Density Evolution and Functional Threshold for the Noisy Min-Sum Decoder

Christiane L. Kameni Ngassa*,#, Valentin Savin*, Elsa Dupraz#, David Declercq#

*CEA-LETI, Minatec Campus, Grenoble, France,
{christiane.kameningassa, valentin.savin}@cea.fr
#ETIS, ENSEA / CNRS UMR-8051 / Univ. Cergy-Pontoise, France,
{elsa.dupraz, declercq}@ensea.fr

***Abstract*—This paper investigates the behavior of the Min-Sum decoder running on noisy devices. The aim is to evaluate the robustness of the decoder in the presence of computation noise, *e.g.* due to faulty logic in the processing units, which represents a new source of errors that may occur during the decoding process. To this end, we first introduce probabilistic models for the arithmetic and logic units of the the finite-precision Min-Sum decoder, and then carry out the density evolution analysis of the noisy Min-Sum decoder. We show that in some particular cases, the noise introduced 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 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], 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 [4]–[8], the authors investigated the asymptotic behavior of the noisy Gallager-B decoder defined over binary and non-binary alphabets. Finite Alphabet Iterative Decoders (FAIDs) under processing errors have also been investigated in [9], [10]. Recently, the Min-Sum decoding under unreliable message storage has been investigated in [11], [12]. 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.

In this paper, we investigate the asymptotic and finite-length behavior of the noisy MS decoder. We refine the probabilistic error models for the noisy MS decoder we used previously in [13], [14], and discuss their symmetry properties. We derive density evolution equations and conduct a thorough asymptotic analysis of the noisy MS decoder. We also highlight a wide variety of more or less conventional behaviors and 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 remainder of the paper is organized as follows. Section II discusses the probabilistic error models for the noisy arithmetic operators and Section III introduces the noisy MS decoder. Section IV 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 V. Finally, Section VI corroborates the asymptotic analysis through finite-length simulations, and Section VII concludes the paper.

II. Probabilistic Error Models for the Noisy Arithmetic Operators

A. Noisy Message-Passing Decoders

The model for noisy Message-Passing (MP) decoders proposed in [3] 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. The message-passing noise is simply modeled as a noisy

The research leading to these results has received funding from the European Union's Seventh Framework Program FP7/2007-2013, under Grant Agreement number 309129 (*i*-RISC project).

Part of this work has been presented at *Asilomar Conference on Signals, Systems and Computers* 2013, *IEEE Global Communications Conference* (GLOBECOM) 2013, and *IEEE Information Theory and Applications Workshop* (ITA) 2014.

channel. Hence, transmitting a message over a noisy wire is emulated by passing that message through the corresponding noisy channel.

However, in [3] 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 [3] 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 [3]). 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.

Finally, we note that the density evolution analysis of noisy decoders has some intrinsic limitations, as it only applies to symmetric decoding functions without memory. Since noise is part of the decoder, it follows that the transient error model has also to be symmetric. While we do not expect real error models to be symmetric, the theoretical analysis conducted in this paper is aimed at understanding the limits of iterative MS decoding under faulty hardware, thus providing boundary conditions on the allowed level of hardware-induced errors.

B. Error Models for Noisy Adders

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 $10 \cdots 0$.

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:

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

where $\oplus$ denotes the bitwise XOR operation. In can be seen that this error model is tantamount to passing v through a $|\mathcal{V}|$-ary symmetric channel, where $|\mathcal{V}|$ is the number of elements of $\mathcal{V}$.

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:

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

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*:

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

The above condition ensures that the noisy decoder is symmetric, which is an essential prerequisite to density evolution analysis (see Section IV).

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 the case where the two's complement representation is used, they are both symmetric, but not highly symmetric.

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

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

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.

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 $\lambda \leq \theta$ [13]. Hence, $\lambda = \theta$ corresponds to the above full-depth model, while $\lambda = \theta - 1$ corresponds to the sign-preserving model. However, for the two's complement representation, such a model is **not** symmetric if $\lambda < \theta - 1$!

Note that it might 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 it 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.

C. Error Models for Noisy Comparators and Noisy XOR Operators

The noisy minimum operator with error probability p_c, denoted by $\mathbf{m}_{\text{pr}}$, is defined as follows:

$$\mathbf{m}_{\text{pr}}(x,y) = \begin{cases} \min(x,y), & \text{with probability } 1-p_c \\ \max(x,y), & \text{with probability } p_c \end{cases} \tag{6}$$

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:

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

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

III. Noisy Min-Sum Decoder

A. Notation

We consider an LDPC code defined by a Tanner graph $\mathcal{H}$, with N variable nodes denoted by $n \in \{1,2,...,N\}$, and M check-nodes denoted by $m \in \{1,2,...,M\}$. The set of nodes connected to a variable node n (resp. a check node m) will be denoted by $\mathcal{H}(n)$ (resp. $\mathcal{H}(m)$). For a message passing decoder, we denote by γ_n and $\tilde{\gamma}_n$ the *a priori* and the *a posteriori* information of the variable node n (i.e. decoder's input and output values), and by $\alpha_{m,n}$ and $\beta_{m,n}$ the *variable-to-check message* sent from n to m and the *check-to-variable* message sent from m to n, respectively.

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** $\alpha_{m,n} = \gamma_n$;
for Iter $= 1, \ldots,$ Max_Iter
  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)

Remark: Let ω be any of the $\mathbf{a}_{\text{pr}}, \mathbf{m}_{\text{pr}}$, or $\mathbf{x}_{\text{pr}}$ operators. We extend the definition of ω from 2 to more operands as follows. If $\{x_i\}_{i=1:n}$ is a set of n operands, we define $\omega\left(\{x_i\}_{i=1:n}\right) \stackrel{\text{def}}{=} \omega(x_1, \omega(x_2, \ldots \omega(x_{n-1}, x_n)\ldots))$.

B. Noisy Min-Sum Decoding

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's 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.

C. 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 (see footnote remark in Algorithm 1). 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.

IV. Density Evolution

A. Concentration and Convergence Properties

First, we note that our definition of *symmetry* is slightly more general than the one used in [3]. Indeed, even if the error injection models satisfy the symmetry condition from Equation (4), the noisy MS decoder does not necessarily verify the ***symmetry*** property from [3]. Nevertheless, the concentration and convergence properties proved in [3] for ***symmetric*** noisy message-passing decoders, can easily be generalized to our definition of *symmetry*.

We summarize below the most important results. We consider an *ensemble* of LDPC codes, with length N and fixed degree distribution polynomials [15]. We choose a random code $\mathbf{C}$ from this ensemble and assume that a random codeword $\underline{\boldsymbol{x}} \in \{-1, +1\}^N$ is sent over a binary-input memoryless symmetric channel. We fix some number of decoding iterations $\ell > 0$, and denote by $E_{\mathbf{C}}^{(\ell)}(\underline{\boldsymbol{x}})$ the expected fraction of incorrect *messages*[1] at iteration ℓ.

Theorem 1: For the noisy MS decoder from Algorithm 1, the following properties hold:

1) [*Conditional Independence of Error*] For any decoding iteration $\ell > 0$, the expected fraction of incorrect messages $E_{\mathbf{C}}^{(\ell)}(\underline{\boldsymbol{x}})$ does not depend on $\underline{\boldsymbol{x}}$. Therefore, we define $E_{\mathbf{C}}^{(\ell)} := E_{\mathbf{C}}^{(\ell)}(\underline{\boldsymbol{x}})$.
2) [*Cycle-Free Case*] If the graph of $\mathbf{C}$ contains no cycles of length 2ℓ or less, $E_{\mathbf{C}}^{(\ell)}$ does not depend on the code $\mathbf{C}$ or the code-length N, but only on the degree distribution polynomials; in this case, it will be further denoted by $E_{\infty}^{(\ell)}(\underline{\boldsymbol{x}})$.
3) [*Concentration Around the Cycle-Free Case*] For any $\delta > 0$, the probability that $E_{\mathbf{C}}^{(\ell)}$ lies outside the interval $\left(E_{\infty}^{(\ell)}(\underline{\boldsymbol{x}}) - \delta, E_{\infty}^{(\ell)}(\underline{\boldsymbol{x}}) + \delta\right)$ converges to zero exponentially fast in N.

Sketch of proof: Let $\imath : \mathcal{V} \times \mathcal{E} \to \mathcal{V}$ by any of the full-depth or sign-preserving error injection models used the define the noisy adder. For any random variable V defined on $\mathcal{V}$, let $\phi_V^{(\imath)}$ and $\phi_{-V}^{(\imath)}$ denote the probability mass functions of the random variables obtained by injecting errors on the output of V and $-V$, respectively. From Equation (4) it follows easily that:

$$\phi_V^{(\imath)}(w) = \phi_{-V}^{(\imath)}(-w), \ \forall w \in \mathcal{V} \tag{8}$$

Finally, using Equation (8), the proof of Theorem 1 can be derived using essentially the same arguments as in [3, Theorems 1-3]. □

B. Density Evolution Equations

The density evolution technique allows recursively computing 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.

The density evolution equations for the noisy MS decoder were included in Appendix A. Here, we only provide the notation and definitions required to understand the asymptotic analysis of the noisy MS decoder conducted in Section V.

C. 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 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 in Algorithm 1.

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

[1]Here, *"messages"* may have any one of the three following meanings: "variable-node messages", or "check-node messages", or "a posteriori information values".

zero (as the number of decoding iterations ℓ goes to infinity), from that where it is bounded above zero [15]. This definition does not hold anymore in the case of noisy decoders. First, the decoding error probability is always bounded above zero if $p_a > 0$ (see Proposition 1), because of the noise in the *a posteriori* computation. Second, the decoding error probability has a more unpredictable behavior. In particular, it does not always converge when the number of iterations goes to infinity (see discussion in Section V). Following [3], we define in the next section the notions of useful decoder and target error rate threshold.

D. Useful Region 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). 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 $P_e^{(\ell)}(\chi)$. Furthermore, if the following limit exists, we denote $P_e^{(\infty)}(\chi) = \lim_{\ell \to \infty} P_e^{(\ell)}(\chi)$.

1) Useful Region: The useful region is defined as the set of channel and hardware parameters yielding an asymptotic probability of error less than the *input error probability*. The latter probability is given by $P_e^{(0)}(\chi) = \sum_{z=-Q}^{-1} C(z) + \frac{1}{2}C(0)$, where C is the probability mass function of the quantized a priori information of the decoder ($\gamma = \mathbf{q}(y)$, see Algorithm 1). Then the decoder is said to be *useful* if $P_e^{(\infty)}(\chi)$ exists and

$$P_e^{(\infty)}(\chi) < P_e^{(0)}(\chi) \tag{10}$$

The ensemble of parameters that satisfy this condition constitutes the *useful region* of the decoder. The useful region indicates what are the faulty hardware conditions and the maximum channel noise such that the decoder can *reduce* the bit error probability (even though it does not indicate how much the bit error probability can be reduced).

2) Target Error Rate Threshold: For a target error probability η, the η-threshold is defined as the maximum channel noise such that the decoder can reduce the bit error probability below η:

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

E. Functional Threshold

Although the η-threshold definition allows determining the maximum channel noise for which the bit error probability can be reduced below a target value, there is not significant change in the behavior of the decoder when the channel noise parameter χ increases beyond the value of $\chi^*(\eta)$. In this section, a new threshold definition is introduced in order to identify the channel and hardware parameters yielding to a sharp change in the decoder behavior, similar to the change that occurs around the threshold of the noiseless decoder. This threshold will be referred to as the *functional threshold*. The aim is to detect a sharp increase (e.g., discontinuity) in the error probability of the noisy decoder, when λ goes beyond this functional threshold value. The threshold definition we propose make use of the Lipschitz constant of the function $\chi \mapsto P_e^{(\infty)}(\chi)$ in order to detect a sharp change of $P_e^{(\infty)}(\chi)$ with respect to χ. The definition of the Lipschitz constant is first restated for the sake of clarity.

Definition 1: Let $f : I \to \mathbb{R}$ be a function defined on an interval $I \subseteq \mathbb{R}$. The *Lipschitz constant* of f in I is defined as

$$L(f, I) = \sup_{x \neq y \in I} \frac{|f(x) - f(y)|}{|x - y|} \in \mathbb{R}_+ \cup \{+\infty\} \tag{12}$$

For $a \in I$ and $\delta > 0$, let $I_a(\delta) = I \cap (a - \delta, a + \delta)$. The *(local) Lipschitz constant* of f in $a \in I$ is defined by:

$$L(f, a) = \inf_{\delta > 0} L(f, I_a(\delta)) \in \mathbb{R}_+ \cup \{+\infty\} \tag{13}$$

Note that if a is a discontinuity point of f, then $L(f, a) = +\infty$. On the opposite, if f is differentiable in a, then the Lipschitz constant in a corresponds to the absolute value of the derivative. Furthermore, if $L(f, I) < +\infty$, then f is uniformly continuous on I and almost everywhere differentiable. In this case, f is said to be *Lipschitz continuous* on I.

The functional threshold is then defined as follows.

Definition 2: For given hardware parameters and a channel parameter χ, the decoder is said to be *functional* if

(a) The function $x \mapsto P_e^{(\infty)}(x)$ is defined on $[0, \chi]$
(b) $P_e^{(\infty)}$ is Lipschitz continuous on $[0, \chi]$
(c) $L\left(P_e^{(\infty)}, x\right)$ is an increasing function of $x \in [0, \chi]$

Then, the functional threshold $\bar{\chi}$ is defined as:

$$\bar{\chi} = \sup\{\chi \mid \text{conditions } (a), (b) \text{ and } (c) \text{ are satisfied}\} \tag{14}$$

The use of the Lipschitz constant allows a rigorous definition of the functional threshold, while avoiding the use of the derivative (which would require $P_e^{(\infty)}(\lambda)$ to be a piecewise differentiable function of λ). As it will be further illustrated in Section V, the functional threshold corresponds to a transition between two modes. The first mode corresponds to the channel parameters leading to a low level of error probability, *i.e.*, for which the decoder can correct most of the errors from the channel. In the second mode, the channel parameters lead to a much higher error probability level. If $L\left(P_e^{(\infty)}, \bar{\chi}\right) = +\infty$, then $\bar{\chi}$ is a discontinuity point of $P_e^{(\infty)}$ and the transition between the two levels is sharp. If $L\left(P_e^{(\infty)}, \bar{\chi}\right) < +\infty$, then $\bar{\chi}$ is an inflection point of $P_e^{(\infty)}$ and the transition is smooth. With the Lipschitz constant, one can characterize the transition in both cases. However, the second case corresponds to a degenerated one, in which the hardware noise is too high and leads to a non-standard asymptotic behavior of the decoder. That is why a set of admissible hardware noise parameters is defined as follows.

Definition 3: The set of *admissible hardware parameters* is the set of hardware noise parameters (p_a, p_c, p_x) for which $L\left(P_e^{(\infty)}, \bar{\chi}\right) = +\infty$.

In the following, as each threshold definition helps at illustrating different effects, one or the other definition will be used, depending on the context.

V. 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:

$$\mathbf{q}_\mu(-1) = -\mu \text{ and } \mathbf{q}_\mu(+1) = +\mu \tag{15}$$

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. Unconventional Behavior of the Min-Sum Decoder ($\mu = 1$)

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.

In noiseless decoder 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 above 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 sign-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 Appendix A), for $\ell \geq 0$. In the noiseless case, we observed by simulation that $\widetilde{C}^{(\ell)}$ reaches a fixed point of the density evolution for $\ell \approx 20$. In the noisy case, $\widetilde{C}^{(\ell)}$ evolves virtually the same as in the noiseless case during the first iterations. However, for $\ell > 20$, the noise present in the adder results in a progressive perturbation of $\widetilde{C}^{(\ell)}$, which eventually allows the decoder to escape from the fixed point attractor.

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 region for the sign-preserving 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 not converge. The region shaded in brown in Figure 2 is the *non-convergence region* of the decoder.

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 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. 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 $\mu = 3$.

Remark: For the noiseless decoder, numerical results obtained by density evolution show that $\mu = 6$ is the best choice of the channel scale factor, for any $d_v \in \{3, 4\}$ and any $d_v < d_c \leq 20$.

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* $\mu = 6$.

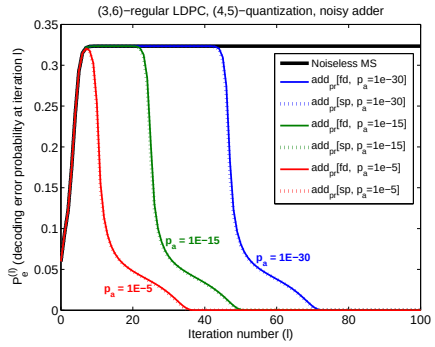


Figure 1. Effect of the noisy adder on the asymptotic performance of the MS decoder ($p_0 = 0.06$)

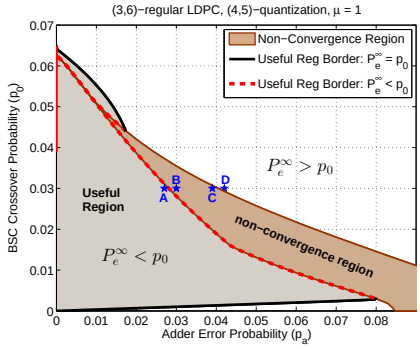


Figure 2. Useful and non-convergence regions of the MS decoder with sign-preserving noisy adder

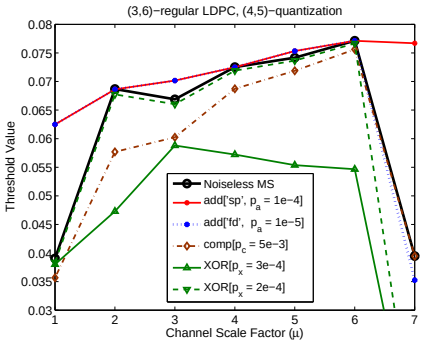


Figure 3. Threshold values of noiseless and noisy MS decoders with various channel scale factors

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 ($p_x = p_c = 0$). These regions are represented in Figure 4 and Figure 5, 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 V-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 actually correspond to the functional threshold defined in Section IV. We note that it does not hide a periodic (non-convergent) behavior, but it is due to the occurrence of an *early plateau phenomenon* in the convergence of $(P_e^{(\ell)})_\ell$ (for details see [16]).

In Figure 6, 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 $p_0^{(\mathrm{FT})}$, corresponding respectively to the values of p_0 on the upper-border of the useful region, and on the discontinuity line. Hence, $p_0^{(\mathrm{FT})}$ corresponds to the functional threshold $\bar{\chi}$ from Definition 2. It coincides with the classical threshold in the noiseless case, and can be seen as an appropriate generalization of the classical threshold to the case of noisy decoders. A similar behavior can be observed for the full-depth noisy adder. Moreover, from Figure 4 and Figure 5, it can be seen that as p_a goes to zero, the functional threshold value (red line) converges to 0.077, which is the classical threshold value of the noiseless decoder. In the following, the region located below the discontinuity line will be referred to as the ***functional region***.

Finally, we note that the functional threshold phenomenon can also be observed for the MS decoder with noisy XOR-operator ($p_x > 0$) or noisy comparator ($p_c > 0$), although not illustrated in this paper due to space limitations. Similar to the noisy adder case, when p_x and p_c go to zero, the functional threshold value converges to the threshold value of the noiseless decoder. It is also worth noting that the MS decoder with noisy comparator exhibits actually a classical threshold phenomenon, *i.e.*, $P_e^{(\infty)} = 0$ for any point in the functional region. This is explained by the fact that if the crossover probability of the channel is small enough, 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. For more details, the reader may refer to [16].

D. Comparison of the Impact of the Different Noisy Components ($\mu = 6$)

To compare the impact of the different noisy components on the decoder performance, the useful region and the η-threshold regions have been plotted for a channel parameter value close to, but slightly below the functional threshold: $p_0 = 0.07$ (this value is fixed throughout this section). Sign-preserving adders are considered in Figure 7 and full-depth adders in Figure 8.

In Figure 7(a) comparators are assumed to be noiseless ($p_c = 0$) and the regions are plotted with respect to p_a and p_x. While for the useful region the maximum admissible value of p_x is slightly less than the maximum admissible value of p_a, they tend to become increasingly closer as the the target η value decreases. Hence, we conclude that the sign-preserving noisy adder and the noisy XOR-operator have comparable impact on the MS decoder performance, especially for target bit error rates below 10^{-6}.

In Figure 7(b) XOR operators are assumed to be noiseless ($p_x = 0$) and the regions are plotted with respect to p_a and p_c. It can be seen that the maximum admissible value of p_c is significantly above the the maximum admissible value of p_a, especially for low η-values, which confirms that the MS decoder performance is much more sensitive to adder noise than to comparator noise. Moreover, it can be observed that the η-regions are delimited on the top by the same curve, given by $p_c = 0.023$ for $p_a < 10^{-4}$. This actually corresponds to

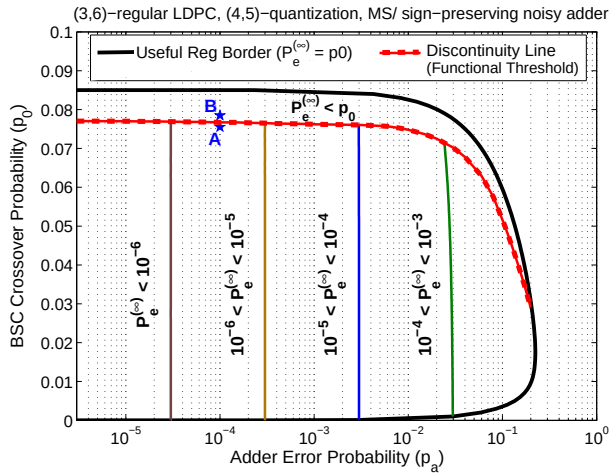


Figure 4. Regions of the MS decoder with sign-preserving noisy adder

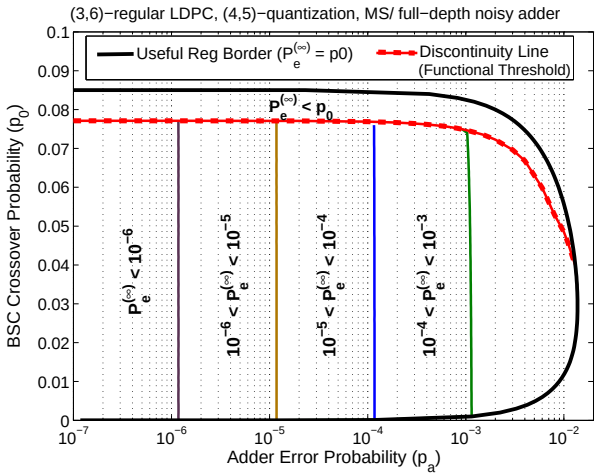
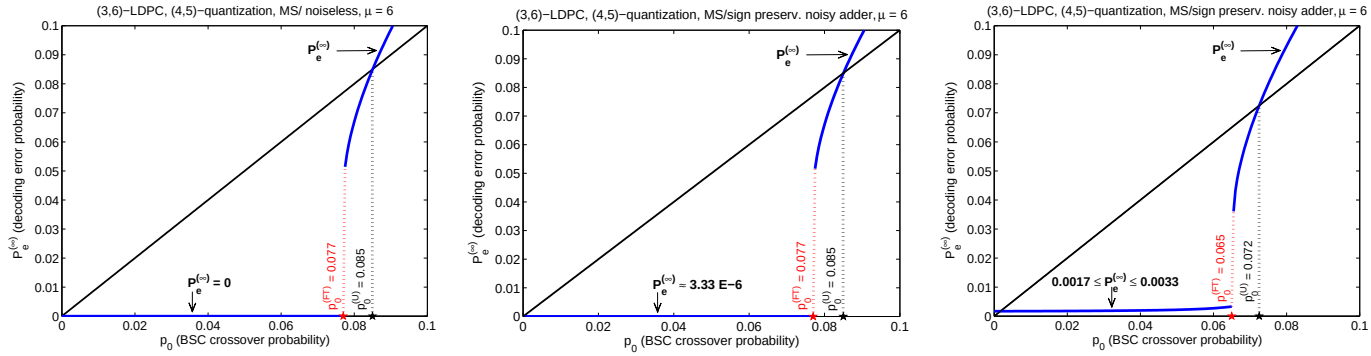
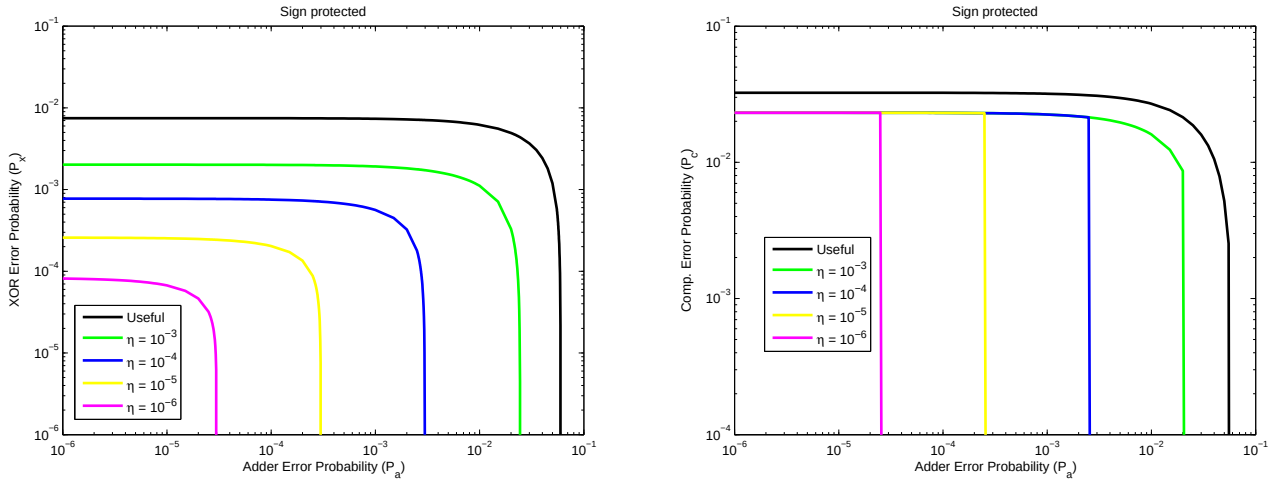


Figure 5. Regions of the MS decoder with full-depth noisy adder



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

Figure 6. Asymptotic error probability $P_e^{(\infty)}$ as a function of p_0; noiseless and noisy MS with sign-preserving noisy adder

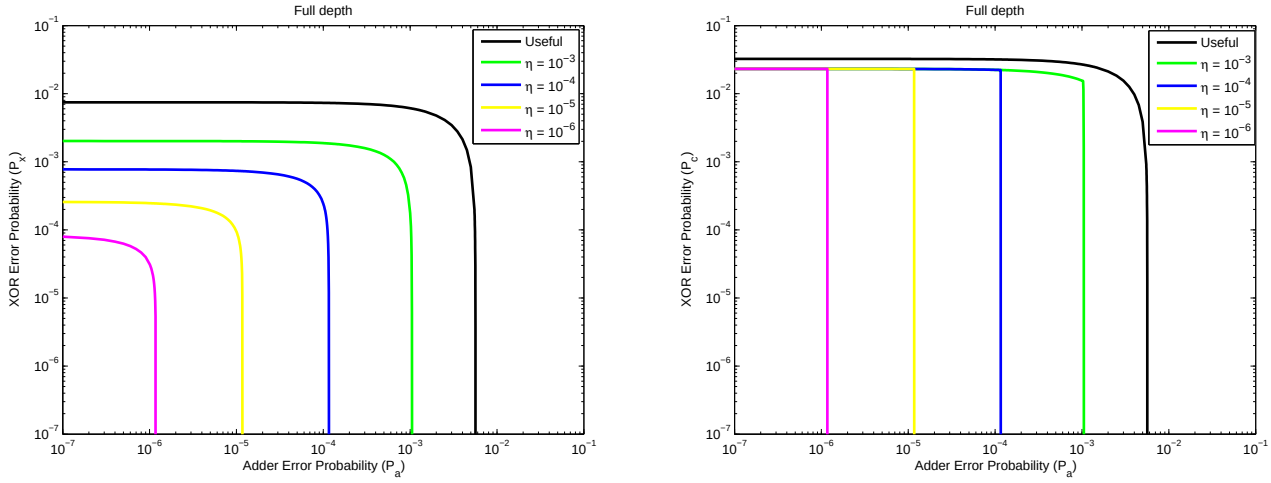


(a) Noisy XOR-operator and noiseless comparator (b) Noisy comparator and noiseless XOR-operator

Figure 7. Useful and η-threshold regions for $p_0 = 0.07$ with sign-preserving noisy adder

the threshold p_c-value of the noisy-comparator MS decoder, for a channel crossover probability $p_0 = 0.07$ (as discussed in Section V-C, the MS decoder with noisy comparator exhibits a classical threshold phenomenon).

Figure 8 presents a similar analysis, when considering the full-depth noisy-adder model. It can be seen that the maximum admissible value of p_a is less than the maximum admissible values of both p_c and p_x, which shows that the full-depth noisy



(a) Noisy XOR-operator and noiseless comparator

(b) Noisy comparator and noiseless XOR-operator

Figure 8. Useful and η-threshold regions for $p_0 = 0.07$ with full-depth noisy adder

adder is the most critical component of the noisy MS decoder.

VI. 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 [17] 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 either 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 the *noiseless* early stopping criterion may significantly improve the bit error rate performance of the *noisy* decoder in the error floor region (see Section VI-C).

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 9 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). These decoders are implemented in floating point, and the input LLR values are computed, as usual, by $\gamma_n = y_n \log\left((1-p)/p\right)$, with $y_n \in \pm 1$.

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 V-B concerning the channel scale factor optimization.

C. Error floor performance

Surprisingly, the BER curves of the noisy decoders from Figure 9 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 4). 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 $\frac{1}{2\widetilde{Q}} p_a N = 0.034$ (for $N = 1008$), and the probability of not having any erroneous bit within one iteration approaches

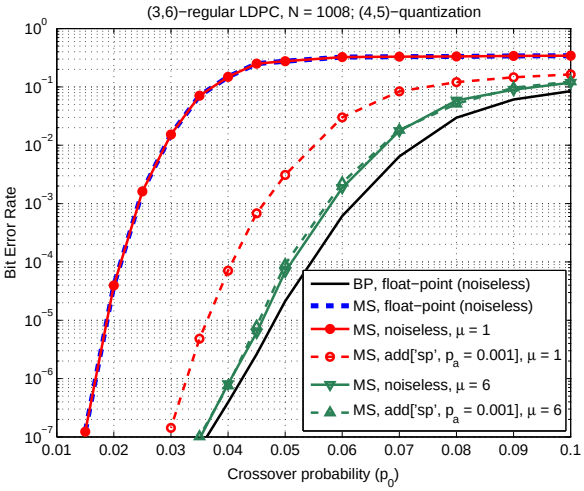


Figure 9. BER performance of noiseless and noisy MS decoders with various channel scale factors

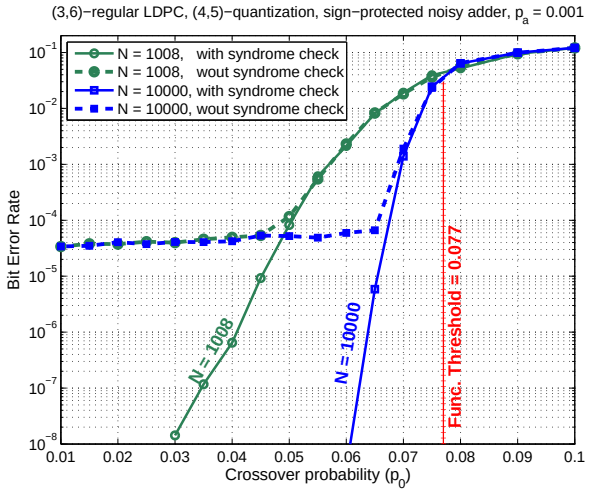


Figure 10. BER performance with and without early stopping criterion (sign-preserving noisy adder, $p_a = 0.001$)

$\left(1 - \frac{1}{2\widetilde{Q}} p_a\right)^N = 0.967$. 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 10 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 the case where 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.

While the noiseless early stopping criterion allows eliminating the error floor caused by the noisy components of the decoder, the price to pay might be a possible increase in the number of decoding iterations. However, our simulation results show that for the noisy decoders presented in Figure 10, the average number of decoding iterations increases by less than 0.5 iterations with respect to the noiseless decoders. Indeed, when the noiseless decoder completes, the probability of not having any erroneous bit within one iteration of the noisy decoder is sufficiently high, approaching 0.967 (for $N = 1008$), as discussed above. While this value is reflecting in the error floor performance, it has a negligible impact on the expected number of extra iterations to be run before the noiseless syndrome check step detects an error-free iteration.

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 $p_a = p_c \geq p_x$. The BER performance is evaluated for both sign-preserving and full-depth noisy adder error models. Simulation results are presented in Figures 11–12. 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. However, 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 be seen that for $p_c = p_a \geq 0.005$, the noisy decoder cannot achieve bit error rates below 10^{-2}.

CONCLUSION

This paper investigated the asymptotic and finite length behavior of the noisy MS decoder. 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.

Note: An extended version of this paper, including results for both BSC and BI-AWGN channels, is available at http://arxiv.org/abs/1405.6594.

APPENDIX A
DENSITY EVOLUTION

In this appendix we derive density evolution equations for the noisy finite-precision MS decoding for a regular (d_v, d_c)

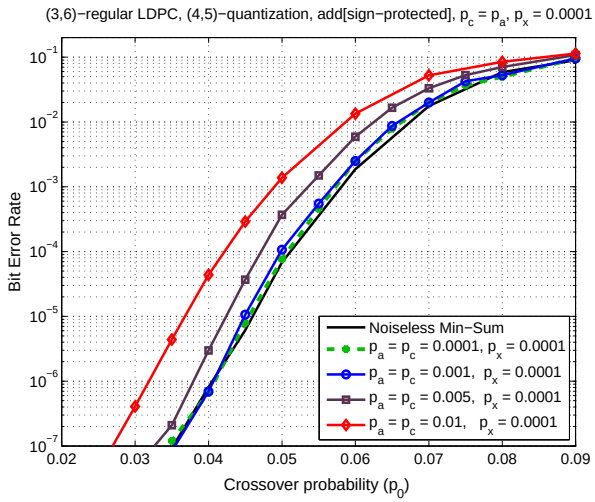


Figure 11. BER performance, noisy MS, sign-preserving noisy adder, $p_c = p_a$, $p_x = 0.0001$

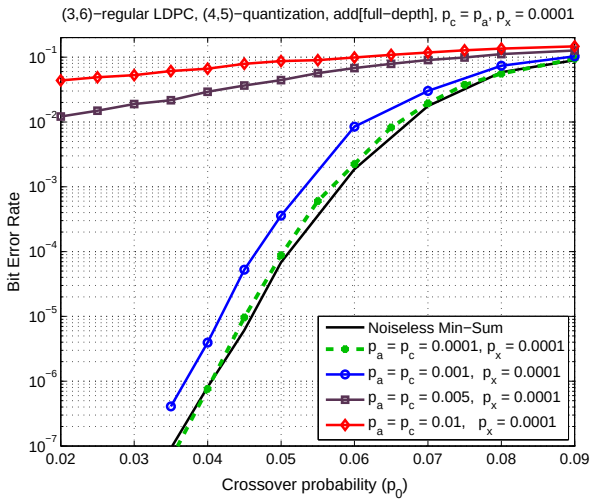


Figure 12. BER performance, noisy MS, full-depth noisy adder, $p_c = p_a$, $p_x = 0.0001$

LDPC code. The study can be easily generalized to irregular LDPC codes, simply by averaging according to the degree distribution polynomials.

Let $\ell > 0$ denote the decoding iteration. Superscript (ℓ) will be used to indicate the messages and the a posteriori information computed at iteration ℓ. To indicate the value of a message on a randomly selected edge, we drop the variable and check node indexes from the notation (and we proceed in a similar manner for the a priori and a posteriori information). The corresponding probability mass functions are denoted as follows.

$$\begin{aligned} C(z) &= \Pr(\gamma = z), & \forall z \in \mathcal{M} \\ \widetilde{C}^{(\ell)}(\tilde{z}) &= \Pr(\tilde{\gamma}^{(\ell)} = \tilde{z}), & \forall \tilde{z} \in \widetilde{\mathcal{M}} \\ A^{(\ell)}(z) &= \Pr\left(\alpha^{(\ell)} = z\right), & \forall z \in \mathcal{M} \\ B^{(\ell)}(z) &= \Pr\left(\beta^{(\ell)} = z\right), & \forall z \in \mathcal{M} \end{aligned}$$

1) Expression of the input probability mass function C: The probability mass function C depends only on the channel and the quantization map $\mathbf{q} : \mathcal{Y} \to \mathcal{M}$, where $\mathcal{Y}$ denotes the channel output alphabet. We also note that for $\ell = 0$, we have $A^{(0)} = C$.

For the BSC, the channel output alphabet is $\mathcal{Y} = \{-1, +1\}$. For a positive integer $\mu \leq Q$, the quantization map $\mathbf{q}_\mu$ is defined by $\mathbf{q}_\mu(y) = \mu \cdot y$, $\forall y \in \mathcal{Y} = \{-1, +1\}$. Considering the all-zero $(+1)$ codeword assumption, the probability mass function C can be computed as follows (where ε is the crossover probability of the BSC):

$$C(z) = \begin{cases} 1-\varepsilon, & \text{if } z = \mu \\ \varepsilon, & \text{if } z = -\mu \\ 0, & \text{otherwise} \end{cases} \tag{16}$$

2) Expression of $B^{(\ell)}$ as a function of $A^{(\ell-1)}$:

In the sequel, we make the convention that $\Pr(\text{sgn}(0) = 1) = \Pr(\text{sgn}(0) = -1) = 1/2$. The following notation will be used:

- $A_{[x,y]} = \sum_{z=x}^{y} A(z)$, for $x \leq y \in \mathcal{M}$
- $A_{[0^+,y]} = \frac{1}{2}A(0) + \sum_{z=1}^{y} A(z)$, for $y \in \mathcal{M},\ y > 0$
- $A_{[x,0^-]} = \frac{1}{2}A(0) + \sum_{z=x}^{-1} A(z)$, for $x \in \mathcal{M},\ x < 0$

For the sake of simplicity, we drop the iteration index, thus $B := B^{(\ell)}$ and $A := A^{(\ell-1)}$. We proceed by recursion on $i = 2, \ldots, d_c - 1$, where d_c denotes the check-node degree. Let $\beta_1 := \alpha_1$, and for $i = 2, \ldots, d_c - 1$ define:

$$\beta_i = \mathbf{x}_{\text{pr}}(\text{sgn}(\beta_{i-1}), \text{sgn}(\alpha_i))\mathbf{m}_{\text{pr}}(|\beta_{i-1}|, |\alpha_i|)$$

Let also B_{i-1} and B_i denote the probability mass functions of β_{i-1} and β_i, respectively (hence, $B_1 = A$).

First of all, for $z = 0$, we have:

$$\begin{aligned} B_i(0) = \Pr(\beta_i = 0) &= A(0)B_{i-1}(0) + \\ &\left[B_{i-1}(0)(1 - A(0)) + A(0)(1 - B_{i-1}(0))\right](1 - p_c) \end{aligned}$$

For $z \neq 0$, we proceed in several steps as follows:

For $z > 0$:

$$\begin{aligned} F'_i(z) &\stackrel{\text{def}}{=} \Pr(\beta_i \geq z \mid p_x = 0) \\ &= \left[B_{i-1}{}_{[0^+,z-1]}A_{[z,Q-1]} + A_{[0^+,z-1]}B_{i-1}{}_{[z,Q-1]}\right] p_c \\ &+ \left[B_{i-1}{}_{[1-z,0^-]}A_{[-Q,-z]} + A_{[1-z,0^-]}B_{i-1}{}_{[-Q,-z]}\right] p_c \\ &+ B_{i-1}{}_{[z,Q-1]}A_{[z,Q-1]} + B_{i-1}{}_{[-Q,-z]}A_{[-Q,-z]} \\ F_i(z) &\stackrel{\text{def}}{=} \Pr(\beta_i \geq z) \\ &= (1 - p_x).F'_i(z) + p_x.G'_i(-z) \\ B_i(z) &= \Pr(\beta_i = z) = F_i(z) - F_i(z+1) \end{aligned}$$

For $z < 0$:

$$\begin{aligned} G'_i(z) &\stackrel{\text{def}}{=} \Pr(\beta_i \leq z \mid p_x = 0) \\ &= \left[B_{i-1}{}_{[0^+,-z-1]}A_{[-Q,z]} + A_{[0^+,-z-1]}B_{i-1}{}_{[-Q,z]}\right] p_c \\ &+ \left[B_{i-1}{}_{[-z,Q-1]}A_{[z+1,0^-]} + A_{[-z,Q-1]}B_{i-1}{}_{[z+1,0^-]}\right] p_c \\ &+ B_{i-1}{}_{[-z,Q-1]}A_{[-Q,z]} + A_{[-z,Q-1]}B_{i-1}{}_{[-Q,z]} \\ G_i(z) &\stackrel{\text{def}}{=} \Pr(\beta_i \geq z) \\ &= (1 - p_x).G'_i(z) + p_x.F'_i(-z) \\ B_i(z) &= \Pr(\beta_i = z) = G_i(z) - G_i(z+1) \end{aligned}$$

Finally, we have that $B = B_{d_c-1}$.

3) Expression of $A^{(\ell)}$ as a function of $B^{(\ell)}$ and C: We derive at the same time the *expression of $\widetilde{C}^{(\ell)}$ as a function of $B^{(\ell)}$ and C.*

For simplicity, we drop the iteration index, so $A := A^{(\ell)}$, $B := B^{(\ell)}$, and $\widetilde{C} := \widetilde{C}^{(\ell)}$. We denote by $\imath : \widetilde{\mathcal{M}} \times \mathcal{E} \to \widetilde{\mathcal{M}}$ the error injection model (either full-depth or sign-preserving) used to define the noisy adder. We decompose each noisy addition into three steps (noiseless infinite-precision addition, saturation, and error injection), and proceed by recursion on $i = 0, 1, \ldots, d_v$, where d_v denotes the variable-node degree:

- For $i = 0$:
$$\Omega_0 \stackrel{\text{def}}{=} \gamma \in \mathcal{M} \subseteq \widetilde{\mathcal{M}},$$
$$\widetilde{C}_0(\tilde{z}) \stackrel{\text{def}}{=} \Pr(\Omega_0 = \tilde{z}) = \begin{cases} C(\tilde{z}), & \text{if } \tilde{z} \in \mathcal{M} \\ 0, & \text{if } \tilde{z} \in \widetilde{\mathcal{M}} \setminus \mathcal{M} \end{cases}$$
- For $i = 1, \ldots, d_v$:
$$\omega_i \stackrel{\text{def}}{=} \Omega_{i-1} + \beta_{m_i,n} \in \mathbb{Z},$$
$$c_i(w) \stackrel{\text{def}}{=} \Pr(\omega_i = w) = \sum_u \widetilde{C}_{i-1}(u) B(w-u), \forall w \in \mathbb{Z}$$
$$\tilde{\omega}_i \stackrel{\text{def}}{=} \mathbf{s}_{\widetilde{\mathcal{M}}}(\omega_i) \in \widetilde{\mathcal{M}},$$
$$\tilde{c}_i(\tilde{w}) \stackrel{\text{def}}{=} \Pr(\tilde{\omega}_i = \tilde{w}) = \begin{cases} c_i(\tilde{w}), & \text{if } \tilde{w} \in \widetilde{\mathcal{M}} \setminus \{\pm\widetilde{Q}\} \\ \sum_{w \le -\widetilde{Q}} c_i(w), & \text{if } \tilde{w} = -\widetilde{Q} \\ \sum_{w \ge +\widetilde{Q}} c_i(w), & \text{if } \tilde{w} = +\widetilde{Q} \end{cases}$$
$$\Omega_i \stackrel{\text{def}}{=} \imath(\tilde{\omega}_i, e) \in \widetilde{\mathcal{M}},$$
$$\widetilde{C}_i(\tilde{z}) \stackrel{\text{def}}{=} \Pr(\Omega_i = \tilde{z}) = \sum_{\tilde{\omega}} \sum_e \delta^{\tilde{z}}_{\imath(\tilde{\omega},e)} p_{\mathcal{E}}(e) \tilde{c}_i(\tilde{\omega}), \forall \tilde{z} \in \widetilde{\mathcal{M}}$$
where $\delta^y_x = 1$ if $x = y$, and $\delta^y_x = 0$ if $x \neq y$.

Note that in the definition of Ω_i above, e denotes an error drawn from the error set $\mathcal{E}$ according to the error probability distribution $p_{\mathcal{E}}$.

Finally, we have: $A = \mathbf{s}_{\mathcal{M}}\left(\widetilde{C}_{d_v-1}\right)$ and $\widetilde{C} = \widetilde{C}_{d_v}$.

Note that applying the saturation operator $\mathbf{s}_{\mathcal{M}}$ on the probability mass function $\widetilde{C}_{d_v-1}$ means that all the probability weights corresponding to values $\tilde{w}$ outside $\mathcal{M}$ must be accumulated to the probability of the corresponding boundary value of $\mathcal{M}$ (that is, either $-Q$ or $+Q$, according to whether $\tilde{w} < -Q$ or $\tilde{w} < +Q$).

Remark: For a noisy adder defined by the full-depth or the sign preserving error injection models defined in Section II, the third equation from the above recursion (expression of $\widetilde{C}_i$ as a function of $\tilde{c}_i$) may be rewritten as follows:

Sign-preserving bitwise-XORed noisy adder

$$\widetilde{C}_i(\tilde{z}) = \begin{cases} (1-p_a)\tilde{c}_i(\tilde{z}) + \frac{1}{\widetilde{Q}} p_a \left(\tilde{c}_{i\,[\le 0^-]} - \tilde{c}_i(z)\right), & \text{if } \tilde{z} < 0 \\ (1-p_a)\tilde{c}_i(0) + \frac{1}{\widetilde{Q}} p_a \left(1 - \tilde{c}_i(0)\right), & \text{if } \tilde{z} = 0 \\ (1-p_a)\tilde{c}_i(\tilde{z}) + \frac{1}{\widetilde{Q}} p_a \left(\tilde{c}_{i\,[\ge 0^+]} - \tilde{c}_i(z)\right), & \text{if } \tilde{z} > 0 \end{cases} \quad (17)$$

where $\tilde{c}_{i\,[\le 0^-]} = \sum_{\tilde{\omega}<0} \tilde{c}_i(\tilde{\omega}) + \frac{1}{2}\tilde{c}_i(0)$, and $\tilde{c}_{i\,[\ge 0^+]} = \frac{1}{2}\tilde{c}_i(0) + \sum_{\tilde{\omega}>0} \tilde{c}_i(\tilde{\omega})$.

Full-depth bitwise-XORed noisy adder

$$\widetilde{C}_i(\tilde{z}) = (1-p_a)\tilde{c}_i(\tilde{z}) + \frac{1}{2\widetilde{Q}} p_a \left(1 - \tilde{c}_i(\tilde{z})\right) \quad (18)$$

Finally, we note that the density evolution equations for the noiseless finite-precision MS decoder can be obtained by setting $p_a = p_c = p_x = 0$.

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

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

[5] 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.

[6] 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.

[7] C.-H. Huang, Y. Li, and L. Dolecek, "Gallager B LDPC decoder with transient and permanent errors," in *Proc. IEEE Int. Symp. on Inf. Theory (ISIT)*, July 2013, pp. 3010–3014.

[8] ——, "Gallager B LDPC decoder with transient and permanent errors," *IEEE Trans. on Comm.*, vol. 62, no. 1, pp. 15–28, January 2014.

[9] C.-H. Huang and L. Dolecek, "Analysis of finite-alphabet iterative decoders under processing errors," in *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, May 2013.

[10] E. Dupraz, D. Declercq, V. Vasic, and V. Savin, "Finite alphabet iterative decoders robust to faulty hardware: Analysis and selection," in *Proc. IEEE Int. Symp. on Turbo Codes and Iterative Inf. Processing (ISTC)*, August 2014.

[11] A. Balatsoukas-Stimming, C. Studer, and A. Burg, "Characterization of min-sum decoding of LDPC codes on unreliable silicon," in *Inf. Theory and Applications Workshop (ITA)*, 2014.

[12] A. Balatsoukas-Stimming and A. Burg, "Density evolution for min-sum decoding of LDPC codes under unreliable message storage," *IEEE Comm. Letters*, vol. PP, no. 99, pp. 1–4, 2014.

[13] C. L. Kameni Ngassa, V. Savin, and D. Declercq, "Min-sum-based decoders running on noisy hardware," in *proc. of IEEE Global Comm. Conf. (GLOBECOM)*, 2013.

[14] C. L. Ngassa, V. Savin, and D. Declercq, "Analysis of min-sum based decoders implemented on noisy hardware," in *Proc. Asilomar Conf. on Signals, Systems and Computers*, 2013.

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

[16] C. L. Kameni Ngassa, V. Savin, and D. Declercq, "Unconventional behavior of the noisy min-sum decoder over the binary symmetric channel," in *Proc. Inf. Theory and Applications Workshop (ITA)*, 2014.

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