymbol{\lambda}_{1,j} \in \mathbb{R}^{d_{c,j}}$ denotes the scaled dual variables for the constraint $\boldsymbol{P}_j\boldsymbol{x} = \boldsymbol{z}_j$, and $\boldsymbol{\lambda}_2 \in \mathbb{R}^N$ denotes the scaled dual variables for the constraint $\boldsymbol{x} = \boldsymbol{y}$.

For solving the problem in (4), the ADMM updates in the k-th iteration include the successive steps of $\boldsymbol{x}$-update, $(\boldsymbol{y},\boldsymbol{z})$-update, and $(\boldsymbol{\lambda}_1,\boldsymbol{\lambda}_2)$-update, which are given as follows

$$\boldsymbol{x}^k := \underset{\boldsymbol{x}\in[0,1]^N}{\operatorname{argmin}}\ L_{\mu_1,\mu_2}\left(\boldsymbol{x},\boldsymbol{y}^{k-1},\boldsymbol{z}^{k-1},\boldsymbol{\lambda}_1^{k-1},\boldsymbol{\lambda}_2^{k-1}\right), \tag{6}$$

$$\boldsymbol{y}^k := \frac{\boldsymbol{x}^k - \frac{1}{2}\mathbf{1}_N + \boldsymbol{\lambda}_2^{k-1}}{\left\|\boldsymbol{x}^k - \frac{1}{2}\mathbf{1}_N + \boldsymbol{\lambda}_2^{k-1}\right\|_2} \cdot \frac{\sqrt{N}}{2} + \frac{1}{2}, \tag{7}$$

$$\boldsymbol{z}_j^k := \Pi_{\mathbb{PP}_{d_{c,j}}}\left(\boldsymbol{P}_j\boldsymbol{x}^k + \boldsymbol{\lambda}_{1,j}^{k-1}\right), \quad j \in \mathcal{M}, \tag{8}$$

$$\boldsymbol{\lambda}_{1,j}^k := \boldsymbol{\lambda}_{1,j}^{k-1} + \boldsymbol{P}_j\boldsymbol{x}^k - \boldsymbol{z}_j^k, \quad j \in \mathcal{M}, \tag{9}$$

$$\boldsymbol{\lambda}_2^k := \boldsymbol{\lambda}_2^{k-1} + \boldsymbol{x}^k - \boldsymbol{y}^k. \tag{10}$$

The derivation of the $\boldsymbol{y}$-update is the same as in [6], except that the update in (7) is expressed in the scaled ADMM form. The operation $\Pi_{\mathbb{PP}_{d_{c,j}}}(\boldsymbol{v})$ in the $\boldsymbol{z}$-update denotes the Euclidean projection of a vector $\boldsymbol{v} \in \mathbb{R}^{d_{c,j}}$ onto the check polytope $\mathbb{PP}_{d_{c,j}}$.

Now we derive a closed-form expression for the $\boldsymbol{x}$-update in (6). By taking the first-order derivative of the augmented Lagrangian function in (5) with respect to $\boldsymbol{x}$, and letting it be zero, we have

$$\frac{\partial f}{\partial \boldsymbol{x}} + \mu_1\sum_{j\in\mathcal{M}}\boldsymbol{P}_j^T\left(\boldsymbol{P}_j\boldsymbol{x} - \boldsymbol{z}_j^k + \boldsymbol{\lambda}_{1,j}^k\right) + \mu_2\left(\boldsymbol{x} - \boldsymbol{y}^k + \boldsymbol{\lambda}_2^k\right) = \mathbf{0}. \tag{11}$$

Next we show that the update of each component x_i $(i \in \mathcal{N})$ can be performed independently. According to [17], $\frac{\partial f}{\partial x_i}$ can be expressed as

$$\frac{\partial f}{\partial x_i} = \left(8\sum_{t=0}^{D} h_t^2\right)x_i + \tilde{f}_{x_i}, \tag{12}$$

where the first term is related to x_i and the second term $\tilde{f}_{x_i}$ is the remainder in $\frac{\partial f}{\partial x_i}$ that is not related to x_i. In addition, $\sum_{j\in\mathcal{M}}\boldsymbol{P}_j^T\boldsymbol{P}_j$ is a diagonal matrix with the (i,i)-th entry equal to $d_{v,i}$ [1]. Therefore, by omitting the superscripts that index the iterations, (11) can be written as

$$\left(\mu_1 |N_v(i)| + 8\sum_{t=0}^{D} h_t^2 + \mu_2\right) x_i = \mu_1 \sum_{j\in N_v(i)} \left(z_j^{(i)} - \lambda_{1,j}^{(i)}\right) + \mu_2\left(y_i - \lambda_2^{(i)}\right) - \tilde{f}_{x_i}, \quad (13)$$

where $z_j^{(i)}$ denotes the i-th element of $\boldsymbol{P}_j^T \boldsymbol{z}_j$, $\lambda_{1,j}^{(i)}$ denotes the i-th element of $\boldsymbol{P}_j^T \boldsymbol{\lambda}_{1,j}$, and $\lambda_2^{(i)}$ is the i-th element of $\boldsymbol{\lambda}_2$. Combined (13) with the constraint $x_i \in [0,1]$, we obtain the update equation for x_i, which is given by

$$x_i = \Pi_{[0,1]}\left(\frac{\mu_1 \sum_{j\in N_v(i)}\left(z_j^{(i)} - \lambda_{1,j}^{(i)}\right) + \mu_2\left(y_i - \lambda_2^{(i)}\right) - \tilde{f}_{x_i}}{\mu_1 |N_v(i)| + 8\sum_{t=0}^{D} h_t^2 + \mu_2}\right), \quad (14)$$

where $\Pi_{[0,1]}(v)$ denotes the operation of projecting v onto the real interval [0,1].

B. l_2-Box ADMM Penalized Decoding

The penalty function included in the objective function of ADMM penalized decoding can improve the error rate performance since it makes non-integral solutions more costly [2]. In the following, we derive the l_2-box ADMM penalized decoding with both the l_1 and l_2 penalty functions.

1) l_1 Penalty Function: With the l_1 penalty function $g_1(x) = -\alpha_1\|\boldsymbol{x} - 0.5\|_1$, the objective function in problem (4) is changed to $f(x) + g_1(x)$ and the constraints in (4) do not change. Therefore, only the $\boldsymbol{x}$-update step needs to be modified. Other update steps in (7)–(10) remain unchanged. The augmented Lagrangian function for the l_2-box ADMM decoding with the l_1 penalty function can be written as

$$L_{l_1} = L_{\mu_1,\mu_2}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}, \boldsymbol{\lambda}_1, \boldsymbol{\lambda}_2) - \alpha_1 \|\boldsymbol{x} - 0.5\|_1 . \quad (15)$$

By setting $\frac{\partial L_{l_1}}{\partial \boldsymbol{x}} = \boldsymbol{0}$, we can obtain the update of x_i,

$$x_i = \Pi_{[0,1]}\left(\frac{q_i + \alpha_1 \cdot \mathrm{sgn}(x_i - 0.5)}{w_i}\right), \quad (16)$$

where

$$q_i = \mu_1 \sum_{j\in N_v(i)} \left(z_j^{(i)} - \lambda_{1,j}^{(i)}\right) + \mu_2\left(y_i - \lambda_2^{(i)}\right) - \tilde{f}_{x_i},$$

$$w_i = \mu_1 |N_v(i)| + 8\sum_{t=0}^{D} h_t^2 + \mu_2.$$

Using the similar method in [2] for the l_1 penalty function, we can eliminate the term $\mathrm{sgn}(x_i - 0.5)$ in the right hand side of (16) and obtain the ultimate update equation of x_i as follows

$$x_i = \begin{cases} \Pi_{[0,1]}\left(\frac{q_i+\alpha_1}{w_i}\right), & \text{if } q_i \geq \frac{w_i}{2}, \\ \Pi_{[0,1]}\left(\frac{q_i-\alpha_1}{w_i}\right), & \text{if } q_i < \frac{w_i}{2}, \end{cases} \quad (17)$$

2) l_2 Penalty Function: With the l_2 penalty function $g_2(x) = -\alpha_2\|\boldsymbol{x} - 0.5\|_2^2$, the objective function in (4) is changed to $f(x) + g_2(x)$. Again we only need to derive the $\boldsymbol{x}$-update in this case. The augmented Lagrangian function for the case is

$$L_{l_2} = L_{\mu_1,\mu_2}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}, \boldsymbol{\lambda}_1, \boldsymbol{\lambda}_2) - \alpha_2 \|\boldsymbol{x} - 0.5\|_2^2 . \quad (18)$$

Algorithm 1: The Proposed l_2-box ADMM Decoding.

Input: The ISI channel coefficients $\{h_0, h_1, \ldots, h_D\}$, the parity-check matrix $\boldsymbol{H}$, the received vector $\boldsymbol{r}$, and the maximum number of iterations I_{max}.
Output: $\hat{\boldsymbol{x}} = (\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_N) \in \{0,1\}^N$.
Initialize $\boldsymbol{\lambda}_j^0 \leftarrow \boldsymbol{0}_{d_{c,j}}$, $\boldsymbol{z}_j^0 \leftarrow \boldsymbol{0.5}_{d_{c,j}}$ for all $j \in \mathcal{M}$, $\boldsymbol{\lambda}_2^0 \leftarrow \boldsymbol{0}_N$ and $\boldsymbol{y}^0 \leftarrow \boldsymbol{0.5}_N$.
for $k = 1$ *to* $I_{\max}$ **do**
  Update x_i^k for all $i \in \mathcal{N}$ by (14) for l_2-box ADMM decoding, or (17) for l_2-box ADMM decoding with the l_1 penalty function, or (19) for l_2-box ADMM decoding with the l_2 penalty function.
  Update $\boldsymbol{y}^k$ by (7).
  Update $\boldsymbol{z}_j^k$ for all $j \in \mathcal{M}$ by (8).
  Update $\boldsymbol{\lambda}_{1,j}^k$ for all $j \in \mathcal{M}$ by (9).
  Update $\boldsymbol{\lambda}_2^k$ by (10).
  Decide $\hat{x}_i = 0$ if $x_i^k < 0.5$, or $\hat{x}_i = 1$ otherwise, for all $i \in \mathcal{N}$.
  If $\boldsymbol{H}\hat{\boldsymbol{x}} = \boldsymbol{0}_M$, then output $\hat{\boldsymbol{x}}$ and exit.
return the decoding failure.

By setting $\frac{\partial L_{l_2}}{\partial \boldsymbol{x}} = \boldsymbol{0}$, we can derive the $\boldsymbol{x}$-update for the l_2-box ADMM decoding with the l_2 penalty function as follows

$$x_i = \Pi_{[0,1]}\left(\frac{q_i - \alpha_2}{w_i - 2\alpha_2}\right). \quad (19)$$

It should be noted that for AWGN channels, the $\boldsymbol{x}$-update steps of the l_2-box ADMM penalized decoding, i.e., (17) and (19), should be changed by modifying q_i and w_i as follows

$$q_i' = \mu_1 \sum_{j\in N_v(i)} \left(z_j^{(i)} - \lambda_{1,j}^{(i)}\right) + \mu_2\left(y_i - \lambda_2^{(i)}\right) - \gamma_i,$$

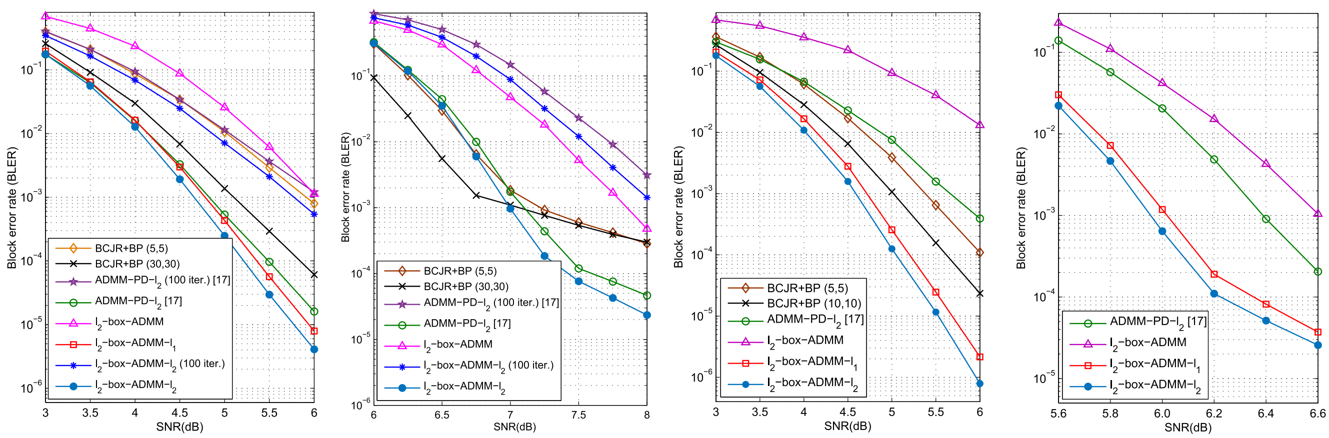
$$w_i' = \mu_1 |N_v(i)| + \mu_2,$$

where γ_i is the log-likelihood ratio of the ith received bit over an AWGN channel [2]. The other ADMM update steps in (7)–(10) should be kept for the same.

C. Algorithm Description and Complexity Analysis

The l_2-box ADMM decoding for LDPC codes over ISI channels is described in *Algorithm 1*. According to the different $\boldsymbol{x}$-update equations, the algorithm is denoted as `l2-box-ADMM` when no penalty function is used, or `l2-box-ADMM-l1` when the l_1 penalty function is used, or `l2-box-ADMM-l2` when the l_2 penalty function is used.

Let $\bar{d}_v$ and $\bar{d}_c$ be respectively the average degrees of the variable and check nodes in the Tanner graph of an LDPC code. In each iteration, the complexity in the $\boldsymbol{x}$-update in (14), (17) or (19) is dominated by the calculations of q_i and w_i. For q_i, the first two terms for all $i \in \mathcal{N}$ can be computed with complexity $\mathcal{O}(N\bar{d}_v)$. The term $\tilde{f}_{x_i}$ for all $i \in \mathcal{N}$ can be calculated with complexity $\mathcal{O}(ND)$ [17]. The complexity of computing w_i for all $i \in \mathcal{N}$ is $\mathcal{O}(N)$ since $8\sum_{t=0}^{D} h_t^2$ can be computed in advance. Thus the overall complexity of the $\boldsymbol{x}$-update per iteration is $\mathcal{O}(N(D + \bar{d}_v))$, which has the same order in complexity as the $\boldsymbol{x}$-update of ADMM penalized decoding proposed in [17]. Indeed, the calculation of x_i in (19) for each $i \in \mathcal{N}$ needs only two extra additions and one extra multiplication when compared with the calculation of x_i [17, Eq. (15)] in the original ADMM decoding. For $\boldsymbol{y}$-update in (7), the complexity per iteration is $\mathcal{O}(N)$. For $\boldsymbol{z}$-update in (8), the complexity per iteration is $\mathcal{O}(M\bar{d}_c)$ [17]. The complexity per iteration for $\boldsymbol{\lambda}_{1,j}$-update in (9) is $\mathcal{O}(M\bar{d}_c)$. For $\boldsymbol{\lambda}_2$-update in (10), the complexity



(a) (b) (c) (d)

Fig. 1. BLER performances for: (a) $\mathcal{C}_1$ over the EPR4 channel, (b) $\mathcal{C}_2$ over the EPR4 channel, (c) $\mathcal{C}_1$ over the PR16 channel, and (d) $\mathcal{C}_2$ over the S-PR16 channel.

TABLE I
PARAMETERS USED FOR ADMM-BASED DECODERS IN FIG. 1

Cases	`ADMM-PD-l`$_2$		`l`$_2$`-box-ADMM`		`l`$_2$`-box-ADMM-l`$_1$			`l`$_2$`-box-ADMM-l`$_2$		
	μ	α_2	μ_1	μ_2	μ_1	μ_2	α_1	μ_1	μ_2	α_2
Fig. 1 (a)	1.8	3.8	1.4	9.9	1.2	0.6	2.6	1.4	1.8	3.7
Fig. 1 (b)	1.0	3.0	0.8	11	–	–	–	0.7	1.4	2.6
Fig. 1 (c)	22.6	45.8	19	118	10	76	9	13	62	19
Fig. 1 (d)	4.4	13.4	2.8	49.2	3.0	0.2	11	3.0	17	8.5

per iteration is $\mathcal{O}(N)$. In addition, the complexities of hard decision and early termination in *Algorithm 1* are $\mathcal{O}(N)$ and $\mathcal{O}(M\bar{d}_c)$ respectively. In summary, the total complexity per iteration for the l_2-box ADMM decoding is $\mathcal{O}(N(D+3)+4M\bar{d}_c)$ (Note that $N\bar{d}_v = M\bar{d}_c$), which is linear with the channel memory length D.

IV. SIMULATION RESULTS

In this section, we compare the error rate performances of the proposed l_2-box ADMM decoding with the ADMM penalized decoding [17] (denoted by `ADMM-PD-l`$_2$) and the TE method [13] (denoted by `BCJR+BP`) that is served as a benchmark for the decoding of LDPC codes over ISI channels. The TE method uses the BCJR and belief propagation (BP) as the detector and decoder, respectively. The SNR for ISI channels is defined as $\mathrm{SNR} \triangleq \sum_{t=0}^{D} h_t^2 \big/ \sigma^2$. The maximum number of iterations for the ADMM-based decoders is set as 900 for all simulations unless otherwise specified. For each SNR value, at least 100 error frames are collected. Simulations are performed with the Microsoft Visual C++ 6.0 development tool on computers whose configurations are i5-3470 3.2 GHz CPU and 4 GB RAM.

Two LDPC codes are used in our simulations: 1) A rate-0.5 LDPC code $\mathcal{C}_1$, named 204.33.484 in [22], with parameters $N = 204$ and $M = 102$; 2) A rate-0.77 LDPC code $\mathcal{C}_2$, named 1057.244.3.352 in [22], with parameters $N = 1057$ and $M = 244$. For ISI channels, we apply 1) the EPR4 channel with $D = 3$ and $(h_0, h_1, h_2, h_3) = (0.5, 0.5, -0.5, -0.5)$; 2) the PR16 channel with $D = 16$ used in [17] and [20]; 3) the sparse ISI channel S-PR16 with $D = 16$ whose non-zero coefficients are $h_0 = 1.0, h_3 = 0.084, h_4 = -0.057, h_{11} = 0.018, h_{15} = -0.07$, and $h_{16} = 1.43$.

It should be noted that the two ISI channels with long memory $D = 16$ are used to illustrate the efficiency of the proposed method. In this case, the BCJR detector needs huge computational costs since the number of states in each section of the trellis is 2^{16}. However, the complexity of the proposed method grows linearly with the channel memory length.

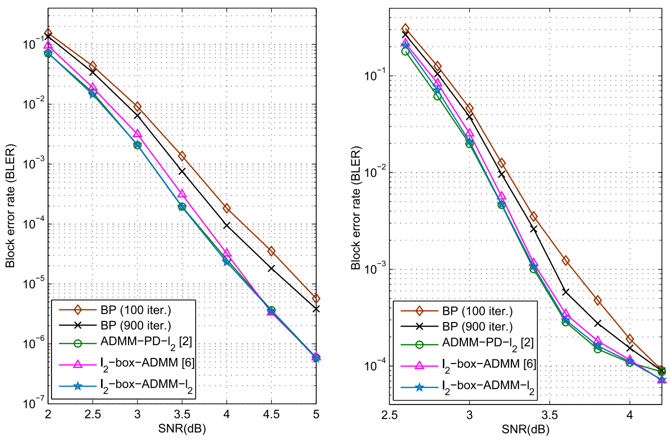
The parameters μ_1, μ_2 and α_1/α_2 involved in the l_2-box ADMM decoding are determined by the grid search with two steps. In the first step, we confine each parameter within a large range (e.g., [0100]) and use a large step size (e.g., 2.0). In the second step, we confine our search within a small range according to the results of the first step and set the search step to a small size (e.g., 0.2). Finally, the parameters with the best error rate performance are chosen. Note that the first step can confine the parameters to an effective range quickly and save a lot of computing time for the grid search. Table I lists the parameters we used for the ADMM-based decoders.

A. Performance Comparisons

Fig. 1(a) and Fig. 1(b) compare the block error rate (BLER) performances of the proposed l_2-box ADMM decoding, the `ADMM-PD-l`$_2$ and the `BCJR+BP` for $\mathcal{C}_1$ and $\mathcal{C}_2$ over the EPR4 channel, respectively. For clarity, we do not provide the BLER performance of `l`$_2$`-box-ADMM-l`$_1$ in Fig. 1(b), which is slightly worse than that of `l`$_2$`-box-ADMM-l`$_2$. It is known from these figures that the BLER performances of the l_2-box ADMM penalized decoding are considerably better than `BCJR+BP` in high SNR regions. In addition, `l`$_2$`-box-ADMM-l`$_2$ performs better than `ADMM-PD-l`$_2$ for both $\mathcal{C}_1$ and $\mathcal{C}_2$. For reference, we also provide the BLER performances of `ADMM-PD-l`$_2$ and `l`$_2$`-box-ADMM-l`$_2$ when the maximum number of iterations is set to 100. It can be seen that the BLER performances with 100 iterations are significantly worse than those with 900 iterations.

Fig. 1 (c) and Fig. 1 (d) show the BLER performances of various decoders for $\mathcal{C}_1$ over the PR16 channel and $\mathcal{C}_2$ over the S-PR16 channel, respectively. In Fig. 1 (d), we are unable to provide the BLER performance for `BCJR+BP` since both the computational and storage requirements are too huge to simulate with our computing resources. It can be seen that the proposed l_2-box ADMM penalized decoding can significantly outperform `ADMM-PD-l`$_2$ for ISI channels with long memory. From Fig. 1 (c) we can see that the proposed l_2-box ADMM penalized decoding can outperform `BCJR+BP` in high SNR regions.

For ease of comparison studies, we also provide the BLER performances of various ADMM decoders and the BP decoder for both $\mathcal{C}_1$ and $\mathcal{C}_2$ over AWGN channels, as shown in Fig. 2.



(a) (b)

Fig. 2. BLER performances for: (a) $\mathcal{C}_1$ and (b) $\mathcal{C}_2$ over the AWGN channel.

From the above simulation results we can observe two interesting phenomena:

1) The BLER performance of $\mathtt{l_2\text{-}box\text{-}ADMM}$ is worse than that of $\mathtt{ADMM\text{-}PD\text{-}l_2}$ for ISI channels. This is different from AWGN channels, where $\mathtt{l_2\text{-}box\text{-}ADMM}$ performs almost the same as or even better than $\mathtt{ADMM\text{-}PD\text{-}l_2}$ (see e.g., Fig. 2 and [6, Fig. 2]). Next, an intuitive explanation on this difference is presented. Generally, the use of ADMM for nonconvex problems lacks convergence guarantees. In particular, ADMM for nonconvex problems need not converge, and when it does converge, it need not converge to an optimal point [23]. Since the decoding problems of $\mathtt{ADMM\text{-}PD\text{-}l_2}$ and $\mathtt{l_2\text{-}box\text{-}ADMM}$ are nonconvex, there is no guarantee on the quality of the ADMM solutions. The objective function of the problem in (3) over ISI channels is a quadratic function, which is more complicated compared with AWGN channels. This makes the search space of $\mathtt{ADMM\text{-}PD\text{-}l_2}$ and $\mathtt{l_2\text{-}box\text{-}ADMM}$ over ISI channels more complicated than that for AWGN channels. Therefore, the possibility that $\mathtt{ADMM\text{-}PD\text{-}l_2}$ and $\mathtt{l_2\text{-}box\text{-}ADMM}$ converge to different local optimal points for ISI channels will be larger than that for AWGN channels. Due to the well-known difficulty of nonconvex optimization problems, the theoretical analysis of this difference seems to be intractable, and we leave it as a future research problem.

2) The performance gap between $\mathtt{l_2\text{-}box\text{-}ADMM\text{-}l_2}$ and $\mathtt{ADMM\text{-}PD\text{-}l_2}$ for long-memory ISI channels is significantly larger than that for short-memory ISI channels. It can be seen that $\mathtt{ADMM\text{-}PD\text{-}l_2}$ outperforms $\mathtt{BCJR+BP}$ for the EPR4 channel. However, $\mathtt{ADMM\text{-}PD\text{-}l_2}$ performs considerably worse than $\mathtt{BCJR+BP}$ for the PR16 channel. This implies that $\mathtt{ADMM\text{-}PD\text{-}l_2}$ has its limitation when decoding LDPC codes for certain ISI channels with long memory. In the following, we present an intuitive explanation of this observation. The decoding of LDPC codes over ISI channels can be jointly represented by a factor graph (see e.g., [14, Fig. 2]). Generally, the factor graph of an ISI channel with long memory is more complicated than that of an ISI channel with short memory. Thus, the polytopes of LP decoding are expected to have more vertices for an ISI channel with long memory. As a result, the possibility that $\mathtt{ADMM\text{-}PD\text{-}l_2}$ converges to a troublesome pseudocodeword will be increased, which leads to a performance degradation for $\mathtt{ADMM\text{-}PD\text{-}l_2}$ over an ISI channel with long memory. Compared with $\mathtt{ADMM\text{-}PD\text{-}l_2}$, $\mathtt{l_2\text{-}box\text{-}ADMM\text{-}l_2}$ is also imposed by another constraint $\|\boldsymbol{x} - \frac{1}{2}\mathbf{1}_N\|_2^2 = \frac{N}{4}$ on the decoding. This constraint can further penalize the pseudocodewords and make the components of $\boldsymbol{x}$ converge to 0 or 1. In other words, $\mathtt{l_2\text{-}box\text{-}ADMM\text{-}l_2}$ is more likely to converge to a codeword than $\mathtt{ADMM\text{-}PD\text{-}l_2}$.

TABLE II
ANI COMPARISONS IN SEVERAL SNR VALUES FOR ADMM-BASED DECODERS

Algorithms	Fig. 1 (a)			Fig. 1 (d)		
	5.0dB	5.5dB	6.0dB	6.2dB	6.4dB	6.6dB
$\mathtt{ADMM\text{-}PD\text{-}l_2}$	21.1	16.1	13.5	71.6	53.9	44.5
$\mathtt{l_2\text{-}box\text{-}ADMM\text{-}l_1}$	31.4	24.7	20.8	66.5	57.8	51.6
$\mathtt{l_2\text{-}box\text{-}ADMM\text{-}l_2}$	22.1	18.6	16.4	62.9	55.1	50.4

TABLE III
AVERAGE DECODING TIME (MS) PER FRAME FOR $\mathcal{C}_2$ OVER THE EPR4 CHANNEL

Algorithms	6.5dB	7.0dB	7.5dB	8.0dB
$\mathtt{BCJR+BP}$ (5,5)	50.6	32.2	20.9	14.4
$\mathtt{BCJR+BP}$ (30,30)	147.0	60.1	28.2	15.8
$\mathtt{ADMM\text{-}PD\text{-}l_2}$	45.0	19.8	12.8	9.5
$\mathtt{l_2\text{-}box\text{-}ADMM\text{-}l_2}$	74.6	39.8	28.2	22.0

TABLE IV
AVERAGE DECODING TIME (MS) PER FRAME FOR $\mathcal{C}_1$ OVER THE PR16 CHANNEL

Algorithms	4.5dB	5.0dB	5.5dB	6.0dB
$\mathtt{BCJR+BP}$ (5,5)	5192	4244	3920	3758
$\mathtt{ADMM\text{-}PD\text{-}l_2}$	10.3	5.6	3.2	2.3
$\mathtt{l_2\text{-}box\text{-}ADMM\text{-}l_2}$	6.9	5.5	4.9	4.4

B. Comparisons of ANI and Running Time

In Table II, we present the average number of iterations (ANIs) in several SNR values for $\mathcal{C}_1$ over the EPR4 channel (cf. Fig. 1 (a)) and $\mathcal{C}_2$ over the S-PR16 channel (cf. Fig. 1(d)) with the l_2-box ADMM penalized decoding and $\mathtt{ADMM\text{-}PD\text{-}l_2}$. For $\mathcal{C}_1$ over the EPR4 channel, the ANIs for $\mathtt{l_2\text{-}box\text{-}ADMM\text{-}l_1}$ and $\mathtt{l_2\text{-}box\text{-}ADMM\text{-}l_2}$ are larger than those for $\mathtt{ADMM\text{-}PD\text{-}l_2}$. For $\mathcal{C}_2$ over the S-PR16 channel, the ANIs for $\mathtt{ADMM\text{-}PD\text{-}l_2}$ are less than those for the l_2-box ADMM penalized decoding at high SNR values. In contrast, the ANIs for the l_2-box ADMM penalized decoding are less than those for $\mathtt{ADMM\text{-}PD\text{-}l_2}$ at SNR=6.2 dB. Moreover, the ANIs for $\mathtt{l_2\text{-}box\text{-}ADMM\text{-}l_2}$ are slightly less than those for $\mathtt{l_2\text{-}box\text{-}ADMM\text{-}l_1}$. For other simulation cases, similar results can be obtained.

Next, we compare the average decoding time per frame for $\mathtt{BCJR+BP}$, $\mathtt{ADMM\text{-}PD\text{-}l_2}$ and $\mathtt{l_2\text{-}box\text{-}ADMM\text{-}l_2}$ in several SNR values. Tables III and IV compare the running time for $\mathcal{C}_2$ over the EPR4 channel and $\mathcal{C}_1$ over the PR16 channel, respectively. For $\mathcal{C}_2$ over the EPR4 channel, the running time of $\mathtt{BCJR+BP}$ is comparable to that of the ADMM-based decoders. However, the ADMM-based decoders run much faster than $\mathtt{BCJR+BP}$ for PR16 channel with long memory size. This is because the BCJR detector for PR16 channel has 2^{16} states in each section of its trellis which makes it expensive to calculate the forward and backward probabilities. It can also be seen that $\mathtt{l_2\text{-}box\text{-}ADMM\text{-}l_2}$ runs slower than $\mathtt{ADMM\text{-}PD\text{-}l_2}$, especially in high SNR regions. Therefore, compared with the ADMM penalized decoding in [17], the proposed l_2-box ADMM penalized decoding provides a better tradeoff between the decoding performance and complexity.

V. CONCLUSION

We have investigated the l_2-box ADMM decoding for LDPC codes over ISI channels. Compared with the TE method, the proposed decoder has two advantages: 1) The complexity of the proposed methods is linear with the channel memory length, while the BCJR+BP has an exponential complexity in the channel memory length; 2) Unlike the BCJR detector which is a serial method in essence, each update steps of the proposed decoder can be performed in parallel. Compared with the existing ADMM penalized decoding, the proposed l_2-box ADMM penalized decoding has better decoding performance with a slight increase in computational complexity.

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] X. Liu and S. C. Draper, "The ADMM penalized decoder for LDPC codes," *IEEE Trans. Inf. Theory*, vol. 62, no. 6, pp. 2966–2984, Jun. 2016.

[3] I. Debbabi, B. L. Gal, N. Khouja, F. Tlili, and C. Jego, "Fast converging ADMM penalized algorithm for LDPC decoding," *IEEE Commun. Lett.*, vol. 20, no. 4, pp. 644–647, Apr. 2016.

[4] X. Jiao, J. Mu, Y. He, and C. Chen, "Efficient ADMM decoding of LDPC codes using look-up tables," *IEEE Trans. Commun.*, vol. 65, no. 4, pp. 1425–1437, Apr. 2017.

[5] 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.

[6] Q. Wu, F. Zhang, H. Wang, J. Lin, and Y. Liu, "Parameter-free l_p-box decoding of LDPC codes," *IEEE Commun. Lett.*, vol. 22, no. 7, pp. 1318–1321, Jul. 2018.

[7] M.-M. Zhao, Q. Shi, Y. Cai, M.-J. Zhao, and Q. Yu, "Decoding binary linear codes using penalty dual decomposition method," *IEEE Commun. Lett.*, vol. 23, no. 6, pp. 958–962, Jun. 2019.

[8] Q. Xia, Y. Lin, S. Tang, and Q. Zhang, "A fast approximate check polytope projection algorithm for ADMM decoding of LDPC codes," *IEEE Commun. Lett.*, vol. 23, no. 9, pp. 1520–1523, Sep. 2019.

[9] M. Wasson, M. Milicevic, S. C. Draper, and G. Gulak, "Hardware-based linear program decoding with the alternating direction method of multipliers," *IEEE Trans. Signal Process.*, vol. 67, no. 19, pp. 4976–4991, Oct. 2019.

[10] J. Bai, Y. Wang, and Q. Shi, "Efficient QP-ADMM decoder for binary LDPC codes and its performance analysis," *IEEE Trans. Signal Process.*, vol. 68, pp. 503–518, Jan. 2020.

[11] F. Gensheimer, T. Dietz, K. Kraft, S. Ruzika, and N. Wehn, "A reduced-complexity projection algorithm for ADMM-based LP decoding," *IEEE Trans. Inf. Theory*, vol. 66, no. 8, pp. 4819–4833, Aug. 2020.

[12] Y. Wei, M.-M. Zhao, M.-J. Zhao, and M. Lei, "ADMM-based decoder for binary linear codes aided by deep learning," *IEEE Commun. Lett.*, vol. 24, no. 5, pp. 1028–1032, May 2020.

[13] R. Koetter, A. C. Singer, and M. Tüchler, "Turbo equalization," *IEEE Signal Process. Mag.*, vol. 21, no. 1, pp. 67–80, Jan. 2004.

[14] M. H. Taghavi and P. H. Siegel, "Graph-based decoding in the presence of ISI," *IEEE Trans. Inf. Theory*, vol. 57, no. 4, pp. 2188–2202, Apr. 2011.

[15] M. F. Flanagan, "A unified framework for linear-programming based communication receivers," *IEEE Trans. Commun.*, vol. 59, no. 12, pp. 3375–3387, Dec. 2011.

[16] B. Kim and H. D. Pfister, "Joint decoding of LDPC codes and finite-state channels via linear-programming," *IEEE J. Sel. Topics Signal Process.*, vol. 5, no. 8, pp. 1563–1576, Dec. 2011.

[17] X. Jiao, J. Mu, Y. He, and W. Xu, "Linear-complexity ADMM updates for decoding LDPC codes in partial response channels," *IEEE Commun. Lett.*, vol. 23, no. 12, pp. 2200–2204, Dec. 2019.

[18] N. C. McGinty, R. A. Kennedy, and P. Hoeher, "Parallel trellis viterbi algorithm for sparse channels," *IEEE Commun. Lett.*, vol. 2, no. 5, pp. 143–145, May 1998.

[19] J. Mietzner, S. Badri-Hoeher, I. Land, and P. Hoeher, "Trellis-based equalization for sparse ISI channels revistied," in *Proc. IEEE Int. Symp. Inf. Theory*, Adelaide, Australia, Sep. 2005, pp. 229–233.

[20] T. Wadayama, "Interior point decoding of linear vector channels based on convex optimization," *IEEE Trans. Inf. Theory*, vol. 56, no. 10, pp. 4905–4921, Oct. 2010.

[21] B. Wu and B. Ghanem, "l_p-box ADMM: A versatile framework for integer programming," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 7, pp. 1695–1708, Jul. 2019.

[22] D. J. C. MacKay, *Encyclopedia of Sparse Graph Codes*. Accessed: Dec. 2019. [Online]. Available: http://www.inference.phy.cam.ac.uk/mackay/codes/data.html.

[23] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, Jan. 2011.