Gradient Descent Bit-Flipping Based on Penalty Factor for Decoding LDPC Codes over Symmetric Alpha-Stable Noise Channels

Chenyu Gao[1], Rongke Liu[1,*], and Bin Dai[1]

[1]School of Electronic and Information Engineering, Beihang University, Beijing, P.R. China; Email: gaochenyu@buaa.edu.cn

***Abstract*—A modified gradient descent bit-flipping (GDBF) algorithm is proposed for decoding low-density parity-check (LDPC) codes over the symmetric alpha-stable (SαS) impulsive noise channel. To simplify the log-likelihood rate (LLR) computational implement, this paper introduces a linear approximation of the LLR for the SαS impulsive noise. Combined with the linear approximation, the new algorithm introduces a penalty factor into the syndrome components of the inversion function. Simulation results show that the GDBF algorithm based on the penalty factor performs better than GDBF algorithm.**

***Index Terms*—LDPC codes, EXIT chart, impulsive noise, symmetric alpha-stable distribution.**

I. Introduction

Low-density parity-check (LDPC) codes [1], [2] are widely applied to different recent standards such as WiMAX and DVB-S2, owing to their high error-correction capability. They can be decoded by soft-decision and hard-decision decoding methods. As a soft-decision scheme, belief propagation (BP) [2] algorithm can achieve excellent performance, but with high implementation complexity. The hard-decision algorithms, such as the bit-flipping (BF) decoding algorithm and its variants [1], [3], offer a tradeoff between performance and complexity. These algorithms have been investigated over additive white Gaussian noise (AWGN) channels. However, some communication systems, such as wireless networks [4], shallow water communications [5] and powerline communications (PLC) [6], are impaired by non-Gaussian noise which exhibits an impulsive interference. The impulsive noises last for a very short duration and have a heavy-tailed characteristic that have much greater amplitudes than Gaussian noises. The non-Gaussian impulsive noise channel can be successfully modeled by the additive white symmetric alpha-stable noise (AWSαSN) . As we know, the decoding process is sensitive to the noise: the performance of the decoding process which has been designed under Gaussian noise assumption is significantly degraded when the noise is impulsive. However, there is very little literature on the study of decoding algorithms for LDPC codes over AWSαSN channels.

For LDPC codes, the maximum likelihood decoding (MLD) problem can be formulated as an integer programming (IP). It was proven that the MLD problem belongs to the class of NP-hard problems. The gradient descent bit-flipping (GDBF) [7] algorithm is proposed based on the optimization of a non-linear objective function which follows the MLD rule. As a bit-flipping algorithm, the GDBF algorithm outperforms the weight BF (WBF) [3] and its variants [8], [9]. Inserting a weighting on syndrome and a zero-mean Gaussian perturbation term, an improved GDBF named the noisy GDBF (NGDBF) [10] was proposed for the AWGN channel. The advance of GDBF algorithm has mainly been achieved over the AWGN and BSC channels. However, the GDBF algorithm suffers severe degradation and its performance is worse than the WBF algorithm over the AWSαSN channel.

In this paper, to improve the performance of GDBF algorithm, we propose a modified GDBF algorithm based on the penalty factor for the AWSαSN channel. Our contributions are divided into two aspects: 1. we propose a linear approximation of the LLR for AWSαSN channels to simplify the LLR computational implement. 2. we derive the lower bound of the penalty factor for the proposed modified GDBF algorithm combined with the linear approximation.

This paper is organized as follows: Section II describes the background knowledge of the system model and the GDBF algorithm. Section III presents the modified GDBF algorithm based on the penalty factor (GDBF-PF) for AWSαSN channels. Results are shown in Section IV and finally we conclude the paper in Section V.

II. Background

A. System Model

The received symbols over AWSαSN channels are given by

$$y = x + z, \tag{1}$$

where $x \in \{-1, +1\}$ is the channel input after the binary phase-shift keying (BPSK) modulation and the noise z follows the SαS distribution. The SαS distribution is a generalization of the Gaussian distribution that accommodates impulsive characteristics. There are no closed-form expression of probability density function (PDF) for SαS distribution except the Gaussian ($\alpha = 2$) and Cauchy ($\alpha = 1$) distribution. Generally, the SαS distribution is defined by its characteristic function

$$\phi(\omega) = e^{-|\gamma\omega|^{\alpha}}, \tag{2}$$

where $\gamma > 0$ is the dispersion which measures the spread of the SαS PDF. α is the characteristic exponent ($0 < \alpha \leq 2$),

$$\lambda_j = \begin{cases} \frac{\sqrt{2}y}{\gamma}, |y| < \sqrt{\sqrt{2}(\alpha+1)\gamma}, \\ \sqrt{\frac{2\sqrt{2}(\alpha+1)}{\gamma}} - \frac{\sqrt{2}(y-\sqrt{\sqrt{2}(\alpha+1)\gamma})}{(\beta-1)\gamma}, \sqrt{\sqrt{2}(\alpha+1)\gamma} \le y \le \beta\sqrt{\sqrt{2}(\alpha+1)\gamma}, \\ -\sqrt{\frac{2\sqrt{2}(\alpha+1)}{\gamma}} - \frac{\sqrt{2}(y+\sqrt{\sqrt{2}(\alpha+1)\gamma})}{(\beta-1)\gamma}, -\beta\sqrt{\sqrt{2}(\alpha+1)\gamma} \le y \le -\sqrt{\sqrt{2}(\alpha+1)\gamma}, \\ 0, |y| > \beta\sqrt{\sqrt{2}(\alpha+1)\gamma}. \end{cases} \quad (3)$$

which represents the tail heaviness of the SαS PDF. Since the second-order moment of SαS distribution does not exist, the geometric signal-to-noise ratio (GSNR) is used instead of SNR for AWSαSN channels. The GSNR is given by [11]

$$\text{GSNR} = \frac{1}{2C_g}\left(\frac{A}{C_g^{\frac{1}{\alpha}-1}\gamma}\right)^2, \quad (4)$$

where $A = 1$ and $C_g \approx 1.78$. The corresponding E_b/N_0 for AWSαSN channels is defined as

$$\frac{E_b}{N_0} = \frac{\text{GSNR}}{2R}, \quad (5)$$

where R is the code rate.

B. The GDBF algorithm

The binary LDPC code $C \in \{0, 1\}$ is a linear block code given by the null space of an $m \times n$ parity-check matrix H that comprises elements $h_{ij} \in \text{GF}(2)$, where $i = 1, 2, \cdots, m$ and $j = 1, 2, \cdots, n$. Let $u = (u_1, u_2, ..., u_n), u_j \in \{+1, -1\}$ be the tentative decoded bipolar sequence at the end of a decoding iteration. Then the syndrome vector $s = (s_1, s_2, ..., s_m)$ can be computed by $s_i = \prod\limits_{j \in N(i)} u_j$.

The GDBF algorithm is proposed in [7] by considering the gradient descent optimization of a objective function which follows the MLD rule. The MLD problem is to find a codeword in C which gives the largest correlation to the received word y. It is equivalent to find the global solution of the following binary integer programming problem:

$$\begin{aligned} &\text{maximize} \sum_{j=1}^{n} \lambda_j \cdot u_j \\ &\text{s.t.} \sum_{i=1}^{m} \prod_{j \in N(i)} u_j = m, \end{aligned} \quad (6)$$

where λ_j is called the likelihood function which is calculated by the the log-likelihood rate (LLR), $\lambda_j = \log \frac{P(y_j|x_j=1)}{P(y_j|x_j=-1)}$. For AWGN channels, the likelihood function LLR is linear, which can be replaced by the received symbols. So the objective function of GDBF combined with the syndrome components is defined as:

$$f(u) \triangleq \sum_{j=1}^{n} y_j \cdot u_j + \sum_{i=1}^{m} \prod_{j \in N(i)} u_j. \quad (7)$$

Then the solution of the MLD problem is to maximize the objective function (7).

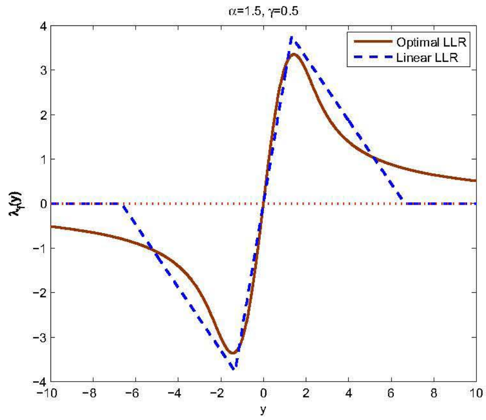

Fig. 1: Oprimal LLR and linear approximation for $\alpha = 1.5$, $\gamma = 0.5$.

III. The GDBF algorithm based on penalty factor

A. Linear approximation of LLR

The original GDBF is designed for AWGN channels, where λ_j in (6) is replaced by y_j. However, it will suffer a huge degradation for AWSαSN channels when using y_j. The main reason is that impulsive noises have much greater amplitudes than Gaussian noises and the likelihood function LLR is nonlinear over AWSαSN channels. Moreover, this nonlinear function has no closed expression, but we can approximate it as a nonlinear piecewise [12] function on y. When y is close to 0, the LLR is almost linear: $\lambda_j \approx \frac{\sqrt{2}y}{\gamma}$. On another hand, when y is large enough, the LLR is approximated as $\lambda_j \approx \frac{2(\alpha+1)}{y}$.

To make the LLR computation simple and easy to implement on hardware, in this paper, we present a linear piecewise approximation to replace the nonlinear piecewise function. The LLR linear piecewise approximation is divided into four parts which can be described as (3), where β is a variable which controls the slope of the second and third parts for the LLR linear approximation. As an example, the optimal LLR and linear approximation as functions on y_j are represented in Fig. 1. In simulation, we let β equal to 5.

B. The improved GDBF algorithm

As the MLD problem is NP-hard, it is difficult to solve this problem directly. The penalty function method which transfers constraints of (6) to the objective function is a important tool to solve this problem. Applying the penalty function method, problem (6) is replaced by the following equivalent problem

$$f(u) \triangleq \sum_{j=1}^{n} \lambda_j \cdot u_j + p \cdot (\sum_{i=1}^{m} \prod_{j \in N(i)} u_j - m), \tag{8}$$

where p is the penalty factor, which is sufficiently large. To let the algorithm implementable, we give a lower bound of p and the deduction is as follow:

The bound of the LLR for the SαS noise is

$$|\lambda_j| \leq \sqrt{\frac{2\sqrt{2}(\alpha+1)}{\gamma}}. \tag{9}$$

Then $\sum_{j=1}^{n} \lambda_j \cdot u_j$ has the bound:

$$-\mathrm{n} \cdot \sqrt{\frac{2\sqrt{2}(\alpha+1)}{\gamma}} \leq \sum_{j=1}^{n} \lambda_j \cdot u_j \leq \mathrm{n} \cdot \sqrt{\frac{2\sqrt{2}(\alpha+1)}{\gamma}}. \tag{10}$$

If u is a codeword, the object function is equal to

$$f(u) = \sum_{j=1}^{n} \lambda_j \cdot u_j. \tag{11}$$

Otherwise, it is

$$f(u) = \sum_{j=1}^{n} \lambda_j \cdot u_j + p \cdot (\sum_{i}^{m} \prod_{j \in N(i)} u_j - m). \tag{12}$$

In this case, we have $-m \leq \sum_{i}^{m} \prod_{j \in N(i)} u_j - m \leq -1$ and hence $p(\sum_{i}^{m} \prod_{j \in N(i)} u_j - m) \leq -p$. For large p enough, the penalty factor insures that $f(u)$ in (8) can not achieve the maximum local when u is not a codeword and implies that

$$f(u) \leq -n \cdot \sqrt{\frac{2\sqrt{2}(\alpha+1)}{\gamma}}. \tag{13}$$

Let $W = \sum_{i}^{m} \prod_{j \in N(i)} u_j - m$, then (13) equals to

$$\sum_{j=1}^{n} \lambda_j \cdot u_j + p \cdot W \leq -n \cdot \sqrt{\frac{2\sqrt{2}(\alpha+1)}{\gamma}}. \tag{14}$$

Since $-m \leq W \leq -1$, we have

$$p \geq \frac{-n \cdot \sqrt{\frac{2\sqrt{2}(\alpha+1)}{\gamma}} - \sum_{j=1}^{n} \lambda_j \cdot u_j}{W}, \tag{15}$$

where $n \cdot \sqrt{\frac{2\sqrt{2}(\alpha+1)}{\gamma}} + \sum_{j=1}^{n} \lambda_j \cdot u_j \geq 0$, and

$$\begin{aligned} \frac{-n \cdot \sqrt{\frac{2\sqrt{2}(\alpha+1)}{\gamma}} - \sum_{j=1}^{n} \lambda_j \cdot u_j}{W} &= \frac{n \cdot \sqrt{\frac{2\sqrt{2}(\alpha+1)}{\gamma}} + \sum_{j=1}^{n} \lambda_j \cdot u_j}{-W} \\ &\leq n \cdot \sqrt{\frac{2\sqrt{2}(\alpha+1)}{\gamma}} + \sum_{j=1}^{n} \lambda_j \cdot u_j \\ &\leq n \cdot \sqrt{\frac{2\sqrt{2}(\alpha+1)}{\gamma}} + n \cdot \sqrt{\frac{2\sqrt{2}(\alpha+1)}{\gamma}} \\ &\leq 2n \cdot \sqrt{\frac{2\sqrt{2}(\alpha+1)}{\gamma}}. \end{aligned} \tag{16}$$

Therefore, if $p \geq 2n \cdot \sqrt{\frac{2\sqrt{2}(\alpha+1)}{\gamma}}$, then $p \geq \frac{-n \cdot \sqrt{\frac{2\sqrt{2}(\alpha+1)}{\gamma}} - \sum_{j=1}^{n} \lambda_j \cdot u_j}{W}$. It means that $2n \cdot \sqrt{\frac{2\sqrt{2}(\alpha+1)}{\gamma}}$ is a sufficient lower bound for the penalty factor p.

To find the maximum of the function $f(u)$, we use the gradient descend algorithm. As a result, the flipping rule is to find the largest absolute value of the partial derivative which depends on the inversion function:

$$E_k = \lambda_k \cdot u_k + p \cdot \sum_{i \in M(k)} \prod_{j \in N(i)} u_k. \tag{17}$$

Then the proposed GDBF algorithm based on penalty factor (GDBF-PF) is described as follows:

Algorithm 1 The GDBF-PF algorithm

1: Compute syndrome vector s. for all i, if $s_i = +1$, stop and u is the codeword, otherwise go to next step.
2: Calculate inversion functions at VNs using (17)
3: Find the flip position given by $k = \arg\min_{k \in \{1,2,\ldots,n\}} E_k$
4: If the stopping rule is not satisfied then return to step 2; otherwise the valid codeword is decoded or maximum number of iterations is reached.

IV. Results and Discussion

Simulation results are performed on the Mackay (PEGReg504x1008) LDPC code [13] over AWSαSN channels. We compare the proposed GDBF-PF algorithm with the GDBF and WBF algorithms. The comparison results under different impulsive levels are shown in Fig.2–4. The characteristic exponent values are set to 1.9, 1.5 and 1.1, which represent the channel environments under weak, moderate and strong impulsive behaviors, respectively. In each figure, the maximum number of decoding iterations for both BF (GDBF, WBF and GDBF-PF) algorithms is set to 100. While the maximum number of decoding iterations for MS decoder is set to 5.

Fig.2 shows the BER performance of different algorithms for the weak impulsive noise ($\alpha = 1.9$). It can be seen that the GDBF algorithm performs worse and presents an error floor at a BER $= 10^-3$. In this case, the BER curve of the WBF algorithm decreases slowly with the E_b/N_0 increases. However, at a BER $= 10^-4$, the proposed GDBF-PF algorithm can significantly improve the performance of the GDBF algorithm and is about 1.3 dB worse than the MS decoder. Moreover, the proposed GDBF-PF also outperforms the WBF algorithm and is better than the WBF algorithm achieves 3dB.

Fig.3 shows the BER performance of different algorithms for the moderate impulsive noise ($\alpha = 1.5$). Compared to $\alpha = 1.9$, the impulsive noise is stronger when $\alpha = 1.5$. As a result, both the GDBF algorithm and the WBF algorithm exhibit error floors. However, with the help of the penalty factor, GDBF-PF algorithm can suit the AWSαSN channels well and still performs well, which is about 2.3 dB worse than the MS decoder at a BER $= 10^-4$.

Fig.4 shows the BER performance of different algorithms for the strong impulsive noise ($\alpha = 1.1$). Similarly, the intensity of impulse noise is further strengthened, so the GDBF

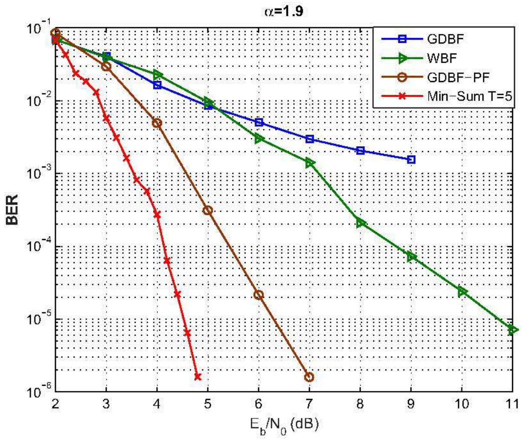

Fig. 2: BER results of several algorithms at $\alpha = 1.9$.

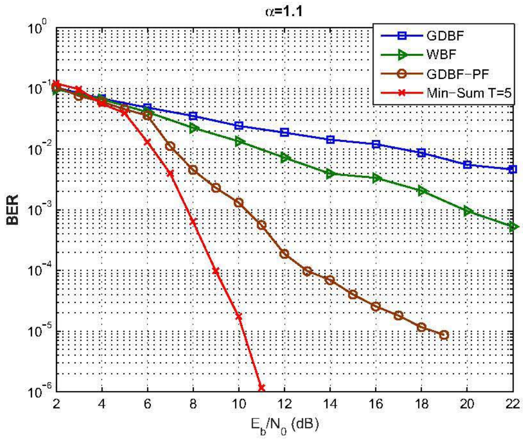

Fig. 4: BER results of several algorithms at $\alpha = 1.1$.

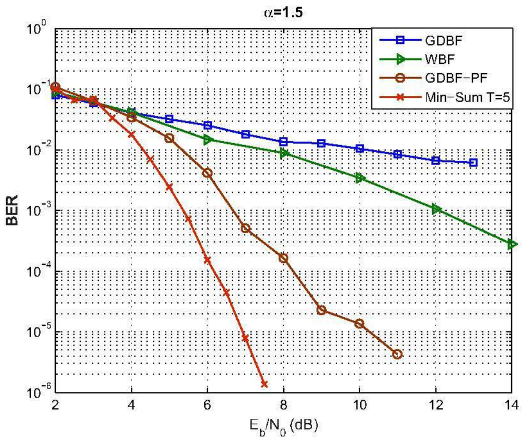

Fig. 3: BER results of several algorithms at $\alpha = 1.5$.

algorithm and WBF algorithm still perform worse and exhibit error floors. On the contrary, at a BER $= 10^-4$, the GDBF-PF algorithm performs 4 dB worse than the MS decoder. From this result, we can find that the GDBF-PF algorithm can greatly improve the performance of GDBF algorithms by its penalty factor, even under poor channel environments.

As we know, the smaller the α value, the stronger the impulse noise. The GDBF algorithm exhibits error floors and even worse than WBF algorithm over the AWSαSN channel. However, the proposed GDBF-PF algorithm can greatly improve the performance of GDBF algorithm especially for the strong impulsive noise. The main reason is that the decoding is more dependent on the parity syndrome values in later iterations and the penalty factor p can enhance the influence of the parity syndrome values on the inversion function. Thus, the performance of the GDBF algorithm can be improved.

V. Conclusion

In this letter, a simplified LLR linear approximation for AWSαSN channels is proposed and an improved GDBF algorithm based on penalty factor for LDPC code over AWSαSN channels is introduced. In the proposed algorithm, the lower bound of the penalty factor is derived. The experimental results show that the GDBF algorithm exhibits error floors and is worse than WBF algorithm for AWSαSN channels. However, the propose GDBF-PF algorithm can achieve a better performance compared with the GDBF and WBF algorithms.

Acknowledgment

This paper is supported in part by the Program for New Century Excellent Talents in University of China under Grant No. NCET-12-0030 and the National Natural Science Foundation of China under Grant No. 91438116 and No.61401010.

References

[1] R. G. Gallager, "Low-density parity-check codes," *IRE Trans. Inf. Theory.*, vol. 8, no. 1, pp. 21–28, Jan. 1962.

[2] D. J. C. MacKay, "Good error-correcting codes based on very sparse matrices," *IEEE Trans. Inf. Theory.*, vol. 45, no. 2, pp. 399–431, Mar. 1999.

[3] Y. Kou, S. Lin, and M. Fossorier, "Low-density parity-check codes based on finite geometries: rediscovery and new results," *IEEE Trans. Inf. Theory.*, vol. 47, no. 7, pp. 2711–2736, Nov. 2001.

[4] K. Gulati, B. Evans, J. Andrews, and K. Tinsley, "Statistics of co-channel interference in a field of poisson and poisson-poisson clustered interferers," *IEEE Trans. Signal Process.*, vol. 58, no. 12, pp. 6207–6222, Dec. 2010.

[5] M. Chitre, "A high-frequency warm shallow water acoustic communications channel model and measurements," *J. Acoust. Soc.America.*, vol. 122, no. 5, pp. 2580–2586, Nov. 2007.

[6] P. Lopes, J. Pinto, and J. Gerald, "Dealing with unknown impedance and impulsive noise in the power-line communications channel," *IEEE Trans. Power Del.*, vol. 28, no. 1, pp. 58–66, Jan. 2013.

[7] T. W. et al., "Gradient descent bit flipping algorithms for decoding ldpc codes," *IEEE Trans. Commun.*, vol. 58, no. 6, pp. 1610–1614, Jun. 2010.

[8] J. Zhang and M. P. C. Fossorier, "A modified weighted bit-flipping decoding of low-density parity-check codes," *IEEE Commun. Lett.*, vol. 8, no. 3, pp. 165–167, Mar. 2004.

[9] F. Guo and L. Hanzo, "Reliability ratio based weighted bit-flipping decoding for low-density parity-check codes," *Electron. Lett.*, vol. 40, no. 21, pp. 1356–1358, Oct. 2004.

[10] a. E. B. G. Sundararajan, C. Winstead, "Noisy gradient descent bit-flip decoding for decoding ldpc codes," *IEEE Trans. Commun.*, vol. 62, no. 10, pp. 3385–3400, Oct. 2014.

[11] J. G. Gonzalez, "Robust techniques for wireless communications in non-gaussian environments," *Ph.D dossertatopm, University of Delaware, Newark,DE,USA*, 1997.

[12] V. Dimanche, A. Goupil, L. Clavier, and G. Gelle, "On detection method for soft iterative decoding in the presence of impulsive interference," *IEEE Commun. Lett.*, vol. 18, no. 6, pp. 945–948, Jun. 2014.

[13] D. J. C. Mackay, "Encyclopedia of sparse graph codes," http://www.inference.phy.cam.ac.uk/mackay/codes/data.html.