Unconventional Behavior of the Noisy Min-Sum Decoder over the Binary Symmetric Channel

Christiane L. Kameni Ngassa*,#, Valentin Savin*, David Declercq#
*CEA-LETI, Minatec Campus, Grenoble, France, {christiane. kameningassa, valentin.savin}@cea.fr
#ETIS, ENSEA / CNRS UMR-8051 / Univ. Cergy-Pontoise, France, declercq@ensea.fr

***Abstract*— This paper investigates the behavior of the noisy Min-Sum decoder over binary symmetric channels. A noisy decoder is a decoder running on a noisy device, which may introduce errors during the decoding process. We show that in some particular cases, the noise introduce by the device can help the Min-Sum decoder to escape from fixed points attractors, and may actually result in an increased correction capacity with respect to the noiseless decoder. We also reveal the existence of a specific threshold phenomenon, referred to as functional threshold. The behavior of the noisy decoder is demonstrated in the asymptotic limit of the code-length, by using "noisy" density evolution equations, and it is also verified in the finite-length case by Monte-Carlo simulation.**

I. Introduction

In traditional models of communication or storage systems with error correction coding, it is assumed that the operations of an error correction encoder and decoder are deterministic and that the randomness exists only in the transmission or storage channel. However, with the advent of nanoelectronics, the reliability of the forthcoming circuits and computation devices is becoming questionable. It is then becoming crucial to design and analyze error correcting decoders able to provide reliable error correction even if they are made of unreliable components.

Over the last years, the study of error correcting decoders, especially Low-Density Parity-Check (LDPC) decoders, running on noisy hardware attracted more and more interest in the coding community. In [1], [2], analytical methods have been proposed to evaluate the performance of one step majority logic LDPC decoders constructed from faulty gates. In [3] and [4] hardware redundancy is used to develop fault-compensation techniques, able to protect the decoder against the errors induced by the noisy components of the circuit. In [5], a class of modified Turbo and LDPC decoders has been proposed, able to deal with the noise induced by the failures of a low-power buffering memory that stores the input soft bits of the decoder. Very recently, the characterization of the effect of noisy processing on message-passing iterative LDPC decoders has been proposed. In [6], the concentration and convergence properties were proved for the asymptotic performance of noisy message-passing decoders, and density evolution equations were derived for the noisy Gallager-A and Belief-Propagation (BP) decoders. In [7]–[9], the authors investigated the asymptotic behavior of the noisy Gallager-B decoder defined over binary and non-binary alphabets. In all these papers the noisy implementation of the iterative message passing decoders was emulated by passing each of the exchanged messages through a noisy channel.

This work was supported by the Seventh Framework Programme of the European Union, under Grant Agreement number 309129 (*i*-RISC project).

In our previous works [10], [11], we proposed various error models for the arithmetic components of the Min-Sum (MS) decoder and derived density evolution equations for the noisy MS. In this paper, we further investigate the asymptotic and finite-length behavior of the noisy MS decoder over the Binary Symmetric Chanel (BSC). We refine the probabilistic error models for the noisy MS decoder we used previously in [10], [11], and discuss their symmetry properties. We conduct a thorough asymptotic analysis of the noisy MS decoder and highlights a wide variety of more or less conventional behaviors. We also reveal the existence of a specific threshold phenomenon, which is referred to as *functional threshold*. Finally, the asymptotic results are also corroborated through finite length simulations.

The paper is organized as follows. Section II discusses the probabilistic error models and and explains the motivations behind them. The noisy MS decoder is also introduced in this section. Section III shortly discusses the density evolution for the noisy MS decoder, and provides the notation and definitions required to understand the asymptotic analysis of the noisy MS decoder, which is then conducted in Section IV. Finally, Section V corroborates the asymptotic analysis through finite-length simulations, and Section VI concludes the paper.

II. Probabilistic Error Models for the Noisy Min-Sum Decoder

A. Noisy Message-Passing Decoders

The model for noisy Message-Passing (MP) decoders proposed in [6] incorporates two different sources of noise: *computation noise* due to noisy logic in the processing units, and *message-passing noise* due to noisy wires (or noisy memories) used to exchange messages between neighbor nodes.

The computation noise is modeled as a random variable, which the variable-node or the check-node processing depends on. Put differently, an outgoing message from a (variable or check) node depends not only on the incoming messages to that node, but also on the realization of a random variable, which is assumed to be independent of the incoming messages.

The message-passing noise is simply modeled as a noisy channel. Hence, transmitting a message over a noisy wire is emulated by passing that message through the corresponding noisy channel.

However, in [6] it has been noted that *"there is no essential loss of generality by combining computation noise and message-passing noise into a single form of noise"*. Consequently, the approach adopted has been to merge computation noisy into message-passing noise, and to emulate noisy decoders by passing the exchanged messages through different noisy channel models. Thus, the noisy Gallager-A decoder has been emulated by passing the exchanged messages over independent and identical BSC wires, while the noisy BP decoder has been emulated by corrupting the exchanged messages with bounded and symmetrically distributed additive noise (*e.g.* uniform noise or truncated Gaussian noise).

The approach we follow in this work differs from the one in [6] in that the computation noise is modeled at the lower level of *arithmetic and logic operations* that compose the variable-node and check-node processing units. This finer-grained noise modeling is aimed at determining the level of noise that can be tolerated in each type of operation. As the main focus of this work is on computation noise, we shall consider that messages are exchanged between neighbor nodes through error-free wires (or memories). However, we note that *this work can readily be extended to include different error models for the message-passing noise* (as defined in [6]). Alternatively, we may assume that the message-passing noise is merged into the computation noise, in the sense that adding noise in wires would modify the probabilistic model of the noisy logic or arithmetic operations.

B. Probabilistic Error Models for Noisy Adders, Comparators and XOR-gates

In this section we describe the probabilistic models for noisy adders, comparators and xor-operators, that will be used in the next section, in order to emulate the noisy implementation of the finite-precision MS decoder.

1) Noisy Adder Model: We consider a θ-bit adder ($\theta \geq 2$). The inputs and the output of the adder are assumed to be in $\mathcal{V} = \{-\Theta, \ldots, -1, 0, +1, \ldots, +\Theta\}$, where $\Theta = 2^{\theta-1}-1$. For inputs $(x, y) \in \mathcal{V}$, the output of the *noiseless* θ-bit adder is given by $v = s_{\mathcal{V}}(x+y)$, where $\mathbf{s}_{\mathcal{V}} : \mathbb{Z} \to \mathcal{V}$ denotes the θ-bit saturation map:

$$\mathbf{s}_{\mathcal{V}}(z) = \mathrm{sgn}(z) \cdot \min(|z|, \Theta) \tag{1}$$

The output of the noisy adder will be defined by *injecting errors* in the output of the noiseless one. Two main error injection models will be used in this work, both of which are based on a bitwise XOR operation between the noiseless output v and an error e. The error e is assumed to be drawn from an *error set* $\mathcal{E} \subseteq \mathcal{V}$, according to an *error probability distribution* $p_{\mathcal{E}} : \mathcal{E} \to [0, 1]$. The two models differ in the definition of the error set $\mathcal{E}$, which is chosen such that the bitwise XOR operation (*i.e.* the error injection) may or may not affect the sign of the noiseless output. In the first case the error injection model is said to be *full-depth*, while in the second it is said to be *sign-preserving*.

We fix a *signed number binary representation*, which can be any of the *sign-magnitude*, *one's complement*, or *two's complement* representation. There are exactly 2^θ signed numbers that can be represented by θ bits in any of the above formats, one of which does not belong to $\mathcal{V}$ (note that $\mathcal{V}$ contains only $2\Theta+1 = 2^\theta - 1$ elements for symmetry reasons!). We denote this element by ζ. For instance, in two's complement format, $\zeta = -(\Theta+1)$, with binary representation $10\cdots0$.

Full-depth error injection: For this error model the error set is $\mathcal{E} = \mathcal{V}$. For symmetry reasons, all errors $e \neq 0$ are assumed to occur with the same probability. It follows that $p_{\mathcal{E}}(0) = 1 - p_a$ and $p_{\mathcal{E}}(e) = \frac{p_a}{2\Theta}$, $\forall e \neq 0$, where $p_a > 0$ is referred to as the error injection probability. Finally, the error injection function is defined by:

$$\imath(v, e) = \begin{cases} v \wedge e, & \text{if } v \wedge e \in \mathcal{V} \\ e, & \text{if } v \wedge e = \zeta \end{cases} \tag{2}$$

Sign-preserving error injection: For this error model the error set is $\mathcal{E} = \{0, +1, \ldots, +\Theta\}$. The error injection probability is denoted by p_a, and all errors $e \neq 0$ are assumed to occur with the same probability (for symmetry reasons). It follows that $p_{\mathcal{E}}(0) = 1 - p_a$ and $p_{\mathcal{E}}(e) = \frac{p_a}{\Theta}$, $\forall e \neq 0$. Finally, the error injection function is defined by:

$$\imath(v, e) = \begin{cases} v \wedge e, & \text{if } v \neq 0 \text{ and } v \wedge e \in \mathcal{V} \\ \pm e, & \text{if } v = 0 \\ 0, & \text{if } v \wedge e = \zeta \end{cases} \tag{3}$$

In the above definition, $\imath(0, e)$ is randomly set to either $-e$ or $+e$, with equal probability (this is due once again to symmetry reasons). Note also that the last two conditions, namely $v = 0$ and $v \wedge e = \zeta$, cannot hold simultaneously (since $e \neq \zeta$).

Finally, both of the above error injection models satisfy the following *symmetry condition*:

$$\sum_{\{e \,|\, \imath(v,e)=w\}} p_{\mathcal{E}}(e) \;=\; \sum_{\{e \,|\, \imath(-v,e)=-w\}} p_{\mathcal{E}}(e), \quad \forall v, w \in \mathcal{V} \tag{4}$$

A particular case in which the symmetry condition is fulfilled is when $\imath(-v, e) = -\imath(v, e)$, for all $v \in \mathcal{V}$ and $e \in \mathcal{E}$. In this case, the error injection model is said to be *highly symmetric*. We note that both of the above models are highly symmetric, if one of the sign-magnitude or the one's complement representation is used. In case that the two's complement representation is used, they are both symmetric, but not highly symmetric.

Remark: It is also possible to define a *variable depth error injection* model, in which errors are injected in only the λ least significant bits, with $\lambda \leq \theta$ [10]. Hence, $\lambda = \theta$ corresponds to the above full-depth model, while $\lambda = \theta - 1$ corresponds to the sign-preserving model. However, for $\lambda < \theta - 1$ such a model will **not** be symmetric, if the the two's complement representation is used!

Finally, for any of the above error injection models, the output of the noisy adder is given by:

$$\mathbf{a}_{\mathrm{pr}}(x, y) = \imath\left(\mathbf{s}_{\mathcal{V}}(x+y), e\right), \tag{5}$$

where e is drawn randomly from $\mathcal{E}$ according to the probability distribution $p_{\mathcal{E}}$. The *error probability of the noisy adder*, *i.e.*

$\Pr\left(\mathbf{a}_{\text{pr}}(x,y) \neq \mathbf{s}_{\mathcal{V}}(x+y)\right)$, assuming uniformly distributed inputs, equals the error injection probability parameter p_a.

Obviously, it would be possible to define more general error injection models, in which the injected error depends on the data (currently and/or previously) processed by the adder. Such an error injection model would certainly be more realistic, but it would also make very difficult to analytically characterize the behavior on noisy MP decoders. As a side effect, the decoding error probability would be dependent on the transmitted codeword, which would prevent the use of the *density evolution* technique for the analysis of the asymptotic decoding performance (since the density evolution technique relies on the all-zero codeword assumption).

2) Noisy Comparator Model: Let $\mathbf{lt}$ denote the noiseless *less than* operator, defined by $\mathbf{lt}(x,y) = 1$ if $x < y$, and $\mathbf{lt}(x,y) = 0$ otherwise. The *noisy less than* operator, denoted by $\mathbf{lt}_{\text{pr}}$, is defined by flipping the output of the noiseless $\mathbf{lt}$ operator, with some probability value that will be denoted in the sequel by p_c.

Finally, the *noisy minimum* operator is defined by:

$$\mathbf{m}_{\text{pr}}(x,y) = \begin{cases} x, & \text{if } \mathbf{lt}_{\text{pr}}(x,y) = 1 \\ y, & \text{if } \mathbf{lt}_{\text{pr}}(x,y) = 0 \end{cases} \tag{6}$$

3) Noisy XOR Model: The noisy XOR operator, denoted by $\mathbf{x}_{\text{pr}}$, is defined by flipping the output of the noiseless operator with some probability value, which will be denoted in the sequel by p_x. It follows that:

$$\mathbf{x}_{\text{pr}}(x,y) = \begin{cases} x \wedge y, & \text{with probability } 1-p_x \\ \overline{x \wedge y}, & \text{with probability } p_x \end{cases} \tag{7}$$

Assumption: We further assume that the inputs and the output of the XOR operator may take values in either $\{0,1\}$ or $\{-1,+1\}$ (using the usual 0,1 to ± 1 conversion). This assumption will be implicitly made throughout the paper.

C. Nested Operators

For the Min-Sum decoding, several arithmetic/logic operations must be *nested*[1] in order to compute the exchanged messages. Since all these operations (additions, comparisons, XOR) are commutative, the way they are nested does not have any impact on the infinite-precision MS decoding. However, this is no longer true for finite-precision decoding, especially in case of noisy operations, and one needs an assumption about how the above operators extend from two to more inputs.

Our assumption is the following. For $n \geq 2$ inputs, we randomly pick any two inputs and apply the operator on this pair. Then we replace the pair by the obtained output, and repeat the above procedure until there is only one output (and no more inputs) left. The formal definition goes as follows. Let $\Omega \subset \mathbb{Z}$ and $\omega : \Omega \times \Omega \to \Omega$ be a noiseless or noisy operator with two operands. Let $\{x_i\}_{i=1:n} \subset \Omega$ be an unordered set of n operands. We define:

$$\omega\left(\{x_i\}_{i=1:n}\right) = \omega(\cdots(\omega(x_{\pi(1)}, x_{\pi(2)}), \cdots), x_{\pi(n)}), \tag{8}$$

where π is a random permutation of $1,\ldots,n$.

[1]For instance, $(d_n - 1)$ additions – where d_n denotes the degree of the variable-node n – are required in order to compute each $\alpha_{m,n}$ message.

Algorithm 1 Noisy Min-Sum (Noisy-MS) decoding

Input: $\underline{\boldsymbol{y}} = (y_1,\ldots,y_N) \in \mathcal{Y}^N$ ▷ *received word*
Output: $\underline{\hat{\boldsymbol{x}}} = (\hat{x}_1,\ldots,\hat{x}_N) \in \{-1,+1\}^N$ ▷ *estimated codeword*

Initialization
  for all $n = 1,\ldots,N$ **do** $\gamma_n = \mathbf{q}(y_n)$;
  for all $n = 1,\ldots,N$ and $m \in \mathcal{H}(n)$ **do** $\alpha_{m,n} = \gamma_n$;
Iteration Loop
  for all $m = 1,\ldots,M$ and $n \in \mathcal{H}(m)$ **do** ▷ *CN-processing*
    $\beta_{m,n} = \mathbf{x}_{\text{pr}}\left(\{\text{sgn}(\alpha_{m,n'})\}_{n' \in \mathcal{H}(m)\setminus n}\right)$
      $\cdot \mathbf{m}_{\text{pr}}\left(\{|\alpha_{m,n'}|\}_{n' \in \mathcal{H}(m)\setminus n}\right)$;
  for all $n = 1,\ldots,N$ and $m \in \mathcal{H}(n)$ **do** ▷ *VN-processing*
    $\alpha_{m,n} = \mathbf{a}_{\text{pr}}\left(\{\gamma_n\} \cup \{\beta_{m',n}\}_{m' \in \mathcal{H}(n)\setminus m}\right)$;
    $\alpha_{m,n} = \mathbf{s}_{\mathcal{M}}\left(\alpha_{m,n}\right)$;
  for all $n = 1,\ldots,N$ **do** ▷ *AP-update*
    $\tilde{\gamma}_n = \mathbf{a}_{\text{pr}}\left(\{\gamma_n\} \cup \{\beta_{m,n}\}_{m \in \mathcal{H}(n)}\right)$;
  for all $\{v_n\}_{n=1,\ldots,N}$ **do** $\hat{x}_n = \text{sgn}(\tilde{\gamma}_n)$; ▷ *hard decision*
  if $\underline{\hat{\boldsymbol{x}}}$ is a codeword **then** exit iteration loop ▷ *syndrome check*
End Iteration Loop

D. Noisy Min-Sum Decoder

We consider a finite-precision MS decoder, in which the *a priori information* (γ_n) and the *exchanged extrinsic messages* ($\alpha_{m,n}$ and $\beta_{m,n}$) are quantized on q bits. The *a posteriori information* ($\tilde{\gamma}_n$) is quantized on $\tilde{q}$ bits, with $\tilde{q} > q$ (usually $\tilde{q} = q+1$, or $\tilde{q} = q+2$). We also denote by $\mathcal{M}$ the alphabet of the a priori information and of the extrinsic messages, and by $\widetilde{\mathcal{M}}$ the alphabet of the a posteriori information. Thus:

- $\mathcal{M} = \{-Q,\ldots,-1,0,+1,\ldots,Q\}$, where $Q = 2^{q-1}-1$;
- $\widetilde{\mathcal{M}} = \{-\widetilde{Q},\ldots,-1,0,+1,\ldots,\widetilde{Q}\}$, where $\widetilde{Q} = 2^{\tilde{q}-1}-1$.

We further consider a *quantization map* $\mathbf{q} : \mathcal{Y} \to \mathcal{M}$, where $\mathcal{Y}$ denotes the channel output alphabet. The quantization map $\mathbf{q}$ determines the q-bit quantization of the decoder soft input.

The noisy finite-precision MS decoder is presented in Algorithm 1. We assume that $\tilde{q}$-bit adders are used to compute both $\alpha_{m,n}$ messages in the **VN-processing** step, and $\tilde{\gamma}_n$ values in the **AP-update** processing step. This is usually the case in practical implementations, and allows us to use the same type of adder in both processing steps. This assumption explains as well the q-bit saturation of $\alpha_{m,n}$ messages in the **VN-processing** step. Note also that the saturation of $\tilde{\gamma}_n$ values is actually done within the adder (see Equation (5)).

Finally, we note that the *hard decision* and the *syndrome check* steps in Algorithm 1 are assumed to be *noiseless*. We note however that the syndrome check step is optional, and if missing, the decoder stops when the maximum number of iterations is reached.

E. Sign-Preserving Properties

Let $\mathbf{U}$ denote any of the VN-processing or CN-processing units of the noiseless MS decoder. We denote by $\mathbf{U}_{\text{pr}}$ the corresponding unit of the noisy MS decoder. We say that $\mathbf{U}_{\text{pr}}$ is *sign-preserving* if for any incoming messages and any noise realization, the outgoing message is of the same sign as the message obtained when the same incoming messages are supplied to $\mathbf{U}$.

Clearly, CN_{pr} is sign-preserving if and only if the XOR-operator is noiseless ($p_x = 0$). In case that the noisy XOR-operator severely degrades the decoder performance, it is possible to increase its reliability by using classical fault-tolerant techniques (as for instance modular redundancy, or multi-voltage design by increasing the supply voltage of the corresponding XOR-gate). The price to pay, when compared to the size or the energy consumption of the whole circuit, would be reasonable.

Concerning the VN-processing, it is worth noting that the VN_{pr} is **not** sign-preserving, even if the noisy adder is so. This is due to the fact that multiple adders must be "nested" in order to complete the VN-processing. However, a sign-preserving adder might have several benefits. First, the error probability of the sign of variable-node messages would be lowered, which would certainly help the decoder. Second, if the noisy adder is sign-preserving and all the variable-node incoming messages have the same sign, then the VN_{pr} does preserve the sign of the outgoing message. Put differently, in case that all the incoming messages agree on the same hard decision, the noisy VN-processing may change the confidence level, but cannot change the decision. This may be particularly useful, especially during the last decoding iterations.

Finally, the motivation behind the sign-preserving noisy adder model is to investigate its possible benefits on the decoder performance. If the benefits are worth it (*e.g.* one can ensure a target performance of the decoder), the sign-bit of the adder could be protected by using classical fault-tolerant techniques.

III. Density Evolution

First, we note that our definition of *symmetry* is slightly more general than the one used in [6]. Indeed, even if the error injection models satisfy the symmetry condition from Equation (4), the noisy MS decoder does not necessarily very the ***symmetry*** property[2] from [6]. Nevertheless, the concentration and convergence properties proved in [6] for ***symmetric*** noisy message-passing decoders, can easily be generalized to our definition of *symmetry*.

The density evolution technique allows to recursively compute the probability mass functions of the exchanged extrinsic messages ($\alpha_{m,n}$ and $\beta_{m,n}$) and of the a posteriori information ($\tilde{\gamma}_n$), through the iterative decoding process. This is done under the independence assumption of exchanged messages, holding in the asymptotic limit of the code length, in which case the decoding performance converges to the cycle-free case. Due to the *symmetry* of the decoder, the analysis can be further simplified by assuming that the all-zero codeword is transmitted through the channel.

Due to space limitations, density evolution equations for the noisy MS decoder are not included here, but we only provide the notation and definitions required to understand the asymptotic analysis of the noisy MS decoder conducted in Section IV. We note however that the density evolution equations we provided in [10] can be ready generalized to the error models used in this paper.

A. Error Probability, Useful and Target Error-Rate Regions

1) Decoding Error Probability: The error probability at decoding iteration $\ell \geq 0$, is defined as:

$$P_e^{(\ell)} = \sum_{\tilde{z}=-\widetilde{Q}}^{-1} \widetilde{C}^{(\ell)}(\tilde{z}) + \frac{\widetilde{C}^{(\ell)}(0)}{2}, \tag{9}$$

where $\widetilde{C}^{(\ell)}(\tilde{z}) := \Pr(\tilde{\gamma}^{(\ell)} = \tilde{z})$ is the probability mass function of a posteriori information[3] at decoding iteration ℓ. Hence, in the asymptotic limit of the code-length, $P_e^{(\ell)}$ gives the probability of the hard bit estimates being in error at decoding iteration ℓ.

The following lower bounds can be derived from the probability of error injection within the *last* of the nested adders used to compute the a posteriori information value.

Proposition 1: The error probability at decoding iteration ℓ is lower-bounded as follows:

(a) For the sign-preserving noisy adder: $P_e^{(\ell)} \geq \dfrac{1}{2\widetilde{Q}} p_a$.

(b) For the full-depth noisy adder: $P_e^{(\ell)} \geq \dfrac{1}{2} p_a + \dfrac{1}{4\widetilde{Q}} p_a$.

Noiseless decoders exhibit a *threshold phenomenon*, separating the region where the decoding error probability goes to zero (as the number of decoding iterations ℓ goes to infinity), from that where it is bounded above zero [12]. Things get more complicated in case of noisy decoders. First, the decoding error probability has a more unpredictable behavior. It does not always converge and it may become periodic when the number of iterations goes to infinity (this will be discussed in Section IV). Second, the decoding error probability is always bounded above zero if $p_a > 0$ (Proposition 1), since there is a non-zero probability of fault injection at any decoding iteration. Hence, a decoding threshold, similar to the noiseless case, cannot longer be defined.

Following [6], we define below the notions of useful decoder and target error rate threshold. We consider a channel model depending on a channel parameter χ, such that the channel is degraded by increasing χ (for example, the crossover probability for the BSC, or the noise variance for the BI-AWGN channel). We will use subscript χ to indicate a quantity that depends on χ. Hence, in order to account for the fact that $P_e^{(\ell)}$ depends also on the value of the channel parameter, it will be denoted in the following by $P_{e,\chi}^{(\ell)}$.

2) Useful Region: The first step is to evaluate the channel and hardware parameters yielding a final probability of error (in the asymptotic limit of the number of iterations) less than the *input error probability*. The latter probability is given by $P_{e,\chi}^{(0)} = \sum_{z=-Q}^{-1} C(z) + \frac{1}{2} C(0)$, where C is the probability mass function of the quantized a priori information of the decoder ($\gamma = \mathbf{q}(y)$, see Algorithm 1).

[2]However, this property is verified in case of *highly symmetric* fault injection.

[3]We drop the variable-node index from the notation, in order to indicate the value of the a posteriori information $\tilde{\gamma}_n$ for a random variable-node n

Following [6], the decoder is said to be *useful* if $\left(P_{e,\chi}^{(\ell)}\right)_{\ell>0}$ is convergent, and:

$$P_{e,\chi}^{(\infty)} \stackrel{\text{def}}{=} \lim_{\ell\to\infty} P_{e,\chi}^{(\ell)} < P_{e,\chi}^{(0)} \tag{10}$$

The ensemble of the parameters that satisfy this condition constitutes the *useful region* of the decoder.

3) Target Error Rate Threshold: For noiseless-decoders, the decoding threshold is defined as the supremum channel noise, such that the error probability converges to zero as the number of decoding iterations goes to infinity. However, for noisy decoders this error probability does not converge to zero, and an alternative definition of the decoding threshold has been introduced in [6]. Accordingly, for a target bit-error rate η, the η-threshold is defined by:

$$\chi^*(\eta) = \sup\left\{\chi \mid P_{e,\chi}^{(\infty)} \text{ exists and } P_{e,\chi}^{(\infty)} < \eta\right\} \tag{11}$$

IV. Asymptotic Analysis of the Noisy Min-Sum Decoder

We consider the ensemble of regular LDPC codes with variable-node degree $d_v = 3$ and check-node degree $d_c = 6$. The following parameters will be used throughout this section with regard to the finite-precision MS decoder:

- The a priori information and extrinsic messages are quantized on $q = 4$ bits; hence, $Q = 7$ and $\mathcal{M} = \{-7, \ldots, +7\}$.
- The a posteriori information is quantized on $\tilde{q} = 5$ bits; hence, $\widetilde{Q} = 15$ and $\widetilde{\mathcal{M}} = \{-15, \ldots, +15\}$.

We restrict our analysis to the BSC channel with crossover probability p_0, and further assume that the channel input and output alphabet is $\mathcal{Y} = \{-1, +1\}$. For each $\mu \in \{1, \ldots, Q\}$ we define the quantization map $\mathbf{q}_\mu : \mathcal{Y} \to \mathcal{M}$ by:

$$\mathbf{q}_\mu(-1) = -\mu \text{ and } \mathbf{q}_\mu(+1) = +\mu \tag{12}$$

Thus, the a priori information of the decoder $\gamma_n \in \{\pm\mu\}$. The parameter μ will be referred to as the *channel-output scale factor*, or simply the *channel scale factor*.

The infinite-precision MS decoder is known to be independent of the scale factor μ. This is because μ factors out from all the processing steps of the decoding algorithm, and therefore does not affect in any way the decoding process. This is no longer true for the finite precision decoder (due to saturation effects), and we will show shortly that even in the noiseless case, the scale factor μ may significantly impact the performance of the finite precision MS decoder.

We start by analyzing the performance of the MS decoder with channel scale factor $\mu = 1$, and then we will analyze its performance with an optimized value of μ.

A. Min-Sum Decoder with Channel Scale Factor $\mu = 1$

The case $\mu = 1$ leads to an "unconventional" behavior, as in some particular cases the noise introduced by the device can help the MS decoder to escape from fixed points attractors, and may actually result in an increased correction capacity with respect to the noiseless decoder. This behavior will be discussed in more details in this section.

We start with the noiseless decoder case. In this case, the decoder exhibits a *classical* threshold phenomenon: there exists a threshold value p_{th}, such that $P_e^{(\infty)} = 0$ for any $p_0 < p_{\text{th}}$. This threshold value, which can be computed by density evolution, is $p_{\text{th}} = 0.039$. Now, we consider a p_0 value slightly greater than the threshold of the noiseless decoder, and investigate the effect of the noisy adder on the decoder performance. Let us fix $p_0 = 0.06$. Figure 1 shows the decoding error probability at iteration ℓ, for different error probability parameters $p_a \in \{10^{-30}, 10^{-15}, 10^{-5}\}$ of the noisy adder. For each p_a value, there are two superimposed curves, corresponding to the full-depth ("fd", solid curve) and sing-preserving ("sp", dashed curve) error models of the noisy adder. The error probability of the noiseless decoder is also plotted (solid black curve): it can be seen that it increases rapidly from the initial value $P_e^{(0)} = p_0$ and closely approaches the limit value $P_e^{(\infty)} = 0.323$ after a few number of iterations. When the adder is noisy, the error probability increases during the first decoding iterations, behaving similarly to the noiseless case. It may approach the limit value from the noiseless case, but starts decreasing after some number of decoding iterations. However, note that it remains bounded above zero (although not apparent in the figure), according to the lower bounds from Proposition 1, and it can actually be numerically verified that these bounds are nearly tight.

The above behavior of the MS decoder can be explained by examining the evolution of the probability mass function of the a posteriori information, denoted by $\widetilde{C}^{(\ell)}$ (see Section III), for $\ell \geq 0$. In the noiseless case, $\widetilde{C}^{(\ell)}$ reaches a fixed point of the density evolution for $\ell \approx 20$. Note that since all variable-nodes are of degree $d_v = 3$, it can be easily seen that for $\ell \geq 1$, $\widetilde{C}^{(\ell)}$ is supported only on even values of $\widetilde{\mathcal{M}}$. These "gaps" in the support of the probability mass function seem to lead to favorable conditions for the occurrence of density-evolution fixed points. In the noisy case, $\widetilde{C}^{(\ell)}$ evolves virtually the same as in the noiseless case during the first iterations. However, the noise present in the adder progressively fills the "gaps" in the support of $\widetilde{C}^{(\ell)}$, which allows the decoder to escape from the fixed point attractor.

It is worth noting that neither the noisy comparator nor the XOR-operator can help the decoder to escape from these fixed-point attractors, as they do not allow "filling the gaps" in the support of $\widetilde{C}^{(\ell)}$.

We focus now on the useful region of the noisy MS decoder. We assume that only the adder is noisy, while the comparator and the XOR-operator are noiseless. The useful useful region for the sign-protected noisy adder model is shown in Figure 2. The useful region is shaded in gray and delimited by either a solid black curve or a dashed red curve. Although one would expect that $P_e^{(\infty)} = p_0$ on the border of the useful region, this equality only holds on the solid black border. On the dashed red border, one has $P_e^{(\infty)} < p_0$. The reason why the useful region does not extend beyond the dashed red border is that for points located on the other side of this border the sequence $(P_e^{(\ell)})_{\ell>0}$ is periodic, and hence it does

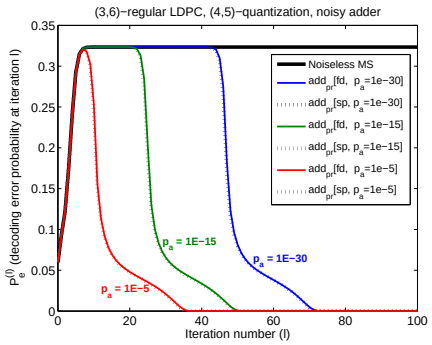


Figure 1. Effect of the noisy adder on the asymptotic performance of the MS decoder ($p_0 = 0.06$)

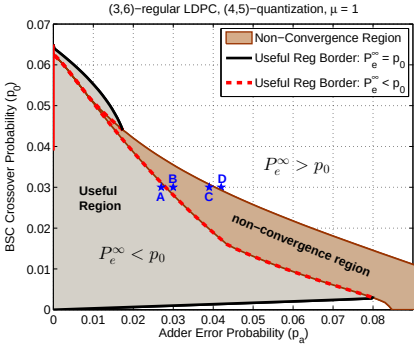


Figure 2. Useful and non-convergence regions of the MS decoder with sign-preserving noisy adder

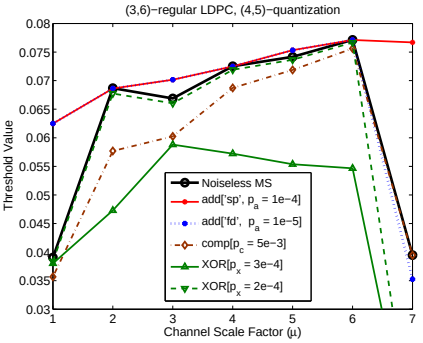
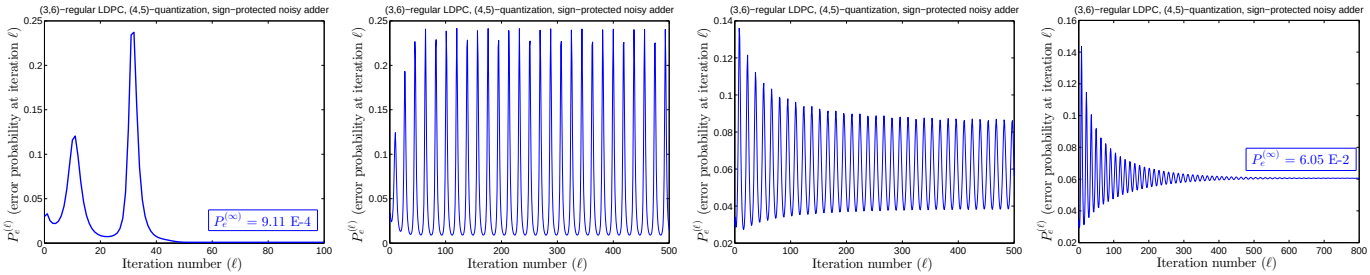


Figure 3. Threshold values of noiseless and noisy MS decoders with various channel scale factors



(a) $A(p_0 = 0.03, p_a = 0.027)$ (b) $B(p_0 = 0.03, p_a = 0.03)$ (c) $C(p_0 = 0.03, p_a = 0.039)$ (d) $D(p_0 = 0.03, p_a = 0.042)$

Figure 4. Decoding error probability $P_e^{(\ell)}$ of the noisy MS decoder, for $p_0 = 0.03$ and sign-preserving noisy adder with various p_a values

not converge. The region shaded in brown in Figure 2 is the *non-convergence region* of the decoder. Note that the non-convergence region gradually narrows in the upper part, and there is a small portion of the useful region delimited by the non-convergence region on the left and the black border on the right. Finally, we note that points with $p_a = 0$ (noiseless decoder) and $p_0 > 0.039$ (threshold of the noiseless decoder) – represented by the solid red line superimposed on the vertical axis in Figure 2 – are excluded from the useful region. Indeed, for such points $P_e^{(\infty)} > p_0$; however, for p_a greater than but close to zero, we have $P_e^{(\infty)} \approx \frac{p_a}{2\widetilde{Q}}$.

We exemplify the decoder behavior on four points located on one side and the other of the left and right boundaries of the non-convergence region. These points are indicated in Figure 2 by A, B, C, and D. For all the four points $p_0 = 0.03$, while $p_a = 0.027, 0.03, 0.039$, and 0.042, respectively. The error probability $(P_e^{(\ell)})_{\ell>0}$ is plotted for each one of these points in Figure 4. The point A belongs to the useful region, and it can be seen from Figure 4(a) that $(P_e^{(\ell)})_{\ell>0}$ converges to $P_e^{(\infty)} = 9.11 \times 10^{-4} < p_0$. For the point B, located just on the other side of the dashed red border of the useful region, $(P_e^{(\ell)})_{\ell>0}$ exhibits a periodic behavior (although we only plotted the first 500 iterations, we verified the periodic behavior on the first 5×10^4 iterations). Crossing the non-convergence region from left to the right, the amplitude between the inferior and superior limits of $(P_e^{(\ell)})_{\ell>0}$ decreases (point C), until it reaches again a convergent behavior (point D). Note that D is outside the useful region, as $(P_e^{(\ell)})_{\ell>0}$ converges to $P_e^{(\infty)} = 0.0605 > p_0$.

The non-convergence region gradually narrows in the upper part, and for $0 \leq p_a < 0.01$ it takes the form of a *discontinuity line*: $P_e^{(\infty)}$ takes values close to 10^{-4} just below this line, and values greater than 0.05 above this line.

Note that points (p_a, p_0) with $p_0 < \frac{p_a}{2\widetilde{Q}} = \frac{p_a}{30}$ cannot belong to the useful region, since from Proposition 1 we have $P_e^{(\infty)} \geq \frac{p_a}{2\widetilde{Q}} > p_0$. Moreover, we note that the bottom border of the useful region (solid black curve) is virtually identical to, but slightly above, the line defined by $p_0 = \frac{p_a}{2\widetilde{Q}}$.

B. Optimization of the Channel Scale Factor

In this section we show that the decoder performance can be significantly improved by using an appropriate choice of the channel scale factor μ. Figure 3 shows the threshold values for the noiseless and several noisy decoders with channel scale factors $\mu \in \{1, 2, \ldots, 7\}$. For the noisy decoders, the threshold values are computed for a target error probability $\eta = 10^{-5}$ (see Equation (11)).

The solid black curve in Figure 3 correspond to the noiseless decoder. The solid red curve and the dotted blue curve correspond to the MS decoder with sign-preserving noisy adder and full-depth noisy adder, respectively. The adder error probability[4] is $p_a = 10^{-4}$ for the sign-preserving noisy adder, and $p_a = 10^{-5}$ for the full-depth adder. The two curves are

[4]According to Proposition 1, a necessary condition to achieve a target error probability $P_e^{(\infty)} \leq \eta = 10^{-5}$ is $p_a \leq 2\widetilde{Q}\eta = 3 \times 10^{-4}$ for the signed-protected adder, and $p_a \leq 2\eta\frac{2\widetilde{Q}+1}{2\widetilde{Q}} = 2.07 \times 10^{-5}$ for the full-depth adder.

superimposed for $1 \leq \mu \leq 6$, and differ only for $\mu = 7$. The corresponding threshold values are equal to those obtained in the noiseless case for $\mu \in \{2, 4, 6\}$. For $\mu \in \{1, 3, 5\}$, the MS decoders with noisy-adders exhibit better thresholds than the noiseless decoder. This is due to the fact that the messages alphabet $\mathcal{M}$ is underused by the noiseless decoder, since all the exchanged messages are necessarily odd (recall that all variable-nodes are of degree $d_v = 3$). For the MS decoders with noisy adders, the noise present in the adders leads to a more efficient use of the messages alphabet, which allows the decoder to escape from fixed-point attractors and hence results in better thresholds (Section IV-A).

Figure 3 also shows a curve corresponding to the MS decoder with a noisy comparator having $p_c = 0.005$, and two curves for the MS decoder with noisy XOR-operators, having respectively $p_x = 2 \times 10^{-4}$ and $p_x = 3 \times 10^{-4}$.

Concerning the noisy XOR-operator, it can be seen that the threshold values corresponding to $p_x = 2 \times 10^{-4}$ are very close to those obtained in the noiseless case, except for $\mu = 7$ (the same holds for values $p_x < 2 \times 10^{-4}$). However, a significant degradation of the threshold can be observed when slightly increasing the XOR error probability to $p_x = 3 \times 10^{-4}$. Moreover, although not shown in the figure, it is worth mentioning that for $p_x \geq 5 \times 10^{-4}$, the target error probability $\eta = 10^{-5}$ can no longer be reached (thus, all threshold values are equal to zero).

Finally, we note that except for the noisy XOR-operator with $p_x = 3 \times 10^{-4}$, the best choice of the channel scale factor is $\mu = 6$. For the noisy XOR-operator with $p_x = 3 \times 10^{-4}$, the best choice of the channel scale factor is $\mu = 3$. This is rather surprising, as in this case the messages alphabet is underused by the decoder: all the exchanged messages are odd, and the fact that the XOR-operator is noisy does not change their parity.
Assumption: In the following sections, we will investigate the impact of the noisy adder, comparator and XOR-operator on the MS decoder performance, *assuming that the channel scale factor is* $\mu = 6$.

C. Study of the Impact of the Noisy Adder ($\mu = 6$)

In order to evaluate the impact of the noisy adder on the MS decoder performance, the useful region and the η-threshold regions have been computed, assuming that only the adders within the VN-processing step are noisy ($p_a > 0$), while the CN-processing step is noiseless ($p_x = p_c = 0$). This regions are represented in Figure 5 and Figure 6, for the sign-preserving and the full-depth noisy adder models, respectively.

The useful region is delimited by the solid black curve. The vertical lines delimit the η-threshold regions, for $\eta = 10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}$ (from right to the left).

Note that unlike the case $\mu = 1$ (Section IV-A), there is no non-convergence region when the channel scale factor is set to $\mu = 6$. Hence, the border of the useful region corresponds to points (p_a, p_0) for which $P_e^{(\infty)} = p_0$. However, it can be observed that there is still a *discontinuity line* (dashed red curve) inside the useful region. This discontinuity line does not hide a periodic (non-convergent) behavior, but it is due to the occurrence of an *early plateau phenomenon* in the convergence of $(P_e^{(\ell)})_\ell$. This phenomenon is illustrated in Figure 8, where the error probability $(P_e^{(\ell)})_\ell$ is plotted as a function of the iteration number ℓ, for the two points A and B from Figure 5. For the point A, it can be observed that the error probability $P_e^{(\ell)}$ reaches a first plateau for $\ell \approx 50$, and then drops to 3.33×10^{-6} for $\ell \geq 250$. For the point B, $P_e^{(\ell)}$ behaves in a similar manner during the first iterations, but it does not decrease below the plateau value as ℓ goes to infinity. Although we have no analytic proof of this fact, it was numerically verified for $\ell \leq 5 \times 10^5$.

In Figure 10, we plotted the asymptotic error probability $P_e^{(\infty)}$ as a function of p_0, for the noiseless decoder ($p_a = 0$), and for the sign-preserving noisy adder with error probability values $p_a = 10^{-4}$ and $p_a = 0.05$. In each plot we have also represented two points $p_0^{(\mathrm{U})}$ and $p_0^{(\mathrm{DL})}$, corresponding respectively to the values of p_0 on the upper-border of the useful region, and on the discontinuity line. Hence, $p_0^{(\mathrm{DL})}$ coincides with the classical threshold of the MS decoder in the noiseless case, and it can be seen as an appropriate generalization of the classical threshold to the case of noisy decoders. In the following, $p_0^{(\mathrm{DL})}$ will be referred to as the ***functional threshold*** of the noisy decoder, and the sub-region of the useful region located below the discontinuity line will be referred to as the ***functional region***. Within this region, if the adder error probability is small enough, it can be observed that the lower-bounds provided in Proposition 1 are tight.

D. Study of the Impact of the Noisy XOR-operator ($\mu = 6$)

The useful region and the η-threshold regions of the decoder, assuming that only the XOR-operator used within the CN-processing step is noisy, are plotted in Figure 7. Similar to the noisy-adder case, a discontinuity line can be observed inside the useful region, which delimits the *functional region* of the decoder.

Comparing the η-threshold regions from Figures 5, 6 and 7, it can be observed that in order to achieve a target error probability $P_e^{(\infty)} \leq 10^{-6}$, the error probability parameters of the noisy adder and of the noisy XOR-operator must satisfy:

- $p_a < 1.17 \times 10^{-6}$, for the full-depth noisy-adder;
- $p_a < 3 \times 10^{-5}$, for the sign-protected noisy-adder;
- $p_x < 7 \times 10^{-5}$, for the noisy XOR-operator.
  (moreover, values of p_x up to 1.4×10^{-4} are tolerable if p_0 is sufficiently small)

The most stringent requirement concerns the error probability of the full-depth noisy-adder, thus we may consider that it has the most negative impact on the decoder performance. On the other hand, the less stringent requirement concerns the error probability of the noisy XOR-operator.

Finally, it is worth noting that in practical cases the value of p_x should be significantly lower than the value of p_a (given the high number of elementary gates contained in the adder). Moreover, since the XOR-operators used to compute the signs of CN messages represent only a small part of the decoder, this part of the circuit could be made reliable by using classical

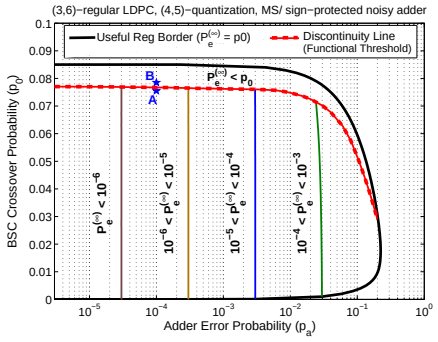


Figure 5. Useful and η-threshold regions of the MS decoder with sing-preserving noisy adder

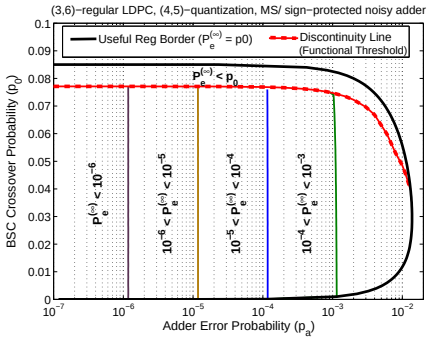


Figure 6. Useful and η-threshold regions of the MS decoder with full-depth noisy adder

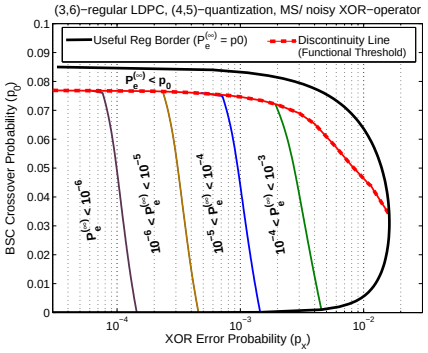
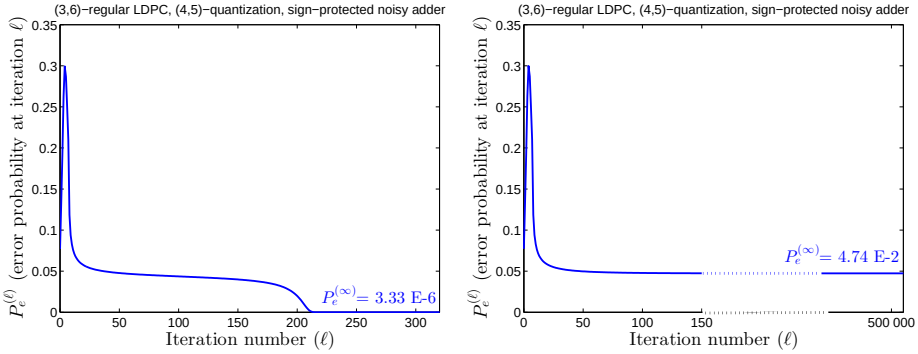


Figure 7. Useful and η-threshold regions of the MS decoder with noisy XOR-operator



(a) Point $A(p_0 = 0.0770, p_a = 10^{-4})$ (b) Point $B(p_0 = 0.0772, p_a = 10^{-4})$

Figure 8. Illustration of the early plateau phenomenon (points A and B from Figure 5)

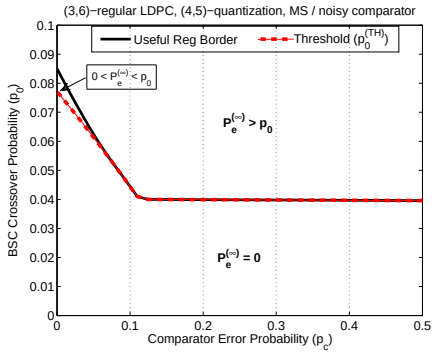
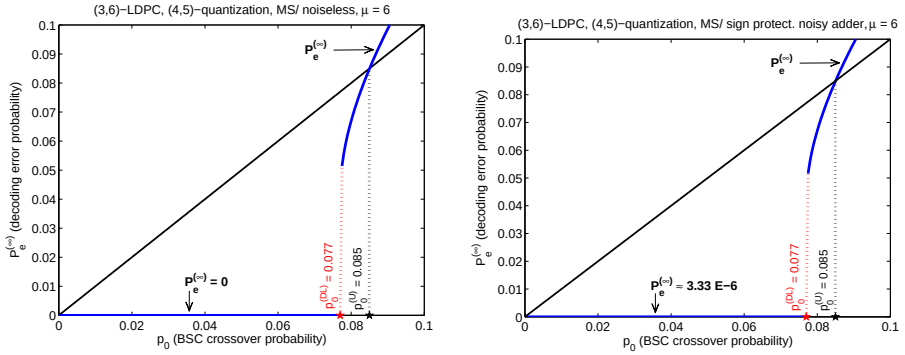


Figure 9. Useful region and threshold curve of the MS decoder with noisy comparator



(a) $p_a = 0$ (noiseless decoder) (b) $p_a = 10^{-4}$ (c) $p_a = 0.05$

Figure 10. Asymptotic error probability $P_e^{(\infty)}$ as a function of p_0; noiseless and noisy MS decoder with sign-protected noisy adder

fault-tolerant methods, with a limited impact on the overall decoder design.

E. Study of the Impact of the Noisy Comparator ($\mu = 6$)

This section investigates the case when comparators used within the CN-processing step are noisy ($p_c > 0$), but $p_a = p_x = 0$. Contrary to the previous cases, this case exhibits a "classical" threshold phenomenon, similar to the noiseless case: for a given $p_c > 0$, the exists a p_0-threshold value, denoted by $p_0^{(\mathrm{TH})}$, such that $P_e^{(\infty)} = 0$ for any $p_0 < p_0^{(\mathrm{TH})}$.

The threshold value $p_0^{(\mathrm{TH})}$ is plotted as a function of p_c in Figure 9. The functional region of the decoder is located below the threshold curve, and $P_e^{(\infty)} = 0$ for any point within this region. In particular, it can be seen that $P_e^{(\infty)} = 0$ for any $p_0 \lessapprox 0.039$ and any $p_c > 0$. Although such a threshold phenomenon might seem surprising for a noisy decoder, it can be easily explained. The idea behind is that in this case the crossover probability of the channel is small enough, so that in the CN-processing step only the sign of check-to-variable

messages is important, but not their amplitudes. In other words a decoder that only computes (reliably) the signs of check-node messages and randomly chooses their amplitudes, would be able to perfectly decode the received word.

Finally, we note that the useful region of the decoder extends slightly above the threshold curve: for p_c close to 0, there exists a small region above the threshold curve, within which $0 < P_e^{(\infty)} < p_0$.

V. Finite Length Performance of the Noisy Min-Sum Decoder

The goal of this section is to corroborate the asymptotic analysis from the previous section, through finite-length simulations. Unless otherwise stated, the $(3, 6)$-regular LDPC code with length $N = 1008$ bits from [13] will be used throughout this section.

A. Early stopping criterion

As described in Algorithm 1, each decoding iteration also comprises a *hard decision* step, in which each transmitted bit is estimated according to the sign of the a posteriori information, and a *syndrome check* step, in which the syndrome of the estimated word is computed. Both steps are assumed to be *noiseless*, and the syndrome check step acts as an *early stopping criterion*: the decoder stops when whether the syndrome is $+1$ (the estimated word is a codeword) or a maximum number of iterations is reached. We note however that the syndrome check step is optional and, if missing, the decoder stops when the maximum number of iterations is reached.

The reason why we stress the difference between the MS decoder with and without the syndrome check step is because, as we will see shortly, the *noiseless* early stopping criterion may significantly improve the bit error rate performance of the *noisy* decoder in the error floor region.

Unless otherwise stated, the MS decoder is assumed to implement the *noiseless* stopping criterion (syndrome check step). The maximum number of decoding iterations is fixed to 100 throughout this section.

B. Finite-length performance for various channel scale factors

Figure 11 shows the bit error rate (BER) performance of the finite-precision MS decoder (both noiseless and noisy) with various channel scale factors. For comparison purposes, we also included the BER performance of the Belief-Propagation decoder (solid black curve, no markers) and of the infinite-precision MS decoder (dashed blue curve, no markers).

It can be observed that the worst performance is achieved by the infinite-precision MS decoder (!) and the finite-precision noiseless MS decoder with channel scale factor $\mu = 1$ (both curves are virtually indistinguishable). The BER performance of the latter improves significantly when using a sign-preserving noisy adder with error probability $p_a = 0.001$ (dashed red curve with empty circles).

For a channel scale factor $\mu = 6$, both noiseless and noisy decoders have almost the same performance (solid and dashed green curves, with triangular markers). Remarkably, the achieved BER is very close to the one achieved by the Belief-Propagation decoder!

These results corroborate the asymptotic analysis from Section IV-B concerning the channel scale factor optimization.

C. Error floor performance

Surprisingly, the BER curves of the noisy decoders from Figure 11 do not show any error floor down to 10^{-7}. However, according to Proposition 1, the decoding error probability should be lower-bounded by $P_e^{(\ell)} \geq \frac{1}{2\widetilde{Q}} p_a = 3.33 \times 10^{-5}$ (see also the η-threshold regions in Figure 5).

The fact that the observed decoding error probability may decrease below the above lower-bound is due to the early stopping criterion (syndrome check step) implemented within the MS decoder. Indeed, as we observed in the previous section, the above lower-bound is tight, when ℓ (the iteration number) is sufficiently large. Therefore, as the iteration number increases, the expected number of erroneous bits gets closer and closer to $\frac{1}{2\widetilde{Q}} p_a N = 0.034$, and the probability of not having any erroneous bit within one iteration approaches $\left(1 - \frac{1}{2\widetilde{Q}} p_a\right)^N = 0.967$. As the decoder performs more and more iterations, it will eventually reach an error free iteration. The absence of errors is at once detected by the noiseless syndrome check step, and the decoder stops.

To illustrate this behavior, we plotted the Figure 12 the BER performance of the noisy MS decoder, with and without early stopping criterion. The noisy MS decoder comprises a sign-preserving noisy adder with $p_a = 0.001$, while the comparator and the XOR-operator are assumed to be noiseless ($p_c = p_x = 0$). Two codes are simulated, the first with length $N = 1008$ bits, and the second with length $N = 10000$ bits. In case that the noiseless early stopping criterion is implemented (solid curves), it can be seen that none of the BER curves show any error floor down to 10^{-8}. However, if the early stopping criterion is not implemented (dashed curves), corresponding BER curves exhibit an error floor at $\approx 3.33 \times 10^{-5}$, as predicted by Proposition 1.

D. Finite-length performance for various parameters of the probabilistic error models

In this section we investigate the finite-length performance when all the MS components (adder, comparator, and XOR-operator) are noisy. In order to reduce the number of simulations, we assume that $p_a = p_c \geq p_x$. Concerning the noisy adder, we evaluate the BER performance for both the sign-preserving and the full-depth error models. Simulation results are presented in Figures 13–16.

In case the noisy-adder is sign-preserving, it can be seen that the MS decoder can provide reliable error protection for all the noise parameters that have been simulated. Of course, depending on the error probability parameters of the noisy components, there is a more or less important degradation of the achieved BER with respect to the noiseless case. But in all cases the noisy decoder can achieve a BER less than 10^{-7}. This is no longer true for the full-depth noisy adder: it can

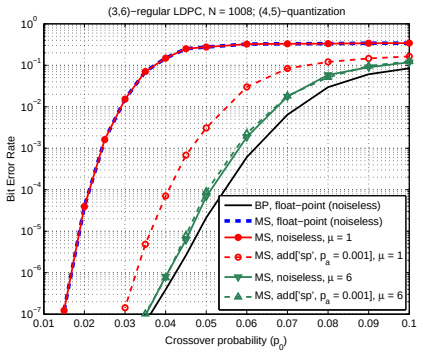


Figure 11. BER performance of noiseless and noisy MS decoders with various channel scale factors

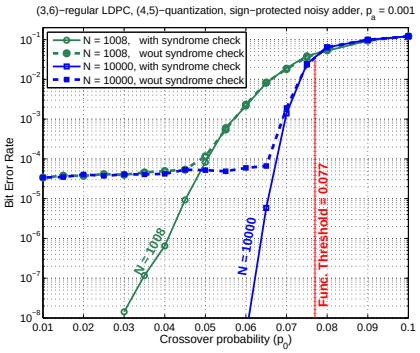


Figure 12. BER performance with and without early stopping criterion (MS decoder with sign-preserving noisy adder, $p_a = 0.001$)

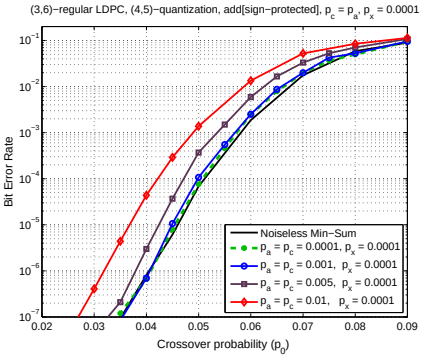


Figure 13. BER performance, noisy MS, sign-preserving noisy adder, $p_c = p_a$, $p_x = 0.0001$

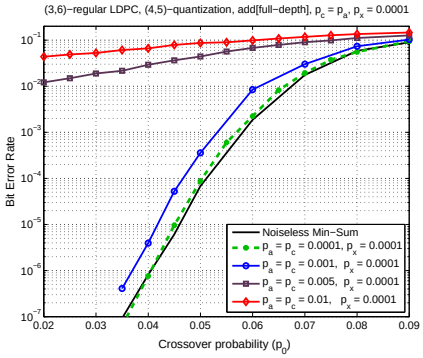


Figure 14. BER performance, noisy MS, full-depth noisy adder, $p_c = p_a$, $p_x = 0.0001$

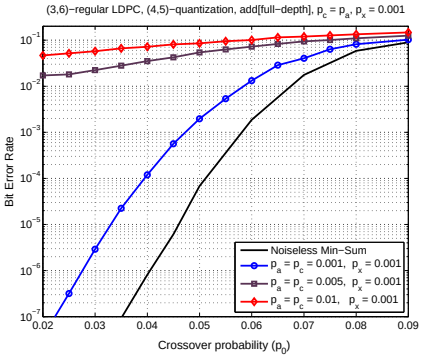


Figure 15. BER performance, noisy MS, full-depth noisy adder, $p_c = p_a$, $p_x = 0.001$

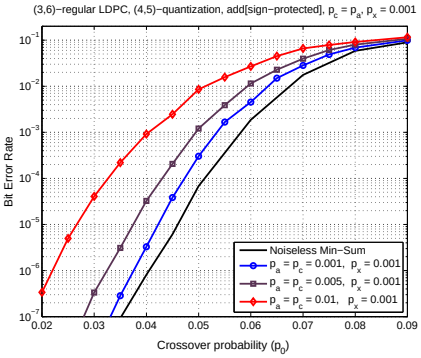


Figure 16. BER performance, noisy MS, sign-preserving noisy adder, $p_c = p_a$, $p_x = 0.001$

be seen that for $p_c = p_a \geq 0.005$, the noisy decoder cannot achieve bit error rates below 10^{-2}.

VI. Conclusion

This paper investigated the asymptotic and finite length behavior of the noisy MS over the BSC channel. We demonstrated the impact of the channel scale factor on the decoder performance, both for the noiseless and for the noisy decoder. We also highlighted the fact that an inappropriate choice the channel scale factor may lead to an *unconventional* behavior, in the sense that the noise introduce by the device may actually result in an increased correction capacity with respect to the noiseless decoder. We analyzed the asymptotic performance of the noisy MS decoder in terms of useful regions and target-BER thresholds, and further revealed the existence of a different threshold phenomenon, which was referred to as functional threshold. Finally, we also corroborated the asymptotic analysis through finite-length simulations.

References

[1] S. K. Chilappagari, M. Ivkovic, and B. Vasic, "Analysis of one step majority logic decoders constructed from faulty gates," in *Proc. of IEEE Int. Symp. on Information Theory*, 2006, pp. 469–473.

[2] B. Vasic and S. K. Chilappagari, "An information theoretical framework for analysis and design of nanoscale fault-tolerant memories based on low-density parity-check codes," *IEEE Trans. on Circuits and Systems I: Regular Papers*, vol. 54, no. 11, pp. 2438–2446, 2007.

[3] C. Winstead and S. Howard, "A probabilistic LDPC-coded fault compensation technique for reliable nanoscale computing," *IEEE Trans. on Circuits and Systems II: Express Briefs*, vol. 56, no. 6, pp. 484–488, 2009.

[4] Y. Tang, C. Winstead, E. Boutillon, C. Jego, and M. Jezequel, "An LDPC decoding method for fault-tolerant digital logic," in *IEEE Int. Symp. on Circuits and Systems (ISCAS)*, 2012, pp. 3025–3028.

[5] A. M. Hussien, M. S. Khairy, A. Khajeh, A. M. Eltawil, and F. J. Kurdahi, "A class of low power error compensation iterative decoders," in *IEEE Global Telecom. Conf. (GLOBECOM)*, 2011, pp. 1–6.

[6] L. R. Varshney, "Performance of LDPC codes under faulty iterative decoding," *IEEE Trans. Inf. Theory*, vol. 57, no. 7, pp. 4427–4444, 2011.

[7] S. Yazdi, H. Cho, Y. Sun, S. Mitra, and L. Dolecek, "Probabilistic analysis of Gallager B faulty decoder," in *IEEE Int. Conf. on Communications (ICC)*, 2012, pp. 7019–7023.

[8] S. Yazdi, C. Huang, and L. Dolecek, "Optimal design of a Gallager B noisy decoder for irregular LDPC codes," *IEEE Comm. Letters*, vol. 16, no. 12, pp. 2052–2055, 2012.

[9] S. Yazdi, H. Cho, and L. Dolecek, "Gallager B decoder on noisy hardware," *IEEE Trans. on Comm.*, vol. 66, no. 5, pp. 1660–1673, 2013.

[10] C. L. Kameni Ngassa, V. Savin, and D. Declercq, "Min-Sum-based decoders running on noisy hardware," in *Proc. of IEEE Global Communications Conference (GLOBECOM)*, 2013.

[11] ——, "Analysis of Min-Sum based decoders implemented on noisy hardware," in *Proc. of Asilomar Conference on Signals, Systems and Computers*, 2013.

[12] T. J. Richardson and R. L. Urbanke, "The capacity of low-density parity-check codes under message-passing decoding," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 599–618, 2001.

[13] D. J. MacKay. Encyclopedia of sparse graph codes. [Online]. Available: http://www.inference.phy.cam.ac.uk/mackay/codes/data.html