LETTER

Triangle Projection Algorithm in ADMM-LP Decoding of LDPC Codes

Yun JIANG[†], Huiyang LIU[†], Xiaopeng JIAO[††], Ji WANG[†], *and* Qiaoqiao XIA[†a)], *Nonmembers*

SUMMARY In this letter, a novel projection algorithm is proposed in which projection onto a triangle consisting of the three even-vertices closest to the vector to be projected replaces check polytope projection, achieving the same FER performance as exact projection algorithm in both high-iteration and low-iteration regime. Simulation results show that compared with the sparse affine projection algorithm (SAPA), it can improve the FER performance by 0.2 dB as well as save average number of iterations by 4.3%.

key words: *alternating direction method of multipliers (ADMM), low-density parity-check (LDPC) codes, check polytope projection, triangle projection algorithm*

1. Introduction

Linear programming (LP) decoding of low-density parity check (LDPC) codes was first proposed by Feldman in [1]. Limited by its high decoding complexity, LP decoding algorithm was not widely used in the early days. Recently, inspired by the idea of the alternating direction multiplier method (ADMM) [2], Barman et al. proposed a linear programming decoding algorithm based on the alternating direction multiplier method (ADMM-LP) [3], reducing the complexity of LP decoding algorithm to a certain extent. However, compared with the belief propagation (BP) decoding algorithm [4], ADMM-LP has poor decoding performance at low signal-to-noise-ratios (SNRs). To enhance its decoding performance, Liu et al. proposed an ADMM decoding algorithm using a weighted penalty function [5]. References [6] and [7] introduced deep learning into ADMM-based decoder and provided a new idea for ADMM decoding. Despite these improvements, a major drawback of ADMM-LP decoding is that its decoding complexity is still high.

Simplifying the Euclidean projection onto check polytope is an efficient way to reduce the computational complexity of ADMM-LP. Zhang et al. proposed a simplification of the Euclidean projection using cut search algorithm (CSA) in [8]. G. Zhang et al. [9] transformed the complex projection onto a check polytope to projection onto a simplex. Gensheimer et al. presented a reduced-complexity projection algorithm relying on the findings that some components of the input can be fixed to 0 or 1, while the remaining components have to be projected onto a smaller-dimensional even or odd parity polytope [10].

Although the above exact projection algorithms have reduced the decoding complexity of ADMM-LP, there is still a lot of room for improvement. Recently, approximate projection algorithms with lower complexity have been proposed by scholars. Xia et al. proposed the even-vertex projection algorithm (EVA) [11], [12] and the line-segment algorithm (LSA) [13], [14], respectively projecting onto the closest even-vertex and the line-segment consisting of the two closest vertices of the polytope. Asadzadeh et al. proposed the sparse affine projection algorithm (SAPA) [15], projecting onto the affine hull of a small number of vertices of the polytope. However, experiments show that these approximate algorithms lose their accuracy in low-iteration regime.

In this work, by replacing line segment projection with projection onto a triangle consisting of the three even-vertices closest to the vector to be projected, we will propose a fast and highly accurate projection algorithm based on LSA. Simulation results show that compared with LSA, the proposed algorithm can improve the FER performance by 0.5 dB as well as save average number of iterations by 11.0%, and compared with SAPA, the proposed algorithm can improve the FER performance by 0.2 dB as well as save average number of iterations by 4.3%.

2. Preliminaries

Consider an LDPC code C defined by an $m \times n$ parity check matrix $\mathbf{H}$. Let $i \in I = \{1,2,3\ldots,n\}$ and $j \in J = \{1,2,3\ldots,m\}$ be the indexes of the columns and rows of check matrix $\mathbf{H}$, respectively. The degree of variable node v_i (check node c_j) is denoted by d_i (d_j). Let $N_v(i)$ ($N_c(j)$) denotes the set of check nodes (variable nodes) adjacent to variable node v_i (check node c_j).

Suppose a codeword $\boldsymbol{x} = \{x_i \in \{0,1\} | i \in I\}$ is transmitted over memoryless binary input output-symmetric (MBIOS) channel and $\boldsymbol{y} = \{y_i | i \in I\}$ is the received vector. The LP decoding model based on ADMM with L2 penalty is described as follows:

$$\begin{aligned} &\min_{\boldsymbol{x}} \boldsymbol{\gamma}^T \boldsymbol{x} - \alpha \left\| \boldsymbol{x} - 0.5 \right\|_2^2 \\ &s.t.\ \boldsymbol{P}_j x = z_j, z_j \in P_{d_j}, \forall j \in J \end{aligned} \tag{1}$$

where $\boldsymbol{\gamma} = \{\gamma_i | i \in I\}$ represents the log-likelihood ratios (LLRs) vector, the i-th entry of which is $\gamma_i = \log(\frac{\Pr(y_i|x_i=0)}{\Pr(y_i|x_i=1)})$. $\boldsymbol{P}_j$ is a $d_j \times n$ transfer matrix and P_{d_j} is the convex hull of all

Manuscript received November 28, 2023.
Manuscript revised February 17, 2024.
Manuscript publicized March 18, 2024.

[†]College of Physical Science and Technology, Central China Normal University, Wuhan, 430079, China.
[††]School of Computer Science and Technology, Xidian University, Xian, 710126, China.
a) E-mail: xiaqq@ccnu.edu.cn (Corresponding author)

DOI: 10.1587/transfun.2023EAL2104

permutations of a length-d_j binary vector with even number of 1s, and it is called the check polytope. $\boldsymbol{z}_j$ is the auxiliary vector.

The augmented Lagrangian function corresponding to formulation (1) and the update rule for $\boldsymbol{z}$ are as follows:

$$L_\mu(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{\lambda}) = \boldsymbol{\gamma}^T \boldsymbol{x} + \sum_{j\in J} \boldsymbol{\lambda}_j{}^T (\boldsymbol{P}_j \boldsymbol{x} - \boldsymbol{z}_j) + \frac{\mu}{2} \sum_{j\in J} \|\boldsymbol{P}_j \boldsymbol{x} - \boldsymbol{z}_j\|_2^2 - \alpha \|\boldsymbol{x} - 0.5\|_2^2 \tag{2}$$

$$\boldsymbol{z}_j{}^{k+1} = \prod\nolimits_{P_{d_j}} (\boldsymbol{P}_j \boldsymbol{x}^{k+1} + \boldsymbol{\lambda}_j^k / \mu) \tag{3}$$

where $\boldsymbol{\lambda}_j \in \mathbb{R}^{d_j}$ denotes the Lagrangian multiplier, $\mu > 0$ is a positive penalty parameter, $\alpha > 0$ is a positive penalty parameter of L2 penalty, $k \geq 0$ is the iteration number, and $\prod_{P_{d_j}}$ is the check polytope projection operation.

3. ADMM-LP Decoding Algorithm with Triangle Projection Algorithm

Recently, a large number of scholars have carried out research on the Euclidean projection onto check polytope, proposing various exact projection algorithms and approximate projection algorithms and improving the decoding performance of ADMM-LP. In this section, we will propose TPA inspired by LSA and illustrate it in detail.

3.1 LSA

Even-vertex algorithm (EVA) is the simplest way to replace Euclidean projection with approximate projection but has poor FER performance. To improve this defect, Xia et al. proposed LSA, replacing check polytope projection with line segment projection. For a given vector $\boldsymbol{v}$, the odd-vertex closest to $\boldsymbol{v}$ can be solved in accordance with the indicator vector $\boldsymbol{\theta}_V$ (referred in [8, Algorithm 2]), which can be calculated as Algorithm 1, line 2-6, as follows:

$$O_i = \begin{cases} 1, & \theta_{V,i} = 1 \\ 0, & \theta_{V,i} = -1 \end{cases} \tag{4}$$

After identifying the odd-vertex closest to $\boldsymbol{v}$, the two even-vertexes closest to $\boldsymbol{v}$ can be obtained as follows:

$$\boldsymbol{A} = \begin{cases} O_i, & i \neq p \\ 1 - O_i, & i = p \end{cases} \qquad \boldsymbol{B} = \begin{cases} O_i, & i \neq q \\ 1 - O_i, & i = q \end{cases} \tag{5}$$

where p and q are the indexes of the two elements closest to 0.5 in the vector $\boldsymbol{v}$.

We can calculate the projection of $\boldsymbol{v}$ onto the line segment $L_{\boldsymbol{AB}}$ by Algorithm 2 in [13].

3.2 TPA

Simulation results show that when combined with over-relaxation method, LSA performs poorly in FER in low-iteration regime. This is because that as an approximate projection algorithm, the result of LSA is far away from the accurate projection, and LSA requires more iterations to converge.

Algorithm 1 Triangle Projection Algorithm (TPA)

```
Input: Vector $\boldsymbol{v} \in \mathbb{R}^{d_j}$
Output: Projection $\boldsymbol{z}$
1: The projection of $\boldsymbol{v}$ onto unit hypercube: $\boldsymbol{u} = \prod_{[0,1]^{d_j}} \boldsymbol{v}$
2: Initialize the indicator vector $\boldsymbol{\theta}_V$ : $\theta_{V,i} = sgn(v_i - 0.5)$
3: if $|\{i | \theta_{V,i} = 1\}|$ is even then
4:     $i^* = \arg\min_{i\in[d_j]} (|v_i - 0.5|)$
5:     $\theta_{V,i^*} = -\theta_{V,i^*}$
6: end if
7: $s = |\{\theta_{V,i} | \theta_{V,i} = 1\}|$
8: if $\boldsymbol{\theta}_V^T \boldsymbol{u} \leq s - 1$ then
9:     $\boldsymbol{z} = \boldsymbol{u}$
10: else
11:     Odd-vertex $\boldsymbol{O}$ : $O_i = \begin{cases} 1, \theta_{V,i} = 1 \\ 0, \theta_{V,i} = -1 \end{cases}$
12:     $p = \arg\min_i (|v_i - 0.5|)$, $q = \arg\min_{i/p} (|v_i - 0.5|)$,
        $r = \arg\min_{i/(p,q)} (|v_i - 0.5|)$
13:     $\boldsymbol{\theta}'_V = (\theta_{V,p}, \theta_{V,q}, \theta_{V,r})$, $\boldsymbol{v}' = (v_p, v_q, v_r)$
14:     Run Algorithm 2 with input $\boldsymbol{v}'$, $\boldsymbol{\theta}'_V$ and get output $\eta$
15:     $\boldsymbol{z} = \boldsymbol{O}$
16:     $z_p = \prod_{[0,1]} (v'_1 - \eta\theta'_{V,1})$, $z_q = \prod_{[0,1]} (v'_2 - \eta\theta'_{V,2})$,
        $z_r = \prod_{[0,1]} (v'_3 - \eta\theta'_{V,3})$
17: end if
18: return $\boldsymbol{z}$
```

Asadzadeh et al. proposed SAPA, projecting onto the affine hull of a small number of vertices of the polytope. Although SAPA has a similar FER performance as that of CSA in high-iteration regime, it loses its accuracy in low-iteration regime as the projection result of SAPA may not be on the check polytope, creating some errors.

In order to make the projection more accurate, we try to replace the line segment projection with the triangle projection. Similar to LSA, to determine a triangle only needs to determine its three vertices, that is, to solve the three even-vertices on the check polytope that are closest to $\boldsymbol{v}$. They can be obtained as follows:

$$\boldsymbol{A} = \begin{cases} O_i, & i \neq p \\ 1 - O_i, & i = p \end{cases} \qquad \boldsymbol{B} = \begin{cases} O_i, & i \neq q \\ 1 - O_i, & i = q \end{cases} \qquad \boldsymbol{C} = \begin{cases} O_i, & i \neq r \\ 1 - O_i, & i = r \end{cases} \tag{6}$$

where p, q, and r are the indexes of the three elements closest to 0.5 in the vector $\boldsymbol{v}$. Since $\boldsymbol{A}$, $\boldsymbol{B}$ and $\boldsymbol{C}$ are only different at p, q, and r, the projection on the triangle ABC can be considered as the projection in 3D space.

The specific process of TPA algorithm is shown in Algorithm 1. Above all, initialize the indicator vector $\boldsymbol{\theta}_V$. Then determine the odd-vertex closest to $\boldsymbol{v}$ and subsequently obtain the three even-vertices. Afterward we refer to the CSA

Algorithm 2 The solution of η

```
Input: Vector $\boldsymbol{v}'$, $\boldsymbol{\theta}'_V$
Output: Scalar $\eta$
 1: $V = \{i|\theta'_{V,i} > 0\}$
 2: $T \leftarrow \{v'_i - 1|v'_i > 1\} \cup \{-v'_i|v'_i < 0\}$,
    $\delta \leftarrow \boldsymbol{\theta}'^{T}_{V}\boldsymbol{v}' - |V| + 1$, and $\zeta = 3$.
 3: if $T \neq \varnothing$ then
 4:     Sort elements in $T = \{t_i\}$ such that $t_i \geq t_{i+1}$.
 5:     for $i = 1$ to $|T|$ do
 6:         if $\delta/\zeta > t_i$ then
 7:             Exit to line 13.
 8:         else
 9:             $\delta \leftarrow \delta - t_i$ and $\zeta \leftarrow \zeta - 1$
10:         end if
11:     end for
12: end if
13: return $\eta = \delta/\zeta$
```

Table 1 Number of operations required by four projection algorithms for P_{d_j}.

Algorithm	Comparison	Addition	Multiplication
	General	General	General
CSA	$\lambda\log\lambda+\lambda+2d$	$2d+2\lambda+2$	λ
LSA	$3d+4$	$d+9$	1
SAPA	$3d-3$	8	1
TPA(proposed)	$4d+19$	$d+16$	1

algorithm in [8] to solve the projection coefficient η. Finally, we obtain the elements at p, q, r in Algorithm 1, line 16 and get the projection $\boldsymbol{z}$.

The solution of η is shown in Algorithm 2.

Table 1 describes the number of comparisons, additions and multiplications for CSA, LSA, SAPA and the proposed TPA, respectively. The number of comparisons, additions and multiplications for CSA, LSA and SAPA can be found in the references [13] and [15]. There, d and λ respectively denote the dimension of the polytope, and the number of elements in set T (referred in [8, Algorithm 3, Line 1]). In conclusion, the complexity of LSA, SAPA, and TPA are all $O(d)$, and the complexity of the proposed TPA is slightly higher than that of LSA and SAPA while much lower than that of CSA.

It should be noted that the dimension of $\boldsymbol{v}'$ in Algorithms 2 is 3, and the sorting elements in $T = \{t_i\}$ only requires 3 comparisons at most. Additionally, the loop operation introduced by Algorithm 2 can be expanded into three separate computational processes, that is, there is no loop in the proposed algorithm and the parallelization capability of the proposed algorithm is almost the same with LSA.

4. Simulation Results

In our simulations, all the algorithms are implemented in C programming language and run on a computer with Intel Core i9 CPU (10 cores and 3.7 GHz), 64GB main memory. The additive white Gaussian noise (AWGN) channel with binary phase shift keying (BPSK) modulation is supposed. We consider four LDPC codes with different rates: the regular (1920, 640) rate-1/3 Gallager code C_1, the regular (2640, 1320) rate-1/2 Margulis code C_2, the irregular (576, 432) rate-3/4 code C_3 from IEEE 802.16e standard [16], and 5G-NR LDPC code C_4 with information length $K = 320$ and rate $R = 8/9$. The check node degrees of C_1-C_4 are 4, 6, {14,15}, and {3, 19}, respectively.

Table 2 The μ and α for each code.

	C_1 (1920,640)	C_2 (2640,1320)	C_3 (576,432)	C_4 $K=320, R=8/9$
μ	5.5	3	4.5	7.5
α	0.9	0.8	1.2	0.1

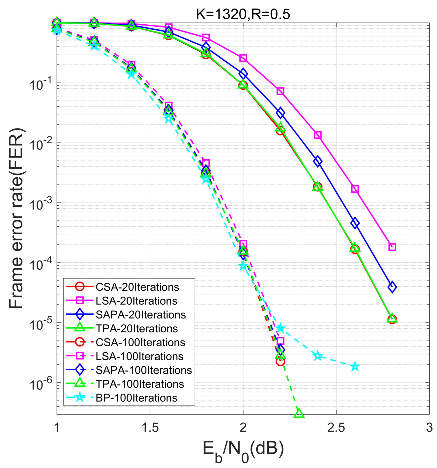


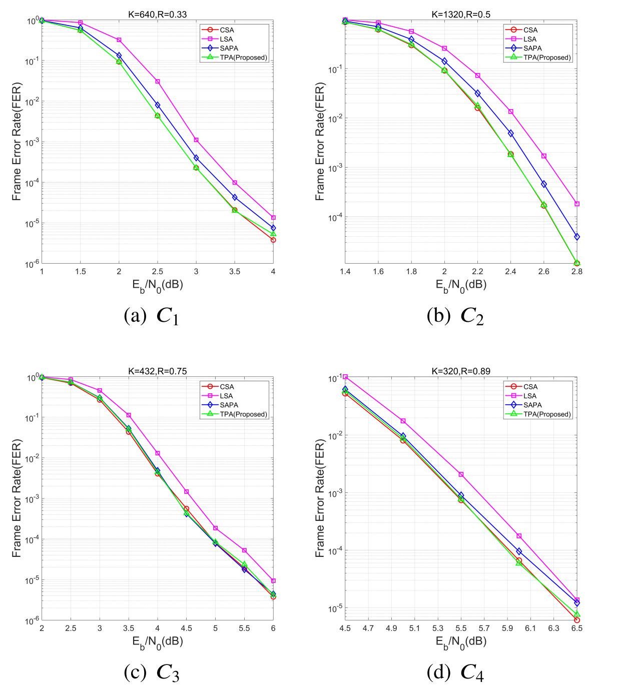
Fig. 1 Frame error rate for CSA, LSA and SAPA for C_2 for both low-iteration (maximum set to 20) and high-iteration (maximum set to 100).

The ADMM-LP decoding algorithm with L2 penalty combined with over-relaxation technique is adopted in the simulations, and the relaxation coefficient is 1.9. Considering that CSA is an exact projection algorithm, we optimize the parameters of CSA and apply them to all algorithms. The parameters for each code are shown in Table 2, where $\mu > 0$ is a positive penalty parameter of the augmented Lagrangian function and $\alpha > 0$ is a positive penalty parameter of L2 penalty function [5].

Figure 1 shows the comparison of the FER performance of approximate projection algorithms LSA, SAPA and the exact projection algorithm CSA for C_2 in low-iteration regime and high-iteration regime, respectively. As shown in the figure, BP algorithm has poor performance in the high-SNR regime, that is it suffers from an “error floor” phenomenon at high SNRs (occurs at about 2.2 dB), and although the FER performance of LSA and SAPA are quite close to CSA in the high-iteration regime, significant gaps are present in the low-iteration regime. Specifically, when FER = 2×10^{-4}, the FER performance of SAPA is worse than that of CSA about 0.1 dB, and the FER performance of LSA is worse than that of CSA about 0.25 dB.

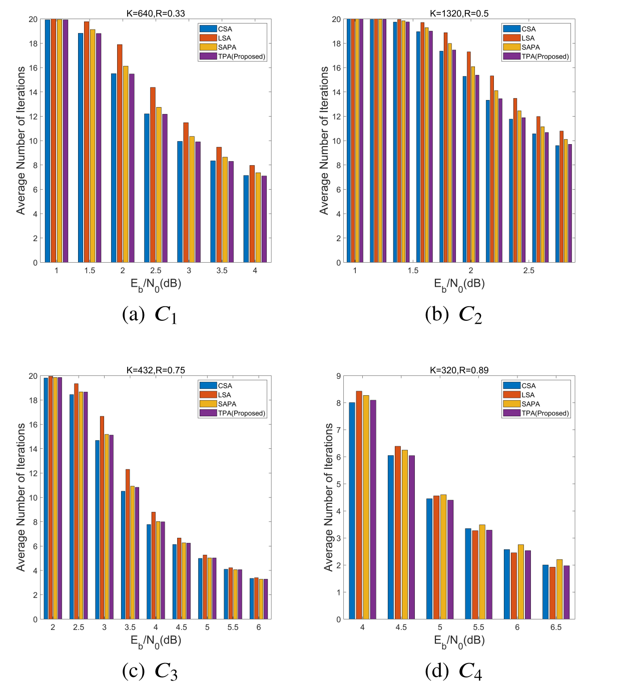
In order to solve this problem, we propose TPA, whose decoding performance can be almost the same as that of CSA even in the low-iteration regime.

Figure 2 plots the FER performance for different projection algorithms for $C_1 - C_4$ for low-iteration. Obviously,



(a) C_1 (b) C_2 (c) C_3 (d) C_4

Fig. 2 Frame error rate for CSA, LSA, SAPA and TPA for $C_1 - C_4$ for low-iteration (maximum set to 20).

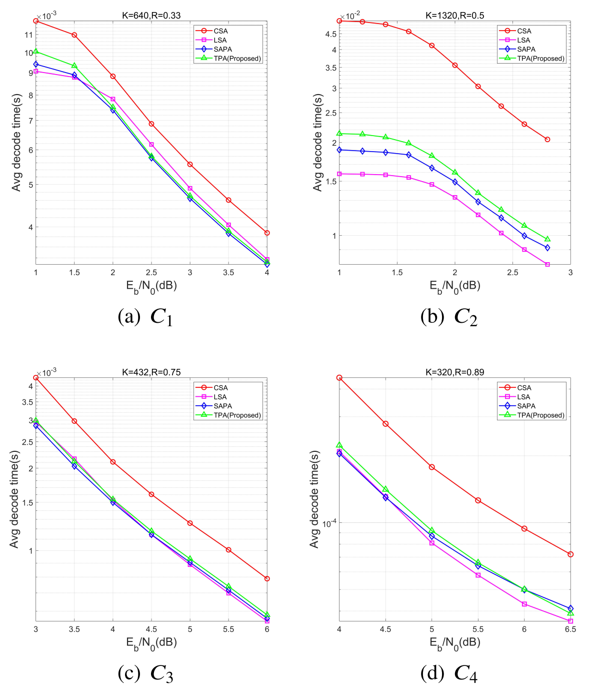


(a) C_1 (b) C_2 (c) C_3 (d) C_4

Fig. 3 The average number of iterations for CSA, LSA, SAPA and TPA for $C_1 - C_4$ for low-iteration (maximum set to 20).

the proposed TPA outperforms LSA and SAPA, and its FER performance is almost the same as that of the exact projection algorithm CSA. For instance, for C_1, when FER $= 2 \times 10^{-5}$, the FER performance of SAPA is worse than that of TPA about 0.2 dB.

Figure 3 shows the average number of iterations for



(a) C_1 (b) C_2 (c) C_3 (d) C_4

Fig. 4 Average decoding time for CSA, LSA, SAPA and TPA for $C_1 - C_4$.

different projection algorithms for $C_1 - C_4$ for low-iteration. All data are acquired by generating at least 100, 000 frames and average number of iterations is obtained by (total number of iterations)/(total number of frames). Figure 3 suggests that the average number of iterations of the decoder with the proposed TPA is almost the same as that of the decoder with CSA and less than that of the decoder with other algorithms, which means that the proposed algorithm converges quickly. For example, the average number of iterations is saved by 11.01% that of LSA, and 4.3% that of SAPA for C_2 at 2.0 dB.

The comparison of average decoding time is plotted in Fig. 4. It is noteworthy that although CSA has the fewest iterations, its average decoding time is the longest because of its much higher complexity than approximate projection algorithms. Figure 4 shows that the average decoding time of TPA is slightly more than that of SAPA and significantly less than that of CSA. Specifically, for C_2, the average decoding time of TPA is 6.7% more than that of SAPA while 55.1% less than that of CSA.

5. Conclusion

By replacing check polytope projection with projection onto a triangle consisting of the three even-vertices closest to the vector to be projected, we proposed a fast and efficient projection algorithm. While most of the existing approximate projection algorithms sacrifice some decoding performance in exchange for decoding efficiency, losing their accuracy at low iterations, the FER performance of TPA is almost the same as that of the exact projection algorithm for low-iteration. Compared with SAPA, the proposed algorithm can improve the FER performance by 0.2 dB as well as save average number of iterations by 4.3%.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under Grant 62101204.

References

[1] J. Feldman, M.J. Wainwright, and D.R. Karger, “Using linear programming to decode binary linear codes,” IEEE Trans. Inf. Theory, vol.51, no.3, pp.954–972, March 2005.

[2] 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.

[3] 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.

[4] F.R. Kschischang, B.J. Frey, and H.-A. Loeliger, “Factor graphs and the sum-product algorithm,” IEEE Trans. Inf. Theory, vol.47, no.2, pp.498–519, Feb. 2001.

[5] X. Liu and S.C. Draper, “The ADMM penalized decoder for LDPC codes,” IEEE Trans. Inf. Theory, vol.62, no.6, pp.2966–2984, June 2016.

[6] 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.

[7] X. Guo, T.-H. Chang and Y. Wang, “Model-driven deep learning ADMM decoder for irregular binary LDPC codes,” IEEE Commun. Lett., vol.27, no.2, pp.571–575, Feb. 2023.

[8] 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, July 2013.

[9] 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.

[10] 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.

[11] Q. Xia, X. Wang, H. Liu, and Q.L. Zhang, “A hybrid check polytope projection algorithm for ADMM decoding of LDPC codes,” IEEE Commun. Lett., vol.25, no.1, pp.108–112, 2020.

[12] Y. Zheng, Y. Lin, Z. Zhang, Q. Zhang, and Q. Xia, “An enhanced HDPC-EVA decoder based on ADMM,” IEICE Trans. Fundamentals, vol.E104-A, no.10, pp.1425–1429, Oct. 2021.

[13] 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, Sept. 2019.

[14] Q. Xia, P. He, X. Wang, H. Liu, and Q. Zhang, “Node-wise scheduling algorithm of ADMM decoding based on line segment projection,” IEEE Commun. Lett., vol.26, no.4, pp.738–742, April 2022.

[15] A. Asadzadeh, M. Barakatain, S.C. Draper, and J. Mitra, “SAPA: Sparse affine projection algorithm in ADMM-LP decoding of LDPC codes,” 2022 17th Canadian Workshop on Information Theory (CWIT), Ottawa, ON, Canada, pp.27–32, 2022.

[16] LDPC Coding for OFDMA PHY, Standard IEEE Std. C802.16e-05/0066r3, 2005.