

A Study on Adaptive Thresholds for Reduced Complexity Bit-Flip Decoding

Julian Webber, Toshihiko Nishimura, Takeo Ohgane and Yasutaka Ogawa
Graduate School of Information Science and Technology, Hokkaido University, Japan
e-mail: julian.webber@m-icl.ist.hokudai.ac.jp

Abstract—Low-density parity-check (LDPC) codes are capacity approaching codes that have rapidly been adopted in modern systems such as the IEEE 802.11n and long term evolution advanced (LTE-A) communications standards. The decoders based on the iterative belief propagation offer exceedingly high performance but unfortunately have high computational complexity. Therefore significant research has focused on lower complexity architectures based on the family of so-called bit-flipping algorithms. In the basic bit-flipping algorithm the number of failed parity checks for each bit is calculated and the sign of the bit with the maximum failed checks is inverted. Inverting bits above a certain threshold removes the complexity of calculating a maximum function and adaptive thresholds on each bit further simplifies the design. The choice of threshold updates directly affects the error and convergence performances. Here we describe a simple architecture that has two decoders with different scaling factors and select the branch with the lowest syndrome sum. It is shown that the addition of a random uniform perturbation to the threshold can reduce the average iteration count further by providing an escape from stuck decoding states.

I. INTRODUCTION

It has been shown that near Shannon capacity limits [1] can be achieved with the soft-decision belief propagation algorithm (BPA) that computes the marginal probabilities at the nodes on factor graphs [2]. Optimal performance is achieved with long-length cycle-free codes and where the soft data at the decoder input is independent. The BPA however features a very high decoding complexity. At the other extreme of the complexity spectrum are the family of bit-flipping algorithms (BFA) that are aimed at low-power portable applications. The average number of iterations in the BFA is higher than with the BPA, but at each iteration a much smaller number of calculations are required, and the overall complexity is much lower.

In previous work we investigated the performance of an adaptive threshold decoder in additive white Gaussian noise channels [3]. In this paper we further the investigation measuring the performance with an additional LDPC code, algorithm structure and examine the effects of adding a small noise perturbation when a trapped search point is reached.

The layout of this paper is as follows. The BF class of algorithms are introduced in Section II. The adaptive thresholding methods together with a dual scaling model are discussed in Section III. Techniques to reduce the iteration count together with performance results are analyzed in Section IV. A discussion for further work is given in Section V and a summary follows in Section VI.

II. ALGORITHMS

The received sample vector is $\mathbf{y} = (y_1, y_2, \dots, y_N)$ and the hard decoded bits are given by $\mathbf{x} \in \{+1, -1\}$. The parity check matrix $\mathbf{H}$ is of size $M \times N$, where the number of checks is M and the number of bits is N . The non-zero elements in column j are given by $M(j)$ and the non-zero elements in row i of $\mathbf{H}$ are $N(i)$. The syndrome associated with the m -th check node is denoted as s_m or alternatively the bipolar syndrome is written as $\prod_{j \in N(i)} x_j$.

In the basic BFA the sign of a bit n with a maximum inversion function, E_n is inverted at each iteration. A weighted BF (WBF) algorithm was proposed by Kou et al. [4] that includes a term related to the energy of the least reliable bit involved in each check. This requires a search over the complete slot-length which contributes to an increased complexity and delay. The WBF is expressed as

$$E_n = \sum_{m \in M(n)} (2s_m - 1)y_m^{min}, \quad (1)$$

where y_m^{min} is the least reliable message node associated with the m -th check.

Given that two message bits have contributed to a failed parity check, it is the message with the highest energy $|y_j|$ that is most likely to be correct. In the reliability ratio algorithm [5] a normalization factor $R_{ij} = \beta \frac{|y_j|}{|y_i^{max}|}$ is computed where $|y_i^{max}|$ is the highest soft magnitude of all message nodes involved with the m -th check (Eq. 2). This normalization improves the performance over the WBF algorithm but the division operation in the initialization stage contributes to an elevated complexity. An implementation-efficient reliability ratio WBF (IRRWBF) algorithm was later proposed which reduced the complexity particularly when the iteration count was low and the code length was large [6].

$$E_n = \sum_{i \in M(n)} (2s_m - 1)/R_{ij}. \quad (2)$$

In addition to the syndrome sum the gradient descent bit flipping (GDBF) algorithm [7] incorporates a correlation term between the initial received soft and hard decision which should be large and positive for a correctly estimated bit [7].

$$\Delta_n^{GD}(\mathbf{x}) = \sum_{i \in M(n)} \prod_{j \in N(i)} x_j + x_n y_n. \quad (3)$$

A bit is inverted if $\Delta_n^{GD}(\mathbf{x})$ is less than a global threshold. An objective function indicates the state of the algorithm at each iteration and is defined as

$$f^{GD}(\mathbf{x}) = \sum_{i \in M(n)} \prod_{j \in N(i)} x_j + \sum_{n=1}^N x_n y_n. \quad (4)$$

By monitoring the objective function it was proposed to switch from a large step-size (multiple bit-flips) to a small step-size if a stuck decoding state was detected. On such occasions a single small-descent of the objective function using a lower set threshold plus a random component was also shown experimentally to improve the performance.

III. ADAPTIVE THRESHOLDS

Adaptive thresholds were recently investigated by Cho [8] for codes with a large number of checks per bit. A ‘flipping gain’ for the BFA was calculated as the benefit obtained from flipping erroneous bits minus the loss from flipping correct bits. A frame was subjected to a known (monotonically increasing) number of errors. The threshold range p that provided a flipping gain greater than one was computed. It was found through simulation that there was an optimal threshold that resulted in the least number of iterations. It was conjectured that similar analysis could be applied for codes with a smaller number of checks per bit, (such as $M=3$ investigated in this paper), using non-integer thresholds (e.g. 2.3) and using a probabilistic strategy (e.g. thresholds 2.0, 2.0, 3.0).

A novel adaptive threshold bit flipping algorithm was proposed in [9], where an inversion threshold, λ_n was set on each bit. A bit was flipped if the inversion function was below the particular bit-threshold (5), and otherwise the threshold was lowered.

$$E_n(k) \leq \lambda_n(k). \quad (5)$$

The updated threshold is then written as (6)

$$\lambda_n(k+1) = \theta \lambda_n(k), \quad (6)$$

where k is the iteration and θ is a threshold scaling parameter. Using this method the algorithm moves automatically from multiple to a single bit-flip as the threshold is successively lowered. The hardware complexity is reduced as each bit-processing operation (i.e. scaling/inversion) is computed independently of other bits. To eliminate the need for a physical hardware multiplier, a scale factor was chosen to be proportional to a power of two (Fig. 1).

A. Scaling factor

In this work, the scaling factor θ is expressed as the sum of two or less fractional components :

$$\theta = 2^{-d_A} + 2^{-d_B}, \quad (7)$$

where d_A and d_B are integer bit-shifts. Typical values of θ with the corresponding shifts are listed in TABLE I. As an example, to obtain the threshold update $\lambda_n(k+1) = 0.75\lambda_n(k)$, the current threshold $\lambda_n(k)$ is shifted once 1 place (i.e. 0.5),

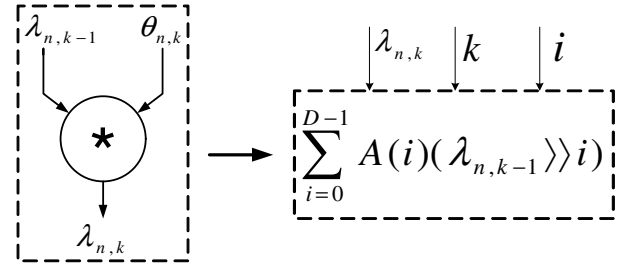


Fig. 1. Updating the threshold λ on each bit n using a scaling factor obtained by summing a number of shifted versions of the current threshold. k =iteration, i =shift, $\mathbf{A} = [a(0), \dots, a(D-1)] \in \{+1, -1\}$

TABLE I
THRESHOLD SCALING VALUES, θ AND PARTIAL COMPONENTS.

θ (decimal)	d_A	d_B	θ_A	θ_B
0.5	1	0	1/2	0
0.625	1	3	1/2	+1/8
0.75	1	2	1/2	+1/4
0.875	0	3	1	-1/8
0.969	0	5	1	-1/32

TABLE II
LDPC CODE PROPERTIES.

Code	Name [11]	Rows	Cols	Col deg.
1	PEGReg504x1008	504	1008	3
2	PEGReg252x504	252	504	3

once 2 places (i.e. 0.25), and the two partial products are summed. Clearly, a small value of θ (e.g. 0.5) corresponds to a large step-size and a large value of θ (e.g. 0.969) equates to a small step-size. Any scaling factor can be created by the appropriate summation of smaller factors, though two additions are chosen here to limit the complexity.

B. Single scaling-factor

Two progressive edge growth (PEG) regular LDPC codes are studied in this paper with their properties tabulated in Table II. The PEG technique aims to maximize the length of the shortest cycle (i.e. girth) and therefore provides good quality codes [10]. A snapshot of the inversion function E_n in a random trial after 60 iterations is plotted in Fig. 2. The bits with the highest value of E_n above the threshold are flipped first. Once a bit is inverted, a change in the value of the inversion function occurs.

C. Dual scaling-factor

A dual-scaling switching algorithm (DSA) is investigated that uses joint fine and coarse scaling step-size decoders. The bits from the decode-branch that have the minimum syndrome sum are passed onto the next iteration (Fig. 3). The algorithm is summarized as

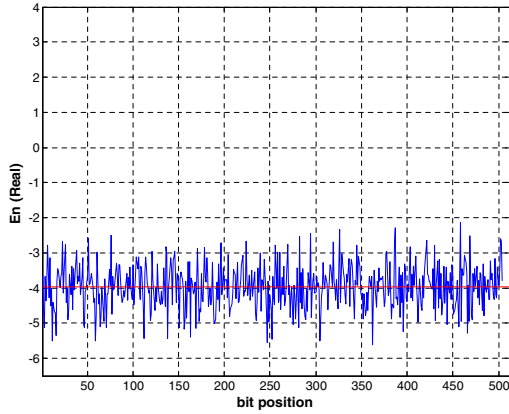


Fig. 2. Plot of the inversion function E_n of each bit at iteration 60 ($\theta=0.97$). The red horizontal line indicates the mean value.

STEP 1- Initialize the inversion thresholds $\lambda_n = \lambda_0, \forall n$. Set State=1.
STEP 2- Calculate the inversion function E_n , for $m = 1, 2, \dots, M$.
STEP 3- If $E_n(\theta_2) \geq \lambda_n$, Invert bit n . Else $\lambda_n = \theta_2 \lambda_{n-1}$. Optionally add perturbation to λ_n .
STEP 4- If state=1, Repeat STEP 3 for $\theta = \theta_1$.
STEP 5- If $\sum s_m(\theta_1) \geq \sum s_m(\theta_2)$, Set state=2; $\lambda_n = \lambda_n(\theta_2)$, $\forall n$ and $\mathbf{x} = \mathbf{x}(\theta_2)$; Deactivate branch no.1 to reduce power consumption. Else $\lambda_n = \lambda_n(\theta_1)$, $\forall n$ and $\mathbf{x} = \mathbf{x}(\theta_1)$.
STEP 6- Terminate if $\sum s_m = 0$ or maximum iterations. Else Goto STEP 2.

The number of failed checks versus iteration of the DSA is plotted in Fig. 4. The solid line (marked ‘.’) shows the syndrome sum when $\theta_1=0.5$ and the solid line (marked ‘+’) shows the syndrome sum when $\theta_2=0.97$.

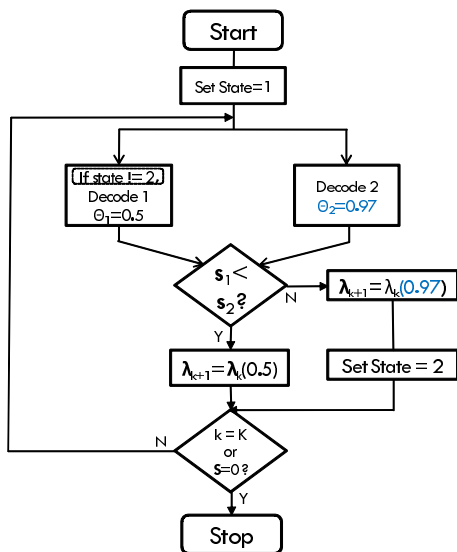


Fig. 3. Simplified flow chart of the dual scaling threshold algorithm.

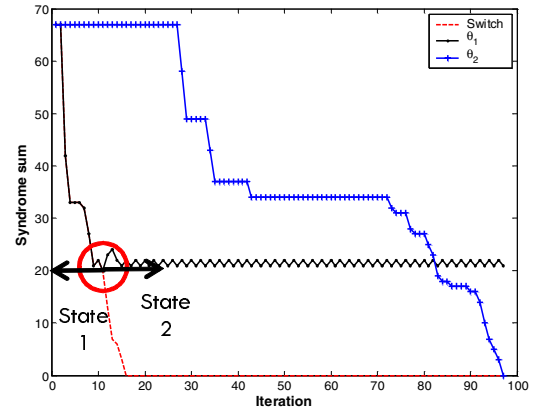


Fig. 4. Plot of the syndrome sum against iteration. Single scaling factor $\theta_1=0.5$ and $\theta_2=0.97$. ‘Switch’ indicates the dual branch decoder.

when $\theta_2=0.969$. The number of failed checks decreased to zero by iteration 98 using the small step-size. However, by switching at iteration 11 from $\theta_1=0.5$ to $\theta_2=0.969$ when the syndrome sum was lower on that branch, the number of failed checks were zero on the 18th iteration. Although the combined complexity is increased it can be capped by deactivating the θ_1 branch once the θ_2 branch produces less errors. The ideal factor is a function of the SNR, modulation and particular code characteristics which makes computing an optimum value computationally complex and an offline statistical approach through experiment trial was taken in [9]. An additional benefit of the dual scaling detector proposed here is that a real-time or offline threshold calculation is avoided. A three-scaling factor detector was also evaluated but only a small reduction in iteration count over the dual-scaling factor decoder was achieved with the particular codes and hence, due to the additional complexity, maybe considered less worthwhile.

In a low-complexity early-stopping algorithm we halted further processing if $f(k)=f(k-10)$ or $f(k-11)$. The algorithm was more stable by using the two conditions in case $f(k)$ oscillates between two values such as seen by the black zig-zag line in Fig. 4.

D. Addition of random perturbation to threshold

One of the features of the gradient descent algorithm concerns that of trapped search points where the decoded sequence differs from the transmitted codeword. We investigate the idea of adding a small random perturbation to the threshold of each bit if the syndrome sum does not change over a sliding window of W samples. When this condition is satisfied a new random sample z_n is added to the current value of E_n as shown by

$$\lambda_n = \lambda_n + cz_n, \quad (8)$$

where z_n is a uniform random sample for bit n and c is a small constant typically between 0.1 and 0.4.

The instantaneous effect on the syndrome sum of adding the noise can be seen in Fig. 5. Here, noise was injected

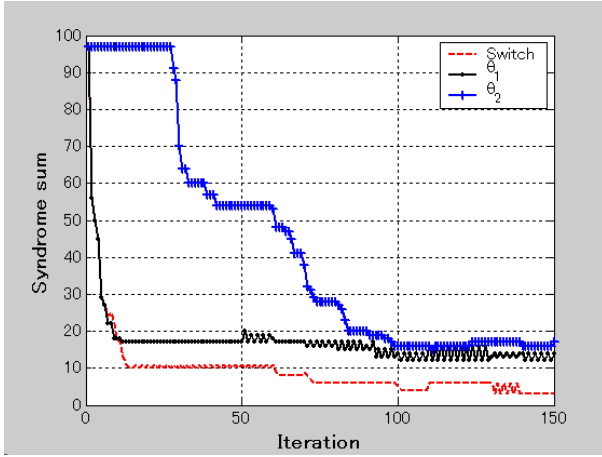


Fig. 5. Plot of the syndrome sum against iteration with noise perturbation at iterations: 50, 60, ... 150.

at iteration 50 and at intervals of 10 iterations thereafter. At iterations 60, 70 and 100 we observe a small but noticeable lowering in the number of syndrome errors. Due to the random nature of the noise addition, there will be occasions when there is no improvement such as at iteration 110 for example. In the BER evaluation the window size, W is set at size 10 samples. The value of the inversion function taken from one decoding trial E_n in the absence of a random perturbation is plotted in Fig. 6. Using the same received signal sequence $\mathbf{y}$, the value of E_n when adding the noise is plotted in Fig. 7. The benefit of the the random perturbation can clearly be seen where a stuck decoding state was successfully exited and the syndrome sum reduced to zero by iteration 42. The particular bits whose inversion functions are nearest to the threshold are more likely to be inverted and by setting the noise power low only a small number of bits are additionally flipped that would not have been otherwise.

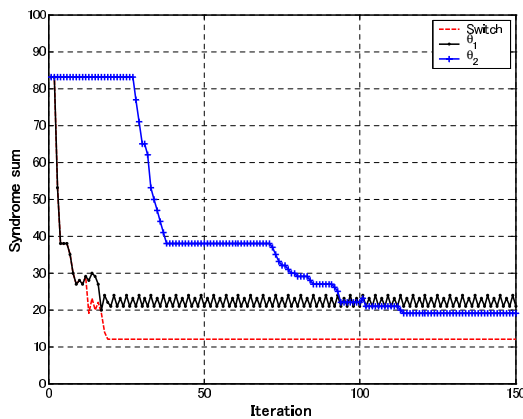


Fig. 6. Syndrome sum without added random perturbation.

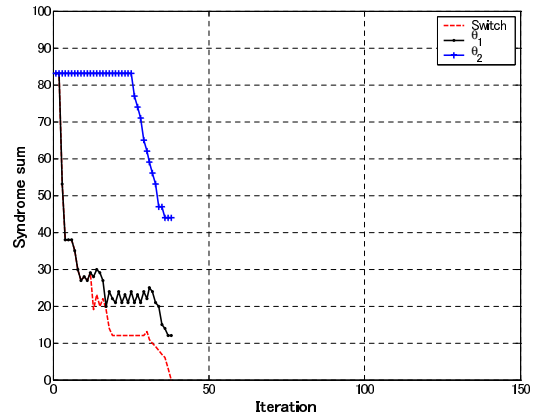


Fig. 7. Syndrome sum with perturbation. This trial used the exact same received soft data, $\mathbf{y}$ as in Fig. 6 for fairness.

IV. BIT ERROR AND COMPLEXITY PERFORMANCES

The decoder performance in an AWGN channel was evaluated through computer simulation. The maximum number of iterations was set at 150 and the modulation was BPSK. Although the IRRWBF algorithm performed the best it has the highest complexity. The BER depends on the particular value of θ as shown in Fig. 8. When $\theta=0.5$, more bits are inverted on average per cycle and the overall performance is worse than when $\theta=0.97$. The DSA with $\theta=0.625/0.97$ performed equally as well as the one with a single $\theta=0.97$ but its advantage is the reduced iteration count as seen in Fig. 9 where the average iteration count is reduced from 66 to 39 at 4 dB SNR. Similarly, the number of iterations of the 'Dual early-stop' algorithm was reduced to 17 at an E_b/N_0 of 4 dB.

The results with the random perturbation using $c=0.25$ are denoted by '+ perturb.'. At the BER of 10^{-3} , an improvement of 0.5 dB was obtained for the dual scaling detector with perturbation (Fig. 10). As the fine step-size decoder slowly converges to the desired solution there is only a small improvement in BER. However, at 3 dB E_b/N_0 the average number of iterations was reduced by (16.4, 9.2, 2.0) for the (dual-scale, $\theta=0.63$ and $\theta=0.97$) detectors respectively (Fig.11). At 4 dB E_b/N_0 the corresponding savings were (11.0, 10.1, 2.6) iterations. An optimum value for c and the window length W can be found experimentally and is an area for future work.

V. FURTHER WORK

We are investigating adaptive threshold architectures for the multiple-input multiple-output (MIMO) fading channel with spatial multiplexing. The MIMO architecture depends on whether the LDPC coding is applied across the entire data frame or independently over each substream. An architecture where the scaling factor step-size is a function of the relative eigenvalue on each sub-channel is being evaluated.

VI. SUMMARY

The error and convergence rates of adaptive threshold bit-flipping algorithms have been studied. A dual-scaling archi-

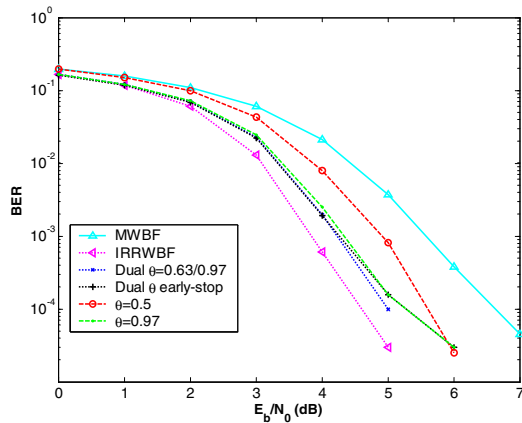


Fig. 8. BER versus E_b/N_0 (dB) with LDPC Code 1 in AWGN.

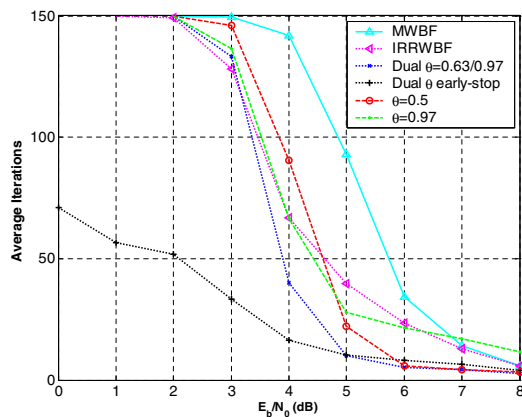


Fig. 9. Average iteration count versus E_b/N_0 (dB) for Code 1.

tecture that uses the bit estimates and thresholds from either a fine or coarse step-size decode branch depending on the syndrome sum at each iteration was proposed. The dual scale architecture with $\theta=0.63/0.97$ performed as well as that of the single step-size $\theta=0.97$ decoder but the average number of iterations was cut from 66 to 39 at 4 dB E_b/N_0 in a SISO channel. By adding a small noise perturbation the iteration count of the dual-scale detector was reduced by 11 cycles at 4 dB E_b/N_0 . An extension of the work for MIMO channels and OFDM systems is part of our future work.

REFERENCES

- [1] S. Chung, G. D. Forney Jr., T. Richardson, and R. Urbanke, "On the design of low-density parity-check codes within 0.0045 dB of the Shannon limit," *IEEE Commun. Letts.*, vol. 5, no. 2, pp. 58-60, Feb. 2001.
- [2] W. Chung, and J. Cruz, "An improved belief-propagation decoder for LDPC-coded partial-response channels," *IEEE Trans. Magnetics*, vol. 46, no. 7, pp. 2639-2648, Jul. 2010.
- [3] J. Webber, T. Nishimura, T. Ohgane, and Y. Ogawa, "Investigation of simple threshold shifting for bit-flip decoding," *Proceedings of the 14th International Symposium on Wireless Personal Multimedia Communications (WPMC'11)*, Brest, France, Oct. 2011.
- [4] Y. Kou, S. Lin, and M. Fossorier, "Low-density parity-check codes based on finite geometries: a rediscovery and new results," *IEEE Trans. Inform. Theory*, vol. 47, no. 7, pp. 2711-2736, Nov. 2001.

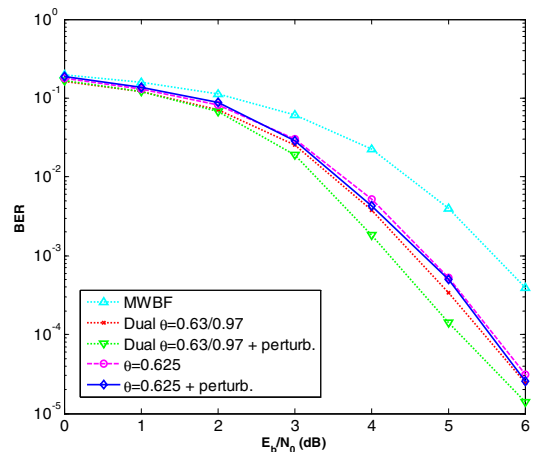


Fig. 10. BER versus E_b/N_0 (dB) for LDPC Code 2 with noise perturbation in AWGN.

- [5] F. Guo and L. Hanzo, "Reliability ratio based weighted bit-flipping decoding for LDPC codes," *IEE Electron. Lett.*, vol. 40, no. 21, pp. 1356-1358, Oct. 2004.
- [6] C. H. Lee and W. Wolf, "Implementation-efficient reliability ratio based weighted bit-flipping decoding for LDPC codes," *IEEE Electronics Lett.*, vol. 41, no. 13, pp. 755-757, Jun. 2005.
- [7] T. Wadayama, K. Nakamura, M. Yagita, Y. Funahashi, S. Usami, and I. Takumi, "Gradient descent bit-flipping algorithms for decoding LDPC codes," *International Symposium on Information Theory and its Applications (ISITA'08)*, Auckland, New Zealand, Dec. 2008.
- [8] J. Cho and W. Sung, "Adaptive threshold technique for bit-flipping decoding of low-density parity-check codes," *IEEE Comm. Letts.*, pp. 857-859, col. 14, no. 9, Sep. 2010.
- [9] M. Ismail, I. Ahmed, J. Coon, S. Armour, T. Kocak, and J.P. McGeehan, "Low latency low power bit flipping algorithms for LDPC decoding," *IEEE PIMRC10*, Sep. 2010.
- [10] E. Psota, and L. Perez, "Iterative construction of regular LDPC codes from independent tree-based minimum distance bounds," *IEEE Comm. Letts.*, vol. 15, no. 3, Mar. 2011.
- [11] D. MacKay, "Encyclopedia of sparse graph codes [Online]," URL: <http://www.inference.phy.cam.ac.uk/mackay/codes/data.html>, Accessed Feb. 2011.

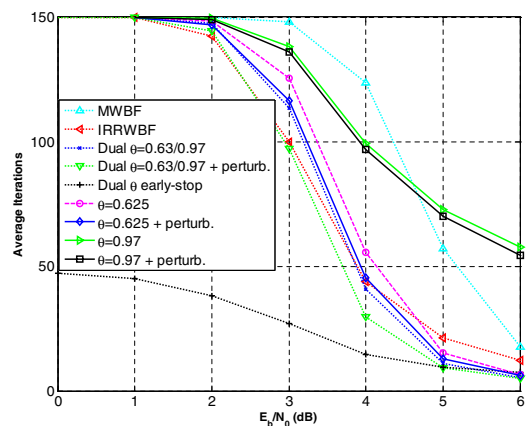


Fig. 11. Average iteration count versus E_b/N_0 (dB) for Code 2 with perturbation.