Analysis and Design of Finite Alphabet Iterative Decoders Robust to Faulty Hardware

Elsa DUPRAZ, David DECLERCQ, Bane VASIĆ and Valentin SAVIN

Abstract

This paper addresses the problem of designing LDPC decoders for robustness to transient errors introduced by a faulty hardware. The Finite Alphabet Iterative Decoder (FAID) framework enables to define a large variety of decoders with common properties but potentially different abilities of robustness to errors introduced by the hardware. Under faulty hardware symmetric error models, Density Evolution equations are derived to obtain the asymptotic error probabilities of the noisy FAIDs. Furthermore, a new noisy threshold definition is introduced to characterize accurately the convergence behavior of the decoders. From this definition, we illustrate the existence of robust and non-robust FAIDs and propose a framework for the design of naturally robust decoders. Finite-length simulations illustrate the gain at considering robust FAIDs on faulty hardware.

This work was funded by the Seventh Framework Programme of the European Union, under Grant Agreement number 309129 (i-RISC project), and by the NSF under grants CCF-0963726 and CCF-1314147.

E. Dupraz and D. Declercq are with the ETIS lab, ENSEA/Université de Cergy-Pontoise/CNRS UMR 8051, 95014 Cergy-Pontoise, France (e-mail:elsa.dupraz@ensea.fr; declercq@ensea.fr).

B. Vasić is with the Department of Electrical and Computer Engineering, University of Arizona, Tucson, AZ, 85721 USA (e-mail: vasic@ece.arizona.edu).

V. Savin is with the CEA-LETI, Minatec Campus, 38000 Grenoble, France (e-mail:valentin.savin@cea.fr)

I. Introduction

Reliability is becoming a major issue in the design of modern electronic devices. Huge increase in the integration factors coupled with important reduction of the chip sizes make them much more sensitive to noise and may induce transient errors during operations performed by a circuit. Furthermore, the involved delicate fabrication process make hardware components more prone to defects and may also cause permanent computation errors. As a consequence, in the context of communication and storage, errors may not only come from transmission channels, but also from the faulty hardware used in transmitters and receivers.

The general problem of reliable function computation using faulty gates was first addressed by von Neumann in [1] and the notion of redundancy was later considered in [2]–[4]. The redundancy is the number of noisy gates required for reliable function computation divided by the number of noiseless gates needed for the same function computation. Gács and Gál [2] and Dobrushin and Ortyukov [3], respectively, provided lower and upper bounds on the redundancy for reliable Boolean function computation from faulty gates. Pippenger [4] showed that finite asymptotic redundancy can be achieved when using Low Density Parity Check (LDPC) codes for the reliable computation of linear Boolean functions. Taylor [5] and Kuznetsov [6] considered memories as a particular instance of this problem and provided an analysis of a memory architecture based on LDPC decoders made of faulty components. More recently, an equivalence between the architecture proposed by Taylor and a noisy Gallager-B decoder was identified by Vasic et al. [7], while Chilappagari et al. [8] analyzed a memory architecture based on one-step majority logic decoders.

As a consequence, there is a need to address the problem of constructing reliable LDPC decoders made of faulty components not only for error correction on faulty hardware, but also as a first step in the context of reliable function computation and storage. Formulating a general method for construction of robust decoders requires understanding if a particular decoder is

inherently robust to errors introduced by the faulty hardware. There is also a need for a rigorous analysis to determine which characteristics of decoders make them robust.

To answer to the first point, Varshney [9] introduced a framework referred to as noisy Density Evolution (noisy-DE) for the performance analysis of noisy LDPC decoders in terms of asymptotic error probability. Based on this framework, the asymptotic performance of a variety of noisy LDPC decoders was analyzed. In [9], infinite precision BP decoders were investigated, which is not useful for actual implementation on faulty hardware. On the contrary, noisy practically important hard-decision decoders, such as noisy Gallager-A [9] and Gallager-E [10] decoders were considered. Gallager-B decoders were analyzed for binary [7], [11], [12] and non-binary [13] alphabets under transient error models, and [12] also considered permanent error models. From the same noisy-DE framework, [14], [15] proposed an asymptotic analysis of the behavior of stronger discrete Min-Sum decoders, for which the exchanged messages are no longer binary but are quantized soft information represented by a finite (and typically small) number of bits.

Recently, a new class of LDPC decoders referred to as Finite Alphabet Iterative Decoders (FAIDs) has been introduced [16]. In these decoders, the messages take their values in small alphabets and the variable node update is derived through a predefined Boolean function. The FAID framework offers the possibility to define a large collection of these functions, each corresponding to a particular decoding algorithm. The FAIDs were originally introduced to address the error floor problem, and designed to correct error events located on specific small topologies of error events referred to as *trapping sets* that usual decoders (Min-Sum, BP-based) cannot correct. When operating on faulty hardware, the FAIDs may potentially exhibit very different properties in terms of tolerance to transient errors and we would like to identify the robust ones among the large diversity of decoders.

In this paper, we propose a rigorous method for the analysis of these properties and for the design of decoding rules robust to transient errors introduced by the hardware. The design procedure

we propose is based on an asymptotic performance analysis of noisy-FAIDs using noisy-DE. In order to characterize the asymptotic behavior of the FAIDs from the DE equations, we introduce a noisy-DE threshold definition referred to as the *functional threshold*. The functional threshold is different from the useful threshold [9] and from the target-BER threshold [9], [14], and relies on more stringent convergence conditions of the noisy DE recursion. The functional threshold thus gives a criterion for the comparison of the asymptotic performance of the decoders. Based on this criterion, we then propose a noisy-DE based framework for the design of decoders inherently more robust to errors introduced by the hardware. Finite-length simulations illustrate the gain in performance at considering robust FAIDs on faulty hardware.

The outline of the paper is as follows. Section II introduces the FAID framework. Section III presents the error models we consider for the faulty hardware. Section IV gives the noisy-DE equation and introduces the definition of the functional threshold. Section V presents the method for the design of robust decoders. Section VI gives the finite-length simulation results.

II. Finite Alphabet Iterative Decoders Running on Faulty Hardware

This section describes the general noiseless framework of FAIDs introduced in [16] and shows how this framework enables to define a large collection of decoders. In the following, we assume that the transmission channel is a Binary Symmetric Channel (BSC) with parameter α.

An N_s-level FAID is defined as a 4-tuple given by $\mathrm{D} = (\mathcal{M}, \mathcal{Y}, \Phi_v, \Phi_c)$. The message alphabet is finite and can be defined as $\mathcal{M} = \{-L_s, \ldots, -L_1, 0, L_1, \ldots, L_s\}$, where $L_i \in \mathbb{R}^+$ and $L_i > L_j$ for any $i > j$. It thus consists of $N_s = 2s+1$ levels to which the message values belong. For the BSC, the set $\mathcal{Y}$, which denotes the set of possible channel values, is defined as $\mathcal{Y} = \{\pm \mathrm{B}\}$, where $\mathrm{B} \in \mathcal{M}$. For the n-th symbol of the codeword, the channel value $y_n \in \mathcal{Y}$ corresponding to node v_n is determined based on its received value. Here, we use the mapping $0 \rightarrow \mathrm{B}$ and $1 \rightarrow -\mathrm{B}$. In the following, $\mu_1, \ldots, \mu_{d_c-1}$ denote the values of extrinsic incoming messages to a Check Node (CN) of degree d_c and let $\eta_1, \ldots, \eta_{d_v-1}$ be the values of extrinsic incoming messages to

a Variable Node (VN) of degree d_v.

At each iteration of the iterative decoding process, the following operations defined in [16] are performed on the messages. The Check Node Update (CNU) function $\Phi_c : \mathcal{M}^{d_c-1} \rightarrow \mathcal{M}$ is used for the message update at a Check Node (CN) of degree d_c. Let $\boldsymbol{\mu}$ denote an $|\mathcal{M}|$-ary (d_c-1)-tuple composed of extrinsic messages coming to a CN. The corresponding outgoing message is calculated as

$$\Phi_c(\boldsymbol{\mu}) = \left(\prod \text{sgn}(\boldsymbol{\mu})\right) \min(|\boldsymbol{\mu}|), \tag{1}$$

where sgn denotes the sign operator and all vector operators are performed componentwise on vector elements. Φ_c corresponds to the CNU of the standard Min-Sum decoding. The Variable Node Update (VNU) function $\Phi_v : \mathcal{M}^{d_v-1} \times \mathcal{Y} \rightarrow \mathcal{M}$ used for the update at a Variable Node (VN) v_n, $n = 0 \ldots N-1$ of degree d_v. Let $\boldsymbol{\eta}$ denote an $|\mathcal{M}|$-ary (d_v-1)-tuple composed of extrinsic messages coming to a VN. The corresponding outgoing message is calculated as

$$\Phi_v(\boldsymbol{\eta}, y_n) = Q\left(\sum \boldsymbol{\eta} + \omega_n \cdot y_n\right), \tag{2}$$

where the function $Q(.)$ is defined based on a threshold set $\mathscr{T} = \{T_i : 1 \leq i \leq s+1\}$ such that $T_i \in \mathbb{R}^+$ and $T_i > T_j$ if $i > j$, and $T_{s+1} = \infty$ and

$$Q(x) = \begin{cases} \text{sgn}(x)L_i, & \text{if } \ T_i \leq |x| < T_{i+1} \\ 0, & \text{otherwise} \end{cases}$$

The weight ω_i is computed from a symmetric function $\Omega : \mathcal{M}^{d_v-1} \rightarrow \mathbb{R}^+$ whose input arguments are the $d_v - 1$ incoming messages of the VNU. At the end of each decoding iteration, the *A Posteriori* Probability (APP) computation produces messages γ calculated from the function $\Phi_a : \mathcal{M}^{d_v} \times \mathcal{Y} \rightarrow \bar{\mathcal{M}}$, where $\bar{\mathcal{M}} = \{-L_{s'}, \ldots, L_{s'}\}$, $s' = 2s+1$. Let $\bar{\boldsymbol{\eta}}$ denote an $|\mathcal{M}|$-ary d_v-tuple composed of extrinsic messages coming to a VN. The function Φ_a is given by

$$\Phi_a(\bar{\boldsymbol{\eta}}, y_n) = \sum \bar{\boldsymbol{\eta}} + y_n \quad . \tag{3}$$

It is defined on a larger alphabet $\bar{\mathcal{M}}$ in order to limit the influence of saturation effects when calculating the sum. The hard-decision bit corresponding to each variable node v_n is given by

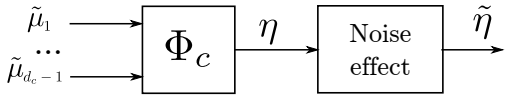


Fig. 1. Function decomposition for the CNU

the sign of the APP computation. If $\Phi_a(\boldsymbol{\eta}_a, y_n) = 0$, then the hard-decision bit is selected at random and takes value 0 with probability $1/2$.

The parameters T_i, L_i, and the function Ω provide degrees of freedom to construct the decoders. Alternatively, Φ_v can be represented as a Look-Up Table (LUT) that has to satisfy the properties of the VNU function and that is defined for a specific channel value. For instance, Table I shows an example of LUT for a 7-level FAID and column-weight three codes when the channel value is $-\mathrm{B}$. The corresponding LUT for the value $+\mathrm{B}$ can be deduced by symmetry. Classical decoders such as the standard Min-Sum and the offset Min-Sum can also be seen as instances of FAIDs. It indeed suffices to derive the specific LUT from the VNU functions of these decoders. Table II gives the VNU of the 7-level offset Min-Sum decoder. Therefore, the VNU formulation enables to define a large collection of decoders with common characteristics but potentially different robustness to noise. Note that determining what makes a particular decoder robust is highly non trivial. Before we proceed for describing a method for analyzing the asymptotic behavior of noisy-FAIDs, we introduce error models for the faulty hardware. This method enables us to compare decoder robustness for different choices of parameters for Φ_v and thus to design decoders robust to faulty hardware.

III. Error Models for the Faulty Hardware

We consider a faulty error model in which noise is introduced at a message level and appears only at the output of a function computation. More precisely, we assume that the noisy function can be decomposed as a noiseless function followed by some noise effect (see Figure 1 for the

TABLE I

LUT $\Phi_v^{(\text{OPT})}$ REPORTED IN [16] OPTIMIZED FOR THE ERROR FLOOR

m_1/m_2	$-L_3$	$-L_2$	$-L_1$	0	$+L_1$	$+L_2$	$+L_3$
$-L_3$	$-L_3$	$-L_3$	$-L_3$	$-L_3$	$-L_3$	$-L_3$	$-L_1$
$-L_2$	$-L_3$	$-L_3$	$-L_3$	$-L_3$	$-L_2$	$-L_1$	L_1
$-L_1$	$-L_3$	$-L_3$	$-L_2$	$-L_2$	$-L_1$	$-L_1$	L_1
0	$-L_3$	$-L_3$	$-L_2$	$-L_1$	0	0	L_1
L_1	$-L_3$	$-L_2$	$-L_1$	0	0	L_1	L_2
L_2	$-L_3$	$-L_1$	$-L_1$	0	L_1	L_1	L_3
L_3	$-L_1$	L_1	L_1	L_1	L_2	L_3	L_3

TABLE II

VNU OF A 3-BIT OFFSET MIN-SUM REPRESENTED AS A FAID

m_1/m_2	$-L_3$	$-L_2$	$-L_1$	0	$+L_1$	$+L_2$	$+L_3$
$-L_3$	$-L_3$	$-L_3$	$-L_3$	$-L_3$	$-L_3$	$-L_2$	$-L_1$
$-L_2$	$-L_3$	$-L_3$	$-L_3$	$-L_3$	$-L_2$	$-L_1$	0
$-L_1$	$-L_3$	$-L_3$	$-L_3$	$-L_2$	$-L_1$	0	0
0	$-L_3$	$-L_3$	$-L_2$	$-L_1$	0	0	0
L_1	$-L_3$	$-L_2$	$-L_1$	0	0	0	L_1
L_2	$-L_2$	$-L_1$	0	0	0	L_1	L_2
L_3	$-L_1$	0	0	0	L_1	L_2	L_3

case of the CNU). As a consequence, η, μ, and γ, represent the messages at the output of the noiseless CNU, VNU, and APP computation respectively, and their noisy versions are denoted $\tilde{\eta}$, $\tilde{\mu}$, $\tilde{\gamma}$. The noise effects at the end of Φ_c and Φ_v are represented by probability transition matrices $\Pi^{(v)}$ and $\Pi^{(c)}$ respectively, with

$$\Pi^{(c)}_{k,m} = \text{Pr}(\tilde{\eta} = m|\eta = k),\ \Pi^{(v)}_{k,m} = \text{Pr}(\tilde{\mu} = m|\mu = k) \tag{4}$$

wherein the matrix entries are indexed by the values in $\mathcal{M}$. This indexing is used for all the vectors and matrices introduced in the remaining of the paper. The noise effect on Φ_a is modeled by the probability transition matrix $\Pi^{(a)}$

$$\Pi^{(a)}_{k,m} = \text{Pr}(\tilde{\gamma} = m|\gamma = k) \tag{5}$$

where $k, m \in \bar{\mathcal{M}}$. The resulting noisy messages $\tilde{\mu}$, $\tilde{\eta}$, are then the inputs of the message updates Φ_c, Φ_v, and APP computation Φ_a. The forms of the probability transition matrices depend on the considered error models. Here, two different models are considered which exhibit quite different behaviors in conjunction with FAIDs.

Note that the above model is in essence a memoryless noise model, where the noise is added only at the output of the noiseless functions. A perhaps more relevant model could be to consider

noise effect introduced *inside* the functions, for example during elementary operations such as the minimum computation between two elements in Φ_c, as in [15]. The two models are not equivalent in general. Indeed, in our model, the noisy output message value depends only on the noiseless output message value, and not on the input values of the function. This is not true for all noisy functions. For instance, the noisy minimum function defined as

$$\tilde{\eta} = \begin{cases} \min(\mu_1, \mu_2) \text{ with probability } 1-p \\ \max(\mu_1, \mu_2) \text{ with probability } p, \end{cases} \tag{6}$$

does not satisfy this condition.

While the models introduced here may not capture all the noise effects, they are sufficient for the analysis of the behavior and robustness of noisy decoders without requiring knowledge of a particular hardware implementation. More accurate faulty hardware models will be considered in future works

A. Sign-Preserving error model

The first model has a Sign-Preserving (SP) property, meaning that noise is assumed to affect only on message amplitude, but not its sign. The values associated to each VN can be estimated from the sign of the related messages, and the amplitudes of the messages represent the reliability of the estimated values. As a consequence, such a model allows us to study the robustness of the decoders to errors on the reliability of the estimated value, and not on the estimated value itself. However, it assumes extra-protection at the hardware level in computing the sign. The probability transition matrices for the SP-Model can be constructed from a SP-transfer matrix defined as follows.

Definition 1: The SP-transfer matrix $\Pi^{(\mathrm{SP})}(p, s)$ is a matrix of size $(2s+1) \times (2s+1)$ such

that

$$\Pi^{(\text{SP})}_{k,k}(p,s) = 1-p, \quad \Pi^{(\text{SP})}_{k,0}(p,s) = \frac{p}{s}, \quad \Pi^{(\text{SP})}_{0,k}(p,s) = \frac{p}{2s}$$
$$\Pi^{(\text{SP})}_{k,m}(p,s) = \frac{p}{s}, \text{ for } m \neq k \neq 0, \text{ sign}(m) = \text{sign}(k)$$
$$0 \text{ elsewhere.} \tag{7}$$

According to this definition, a strictly positive message value can give only another positive message value with respect to a $(s+1)$-ary symmetric probability model of parameter p. The same holds for strictly negative values.

The matrices $\Pi^{(c)}$, $\Pi^{(v)}$, and $\Pi^{(a)}$ can be now obtained from $\Pi^{(\text{SP})}$ as a template. The noise level parameter at the output of Φ_c is given by the parameter p_c, and the corresponding probability transition matrix is given by $\Pi^{(c)} = \Pi^{(\text{SP})}(p_c, s)$. In the same way, the noise level parameters at the output of $\Phi^{(v)}$ and $\Phi^{(a)}$ are denoted p_v and p_a respectively, and the corresponding probability transition matrices are given by $\Pi^{(v)} = \Pi^{(\text{SP})}(p_v, s)$ and $\Pi^{(a)} = \Pi^{(\text{SP})}(p_a, s')$.

B. Full-Depth error model

The second model is called the Full-Depth (FD) model. This model is potentially more harmful than the SP-Model but does not require hardware sign-protection any more. The FD-transfer matrix is defined as follows.

Definition 2: The FD-transfer matrix $\Pi^{(\text{FD})}(p,s)$ is a matrix of size $(2s+1) \times (2s+1)$ such that

$$\Pi^{(\text{FD})}_{k,k}(p,s) = 1-p,$$
$$\Pi^{(\text{FD})}_{k,m}(p,s) = \frac{p}{s}, \text{ for } m \neq k. \tag{8}$$

The FD-transfer Matrix defines a $(2s+1)$-ary symmetric model of parameter p. The noise level parameters at the end of Φ_c, Φ_v, Φ_a, are denoted as before p_c, p_v, p_a, respectively. The corresponding probability transition matrices are given by $\Pi^{(c)} = \Pi^{(\text{FD})}(p_c, s)$, $\Pi^{(v)} = \Pi^{(\text{FD})}(p_v, s)$, and $\Pi^{(a)} = \Pi^{(\text{FD})}(p_a, s')$.

IV. Density Evolution of Noisy Message Passing Decoding

This section presents the noisy-DE framework for asymptotic analysis of FAIDs on faulty hardware. The DE [9] consists of expressing the Probability Mass Function (PMF) of the messages at successive iterations under the local independence assumption, that is the assumption that the messages coming to a node are independent. The DE analysis enables characterization of the asymptotic behavior of a decoding algorithm under particular decoder noise conditions. It gives the error probability of the considered decoder and is valid on average over all possible LDPC code constructions, when infinite codeword length is considered.

In this paper, not only the considered channel model, but also the noiseless functions and the decoding noise models are symmetric in the sense of [9]. Indeed, Φ_c corresponds to the standard min-sum CNU, while Φ_v is symmetric by construction (see Section II). Furthermore, as the decoder noise is assumed to be only at the end of the function computation, [9, Definition 5] can be applied to show that both the SP-Model and the FD-Model are symmetric. Thus the final error probability of the decoder does not depend on the transmitted codeword and consequently we can assume that the all-zero codeword was transmitted.

In the following, we first give the expression of the PMF of the message values at successive iterations and then explain how they can be used to characterize the asymptotic behavior of the decoders. The presented analysis holds for regular LDPC codes. However, the generalization to irregular codes is straightforward.

A. Noisy Density Evolution Recursion

Let the $|\mathcal{M}|$-tuple $\mathbf{q}^{(\ell)}$ specify the PMF of an outgoing message from a VN at ℓ-th iteration. In other words, the μ-th component $q_\mu^{(\ell)}$ of $\mathbf{q}^{(\ell)}$ is the probability that the outgoing message takes the value $\mu \in \mathcal{M}$. Similarly, let $\mathbf{r}^{(\ell)}$ specify the PMF of an outgoing message from a CN. The PMFs of noisy messages are represented by $\tilde{\mathbf{q}}^{(\ell)}$ and $\tilde{\mathbf{r}}^{(\ell)}$, respectively. In the following, the noisy-DE recursion is expressed with respect to general probability transition matrices $\Pi^{(c)}$,

$\Pi^{(v)}$, $\Pi^{(a)}$. To obtain the DE equations for a specific model, it suffices to replace these general probability transition matrices with the ones corresponding to the considered model.

The density evolution is initialized with the PMF of the channel value

$$q_{-B}^{(0)} = 1-\alpha \quad q_{+B}^{(0)} = \alpha \quad q_{k}^{(0)} = 0 \text{ elsewhere.}$$

Denote $\tilde{\mathbf{q}}_{\boldsymbol{\mu}}^{(\ell-1)}$ the (d_c-1)-tuple associated to $\boldsymbol{\mu}$. More precisely, if the k-th component of $\boldsymbol{\mu}$ is given by μ_k, then the k-th component of $\tilde{\mathbf{q}}_{\boldsymbol{\mu}}^{(\ell-1)}$ is given by $\tilde{q}_{\mu_k}^{(\ell-1)}$. The PMF $\mathbf{r}^{(\ell)}$ of the output of the CNU is obtained from the expression of Φ_c as $\forall \eta \in \mathcal{M}$,

$$r_{\eta}^{(\ell)} = \sum_{\boldsymbol{\mu}:\Phi_c(\boldsymbol{\mu})=\eta} \prod \tilde{\mathbf{q}}_{\boldsymbol{\mu}}^{(\ell-1)} \tag{9}$$

where the vector product operator is performed componentwise on vector elements. The noisy PMF is then obtained directly in vector form as

$$\tilde{\mathbf{r}}^{(\ell)} = \Pi^{(c)}\mathbf{r}^{(\ell)}. \tag{10}$$

Denote $\tilde{\mathbf{r}}_{\boldsymbol{\eta}}^{(\ell)}$ the (d_v-1)-tuple associated to $\boldsymbol{\eta}$. The PMF $\mathbf{q}^{(\ell)}$ of the output of the CNU is obtained from the expression of Φ_v as $\forall \mu \in \mathcal{M}$,

$$q_{\mu}^{(\ell)} = \sum_{\boldsymbol{\eta}:\Phi_v(\boldsymbol{\eta},-B)=\mu} q_{-B}^{(0)} \prod \tilde{\mathbf{r}}_{\boldsymbol{\eta}}^{(\ell)} \quad + \sum_{\boldsymbol{\eta}:\Phi_v(\boldsymbol{\eta},+B)=\mu} q_{+B}^{(0)} \prod \tilde{\mathbf{r}}_{\boldsymbol{\eta}}^{(\ell)} \tag{11}$$

and

$$\tilde{\mathbf{q}}^{(\ell)} = \Pi^{(v)}\mathbf{q}^{(\ell)}. \tag{12}$$

Finally, applying the sequence of 4 equations (9), (10), (11) and (12) implements one recursion of the Noisy-DE for FAIDs over the BSC channel. Note that, to prevent from an important computational complexity increase when the VN and CN degrees increase, (9) and (11) can be computed recursively on the inputs as in [15].

The error probability of the decoder can be obtained from the above recursion and from the PMF of the messages at the end of the APP computation. Denote $\tilde{\mathbf{r}}_{\tilde{\boldsymbol{\eta}}}^{(\ell)}$ the d_v-tuple associated

to $\bar{\boldsymbol{\eta}}$, and denote $\mathbf{q}_{\text{app}}^{(\ell)}$ and $\tilde{\mathbf{q}}_{\text{app}}^{(\ell)}$ the respective noiseless and noisy PMFs of the messages at the output of the APP computation. They can be expressed from (3) as $\forall \gamma \in \bar{\mathcal{M}}$,

$$q_{\text{app},\gamma}^{(\ell)} = \sum_{\bar{\boldsymbol{\eta}}:\Phi_a(\bar{\boldsymbol{\eta}},-B)=\gamma} q_{-B}^{(0)} \prod \tilde{\mathbf{r}}_{\bar{\boldsymbol{\eta}}}^{(\ell)} \quad + \sum_{\bar{\boldsymbol{\eta}}:\Phi_a(\bar{\boldsymbol{\eta}},+B)=\gamma} q_{+B}^{(0)} \prod \tilde{\mathbf{r}}_{\bar{\boldsymbol{\eta}}}^{(\ell)}$$

and

$$\tilde{\mathbf{q}}_{\text{app}}^{(\ell)} = \Pi^{(a)} \mathbf{q}_{\text{app}}^{(\ell)}. \tag{13}$$

Finally, for a given α and letting $\nu = (p_v, p_c, p_a)$, the error probability at each iteration can be computed under the all-zero codeword assumption as

$$P_{e,\nu}^{(\ell)}(\alpha) = \frac{1}{2}\tilde{q}_{\text{app},0}^{(\ell)} + \sum_{k<0} \tilde{q}_{\text{app},k}^{(\ell)}. \tag{14}$$

Proposition 1: The following lower bound holds at every iteration ℓ

1) For the SP model, $P_{e,\nu}^{(\ell)}(\alpha) \geq \frac{1}{2s'} p_a$
2) For the FD model, $P_{e,\nu}^{(\ell)}(\alpha) \geq \frac{1}{2} p_a + \frac{p_a}{4s'}$

Proof: Assume that the VNU and the CNU functions are noiseless and that the iterative decoding process (VNU + CNU part only) has been able to correct all the errors from the channel. Then the errors in the codeword estimate come only from the APP computation part and the error probability is given by $\frac{1}{2}\tilde{q}_{\text{app}}^{(\ell)}(0) = \frac{1}{2s'} p_a$ for the SP model, and by $\frac{1}{2}\tilde{q}_{\text{app}}^{(\ell)}(0) + \sum_{k<0} \tilde{q}_{\text{app}}^{(\ell)}(k) = \frac{1}{2} p_a + \frac{1}{4s'} p_a$ for the FD model.

■

It suffices to study the asymptotic error probability for a given Φ_v, that is the limit of $P_{e,\nu}^{(\ell)}(\alpha)$ when ℓ goes to infinity, to characterize the asymptotic behavior of the noisy decoder. If the limit exists, let $P_{e,\nu}^{(+\infty)}(\alpha) = \lim_{\ell \to +\infty} P_{e,\nu}^{(\ell)}(\alpha)$. For noiseless decoders ($p_v = p_c = p_a = 0$), the maximum channel parameter α such that $P_{e,\nu}^{(+\infty)}(\alpha) = 0$ is called the *threshold* of the decoder [17]. However, this condition cannot be reached in general for noisy-FAIDs. For instance, from Proposition 1, we see that the noise in the APP computation prevents the decoder from achieving a zero-error decoding. As a consequence, there is a need to introduce another threshold definition to characterize the asymptotic behavior of noisy decoders.

B. Analysis of Convergence Behaviors of Noisy Decoders

Varshney in [9] defines the *useful* region as the set of parameters α for which $P_{e,\nu}^{(+\infty)}(\alpha) \leq \alpha$. The useful region indicates what are the faulty hardware and channel noise conditions that a decoder can tolerate to *reduce* the level of noise. However, for a given α, it does not indicate which decoders can correct most of the errors from the channel. On the other hand, in [9], [14], a constant value λ is fixed and the target-BER threshold is defined as the maximum parameter α such that $P_{e,\nu}^{(+\infty)}(\alpha) \leq \lambda$. However, although the target-BER threshold enables to identify the channel parameters for which the decoder can guarantee a given level of error probability, the choice of λ is arbitrary and is not related to any particular intrinsic behavior of the decoder. To finish, the threshold definition we propose in [18] characterizes accurately the behavior of the decoder when $p_v = p_c$ but can give non-consistent results when this condition is not fulfilled.

Here, we introduce another threshold definition which enables to characterize more accurately the asymptotic behavior of a decoder. The threshold definition we propose accounts for the Lipschitz constant of the function $\alpha \mapsto P_{e,\nu}^{(+\infty)}(\alpha)$. The definition of the Lipschitz constant is first restated for the sake of clarity.

Definition 3: 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

$$L(f, I) = \sup_{x \neq y \in I} \frac{|f(x) - f(y)|}{|x - y|} \in \mathbb{R}_+ \cup \{+\infty\} \tag{15}$$

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:

$$L(f, a) = \inf_{\delta > 0} L(f, I_a(\delta)) \in \mathbb{R}_+ \cup \{+\infty\} \tag{16}$$

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 4: For given decoder noise conditions $\nu = (p_v, p_c, p_a)$ and a particular channel parameter α the decoder is said to be *functional* if

(a) The function $x \mapsto P_{e,\nu}^{(+\infty)}(x)$ is defined on $[0, \alpha]$

(b) $P_{e,\nu}^{(+\infty)}$ is Lipschitz continuous on $[0, \alpha]$

(c) $L\left(P_{e,\nu}^{(+\infty)}, x\right)$ is an increasing function of $x \in [0, \alpha]$

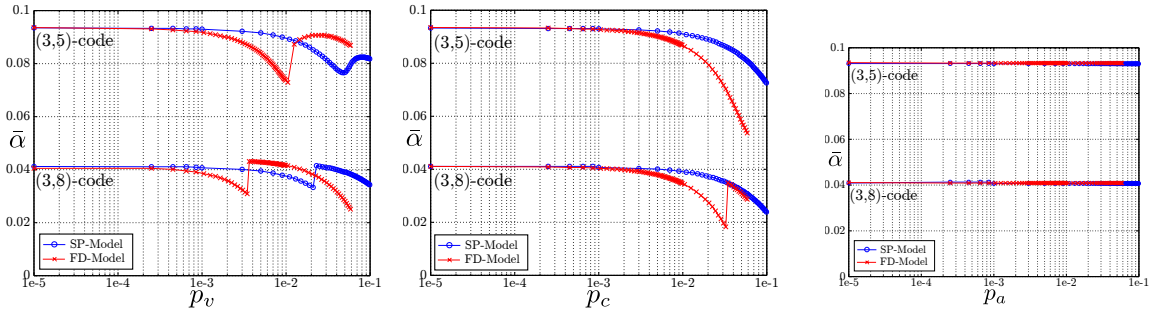
Then the functional threshold $\bar{\alpha}$ is defined as

$$\bar{\alpha} = \sup\{\alpha \mid \text{conditions } (a), (b) \text{ and } (c) \text{ above are satisfied}\} \tag{17}$$

The function $P_{e,\nu}^{(+\infty)}(x)$ is defined provided that there exist a limit of $P_{e,\nu}^{(\ell)}(x)$ when ℓ goes to infinity. Condition (a) is required because $P_{e,\nu}^{(\ell)}(x)$ does not converge for some particular decoders and noise conditions [19].

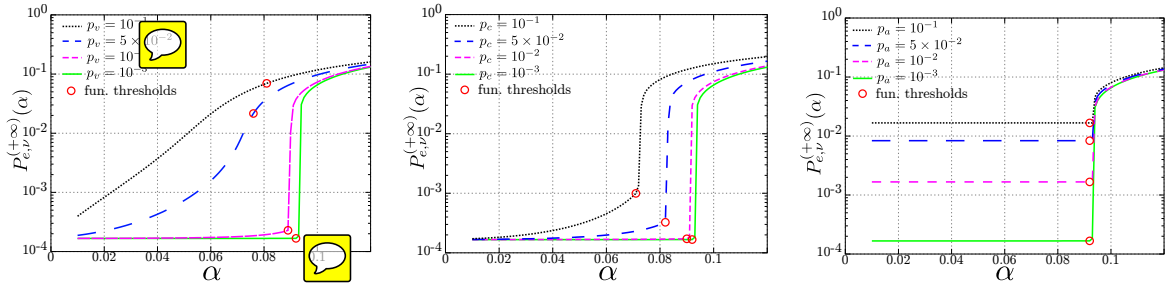
The functional threshold is defined as the transition between two modes. The first mode corresponds to the channel parameters leading to a low level of error probability, *i.e.*, for which the decoder can correct most of the errors from the channel. In the second mode, the channel parameters lead to a high level of error probability, meaning that the decoding operation actually fails. Note that there are two possibilities. If $L\left(P_{e,\nu}^{(+\infty)}, \bar{\alpha}\right) = +\infty$, then $\bar{\alpha}$ is a discontinuity point of $P_{e,\nu}^{(+\infty)}$ and the transition between the two levels is sharp. If $L\left(P_{e,\nu}^{(+\infty)}, \bar{\alpha}\right)$, then $\bar{\alpha}$ is an inflection point of $P_{e,\nu}^{(+\infty)}$ 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 decoder noise parameters is defined as follows.

Definition 5: The set of *admissible decoder noise parameters* is the set of parameters (p_v, p_c, p_a) for which $L\left(P_{e,\nu}^{(+\infty)}, \bar{\alpha}\right) = +\infty$.



(a) (b) (c)

Fig. 2. Functional regions for the offset min-sum, for $C = 1$, (a) w.r.t. p_v, with $p_c = p_a = 10^{-3}$ (SP-Model) and $p_c = p_a = 10^{-4}$ (FD-Model), (b) w.r.t. p_c, with $p_v = p_a = 10^{-3}$ (SP-Model) and $p_v = p_a = 10^{-4}$ (FD-Model), (c) w.r.t. p_a, with $p_v = p_c = 10^{-3}$ (SP-Model) and $p_v = p_c = 10^{-4}$ (FD-Model)

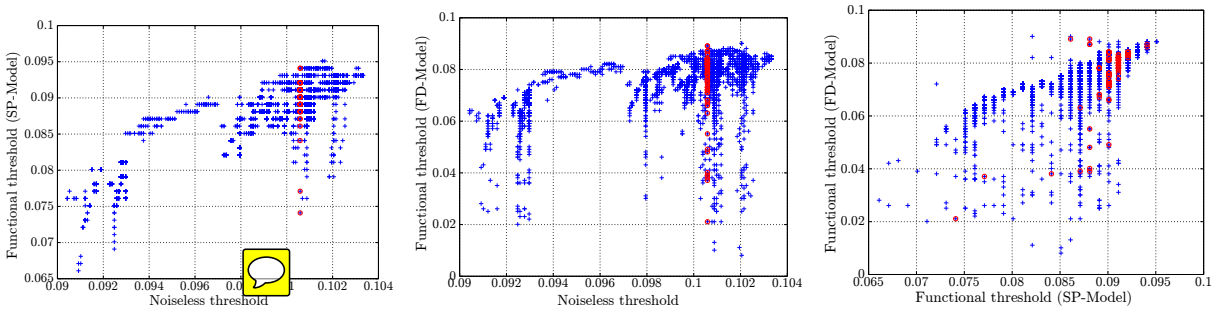


(a) (b) (c)

Fig. 3. Asymptotic error probabilities for $(3, 5)$ codes for the offset Min-Sum, for $C = 1$, for the SP-Model, with (a) $p_c = 10^{-3}$, $p_a = 10^{-3}$, (b) $p_v = 10^{-3}$, $p_a = 10^{-3}$, (c) $p_v = 10^{-3}$, $p_c = 10^{-3}$

C. *Examples of Functional regions*

In this section, $(3, 5)$ and $(3, 8)$ regular codes are considered. Figure 2 (a) shows the functional thresholds with respect to p_v, for the offset Min-Sum decoder. For the SP-Model, we consider $p_c = p_a = 10^{-3}$, and for the FD-Model, $p_c = p_a = 10^{-4}$. As expected, when the rate of the code increases, the functional threshold values decrease. However, when p_v becomes too large, the functional threshold gives non consistent results and thus fails at characterizing the behavior of



(a) (b) (c)

Fig. 4. (a) Noiseless thresholds vs functional thresholds for the SP-Model ($p_v = p_c = p_a = 10^{-2}$) , (b) Noiseless thresholds vs functional thresholds for the FD-Model ($p_v = p_c = p_a = 5 \times 10^{-3}$) (c) Functional thresholds for the SP-Model ($p_v = p_c = p_a = 10^{-2}$) vs functional thresholds for the FD-Model ($p_v = p_c = p_a = 5 \times 10^{-3}$)

the decoder. Figure 3 (a) illustrates this effect. For large values of p_v, there is no discontinuity point anymore and the functional threshold is given by the inflection point of the curve. However, this inflection point does not predict accurately which channel parameters lead to ~~a~~ low level of error probability. That is why the set of admissible noise parameters is introduced in Definition 5. From Fig~~ure~~ 2 (b), we see that the same effect ~~can appear with~~ ~~p_c~~. To ~~finish~~, from Fig~~ures~~ 2 (c) and 3 (c), we observe that the functional threshold does not depend on the value of p_a. This ~~result~~ is expected, because the APP computation does not ~~participate to~~ the iterative decoding process. As a consequence, the noise in the APP only adds noise in the final codeword estimate, but does not make the decoding process fail.

V. Design of FAIDs Robust to Faulty Hardware

In this paper, we want to capitalize on the diversity of VNU functions to design FAIDs which are naturally robust to transient errors introduced by the faulty hardware. Table III shows the total numbers of different FAIDs for $N_s = 5$, $N_s = 7$, and $N_s = 9$ levels.

TABLE III

NUMBER OF FAIDS [16, THEOREM 1]

Total number of VNU functions ($N_s = 5$)	28 314
Total number of VNU functions ($N_s = 7$)	530 803 988
Total number of VNU functions ($N_s = 9$)	230 316 871 499 560

Even by restricting the message alphabet size to $N_s = 7$, the number of possible FAIDs is too large for a systematic analysis. Instead, we ~~account for~~ previous work on FAIDs, and start with a collection of $N_D = 5291$ FAIDs which correspond to column-weight tree codes ~~and have been~~ selected from the ~~analysis on~~ trapping sets presented in [16]. As a result of this selection process, each of the N_D FAIDs have both good noiseless threshold, and good performance in the error floor. We now conduct a noisy-DE analysis on this set by computing, for each of the N_D FAIDs, the value of their functional threshold.

As an example, Fig~~ures~~ 4 (a) and ~~4~~ (b) represent the obtained functional thresholds with respect to the noiseless thresholds. For the SP-Model, the functional thresholds are calculated for $p_v = p_c = p_a = 10^{-2}$, and for the FD-Model, $p_v = p_c = p_a = 5 \times 10^{-3}$. Although all the considered decoders have good noiseless threshold (between 0.09 and 0.104), a ~~large~~ range of behaviors can be observed when the decoder is noisy. Indeed, for the SP-Model, the functional threshold values are between 0.0~~85~~ and 0.095, thus illustrating the existence of both robust and non-robust decoders. In particular, even decoders with approximately the same noiseless threshold value (e.g. around 0.101) can exhibit different robustness ~~abilities~~. This is even more critical for the FD-Model, for which the functional threshold values are between 0.01 and 0.085. These observations illustrate the importance of selecting robust decoders to operate on faulty hardware.

Furthermore, Figure 4 (c) represents the functional thresholds obtained for the FD-Model (for

TABLE IV

FAID RULE $\Phi_v^{(\text{ROBUST})}$ ROBUST TO THE FAULTY HARDWARE (SP-MODEL)

m_1/m_2	$-L_3$	$-L_2$	$-L_1$	0	$+L_1$	$+L_2$	$+L_3$
$-L_3$	$-L_3$	$-L_3$	$-L_3$	$-L_3$	$-L_3$	$-L_2$	0
$-L_2$	$-L_3$	$-L_3$	$-L_3$	$-L_3$	$-L_2$	$-L_2$	L_1
$-L_1$	$-L_3$	$-L_3$	$-L_3$	$-L_2$	$-L_1$	$-L_1$	L_1
0	$-L_3$	$-L_3$	$-L_2$	$-L_1$	$-L_1$	0	L_1
$+L_1$	$-L_3$	$-L_2$	$-L_1$	$-L_1$	0	L_1	L_2
$+L_2$	$-L_2$	$-L_2$	$-L_1$	0	L_1	L_2	L_2
$+L_3$	0	L_1	L_1	L_1	L_2	L_2	L_3

TABLE V

FAID RULE $\Phi_v^{(\text{NON-ROBUST})}$ NOT ROBUST TO FAULTY HARDWARE (SP-MODEL)

m_1/m_2	$-L_3$	$-L_2$	$-L_1$	0	$+L_1$	$+L_2$	$+L_3$
$-L_3$	$-L_3$	$-L_3$	$-L_3$	$-L_3$	$-L_3$	$-L_3$	0
$-L_2$	$-L_3$	$-L_3$	$-L_3$	$-L_3$	$-L_2$	0	L_2
$-L_1$	$-L_3$	$-L_3$	$-L_2$	$-L_2$	$-L_1$	0	L_2
0	$-L_3$	$-L_3$	$-L_2$	$-L_1$	0	L_1	L_3
$+L_1$	$-L_3$	$-L_2$	$-L_1$	0	0	L_1	L_3
$+L_2$	$-L_3$	0	0	L_1	L_1	L_1	L_3
$+L_3$	0	L_2	L_2	L_3	L_3	L_3	L_3

$p_v = p_c = p_a = 5 \times 10^{-3}$) with respect to the functional thresholds obtained for the SP-Model (for $p_v = p_c = p_a = 10^{-2}$). ~~Here~~ also a large variety of behaviors can be observed. Indeed, only a small amount of decoders are robust to both error models, while some of them are robust only to the SP-Model, and some others only to the FD-Model. This suggests that robustness to different error models may require different decoders.

Following these observations, for each model~~,~~ two decoders ~~are extracted~~ from the set of N_D FAIDs. The first one denoted $\Phi_v^{(\text{robust})}$ is the decoder that *minimizes* the discrepancy between noiseless and noisy decoding. The second one $\Phi_v^{(\text{non-robust})}$ is selected to *maximize* the difference between noiseless and noisy decoding. The LUTs of $\Phi_v^{(\text{robust})}$ and $\Phi_v^{(\text{non-robust})}$ are given respectively in Table~~s~~ IV and V for the SP-Model, and in Table~~s~~ VI and VII for the FD-Model.

VI. FINITE LENGTH SIMULATIONS RESULTS

A. Noisy FAIDs

This section gives finite-length simulation results with the noisy FAIDs $\Phi_v^{(\text{robust})}$ and $\Phi_v^{(\text{non-robust})}$ that have been identified by the noisy DE analysis. For the sake of comparison, a third decoder denoted $\Phi_v^{(\text{opt})}$ (Table I) is introduced. It is given in [16] and has been optimized for low error

TABLE VI

FAID RULE $\Phi_v^{(\text{ROBUST})}$ ROBUST TO THE FAULTY HARDWARE (FD-MODEL)

m_1/m_2	$-L_3$	$-L_2$	$-L_1$	0	$+L_1$	$+L_2$	$+L_3$
$-L_3$	$-L_3$	$-L_3$	$-L_3$	$-L_3$	$-L_3$	$-L_1$	0
$-L_2$	$-L_3$	$-L_3$	$-L_3$	$-L_3$	$-L_1$	$-L_1$	L_2
$-L_1$	$-L_3$	$-L_3$	$-L_2$	$-L_2$	$-L_1$	0	L_2
0	$-L_3$	$-L_3$	$-L_2$	$-L_1$	0	0	L_3
$+L_1$	$-L_3$	$-L_1$	$-L_1$	0	0	L_1	L_3
$+L_2$	$-L_1$	$-L_1$	0	0	L_1	L_1	L_3
$+L_3$	0	L_2	L_2	L_3	L_3	L_3	L_3

TABLE VII

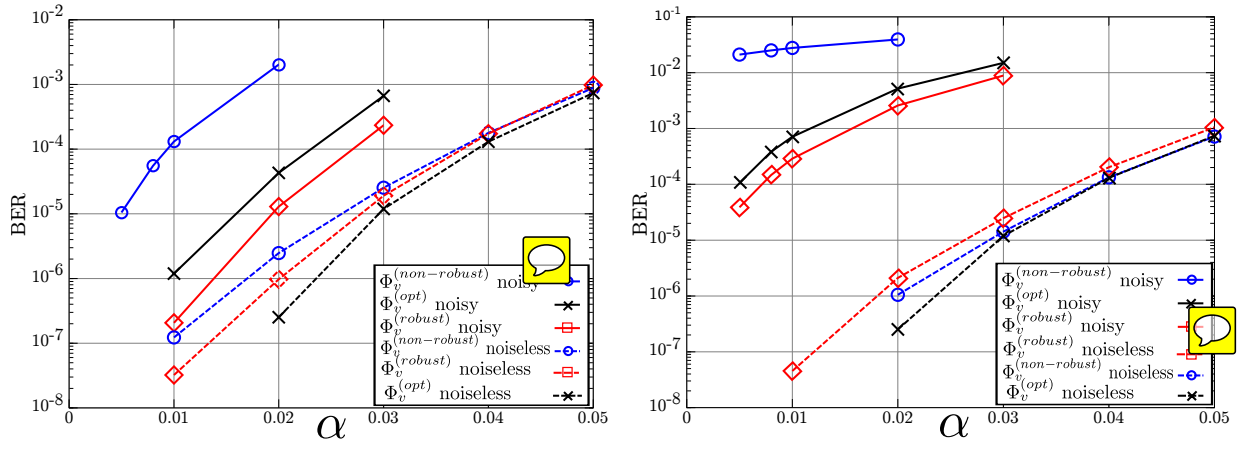
FAID RULE $\Phi_v^{(\text{NON-ROBUST})}$ NOT ROBUST TO FAULTY HARDWARE (FD-MODEL)

m_1/m_2	$-L_3$	$-L_2$	$-L_1$	0	$+L_1$	$+L_2$	$+L_3$
$-L_3$	$-L_3$	$-L_3$	$-L_3$	$-L_3$	$-L_2$	$-L_2$	0
$-L_2$	$-L_3$	$-L_3$	$-L_3$	$-L_3$	$-L_2$	$-L_1$	L_2
$-L_1$	$-L_3$	$-L_3$	$-L_2$	$-L_2$	$-L_1$	0	L_2
0	$-L_3$	$-L_3$	$-L_2$	$-L_1$	0	0	L_3
$+L_1$	$-L_2$	$-L_2$	$-L_1$	0	0	L_1	L_3
$+L_2$	$-L_2$	$-L_1$	0	0	L_1	L_1	L_3
$+L_3$	0	L_2	L_2	L_3	L_3	L_3	L_3

floor ~~when~~ noiseless decoding ~~is considered~~. In the following, the number of iterations is set to 100 and we consider the $(155, 93)$ Tanner code with degrees $(d_v = 3, d_c = 5)$ given in [20]. For the SP-Model, we fix $p_v = p_c = p_a = 0.05$, and for the FD-Model, $p_v = p_c = p_a = 0.02$.

Figure 5 (a) represents the Bit Error Rates (BER) with respect to channel parameter α obtained for the SP-Model. For the noiseless curves, as $\Phi_v^{(\text{opt})}$ has been optimized for low error floor, it performs better, as expected, than the two other FAIDs. But as $\Phi_v^{(\text{robust})}$ and $\Phi_v^{(\text{non-robust})}$ belong to a predetermined set of good FAID decoders, they also have reasonable performance in the noiseless case. Now, from the noisy curves, we see that the results are in compliance with the conclusions from the noisy functional thresholds analysis. Indeed,when the decoder is noisy, $\Phi_v^{(\text{robust})}$ performs better than $\Phi_v^{(\text{opt})}$ while $\Phi_v^{(\text{non-robust})}$ has ~~an important~~ loss in performance compared to the two other decoders.

For the FD-Model, ~~the same conclusions are obtained from Figure 5 (b)~~. In this case, the noisy decoders give more important BER than for the SP-Model despite the lower decoder error levels. The FD-Model is indeed more harmful because not only the amplitudes, but also the signs of the messages can be corrupted by the noise. In particular, the non-robust decoder $\Phi_v^{(\text{non-robust})}$



(a) (b)

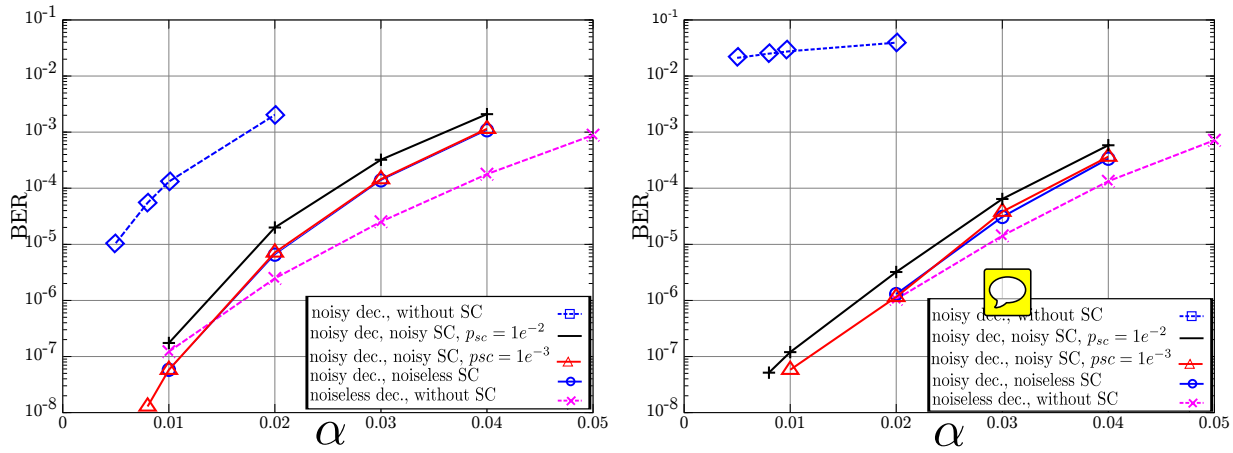
Fig. 5. (155, 93) Tanner Code, $d_v = 3$, $d_c = 5$, 100 iterations, (a) BER for the SP-Model, with $p_v = p_c = p_a = 0.05$, (b) BER for the FD-Model, with $p_v = p_c = p_a = 0.02$

~~can be seen to~~ perform ~~extremely~~ poorly.

To ~~finish~~, we see that the lower bound conditions of Proposition 1 are not ~~fulfilled~~ here. ~~I~~ndeed, in our simulations, we considered an early stopping criterion, ~~that is if the sequence estimated from the APP is a codeword, then the decoding process is stopped before the end of the iterations.~~ ~~T~~he results of Proposition 1 consider ~~a~~veraged error probabilities at a fixed ~~number of~~ iterations, and thus doe ~~n~~ot take into account the early stopping criterion.

B. Self-Correction

Self-Correction (SC) was originally introduced for noiseless Min-Sum decoding in [21]. It was considered for Min-Sum decoding on faulty hardware in [15] and shown to improve the performance of noisy decoders. When SC is introduced, the VN messages obtained at one iteration are stored in memory. Then, at the next iteration, the newly computed VN message values are compared with the previously stored values. If the sign of two compared messages is different, then the message value is reset to 0. When the decoder is noisy, if transient errors



(a) (b)

Fig. 6. $\Phi_v^{(\text{non-robust})}$ decoders, $(155, 93)$ Tanner Code, $d_v = 3$, $d_c = 5$, 100 iterations, (a) BER for the SP-Model, with $p_v = p_c = p_a = 0.05$, (b) BER for the FD-Model, with $p_v = p_c = p_a = 0.02$

cause the sign of some messages to change during one iteration, SC may be able to detect it and to overcome this effect. In this section, we ~~want to~~ analyse the potential gain ~~in error probability at considering~~ SC-FAIDs. For that purpose~~, here,~~ only the non-robust decoders $\Phi_v^{(\text{non-robust})}$ obtained for each model ~~will be~~ considered.

As in the previous section, the $(155, 93)$ Tanner Code is considered, and the number of iterations is set to 100. As SC may also be noisy, the error level in the SC unit is denoted p_{sc}. As the SC information consists only of binary variables that indicate whether a given message may be reset to 0, p_{sc} is simply the symmetric error probability over a binary random variable.

Figure 6 (a) gives the BER for the SP-Model, with $p_v = p_c = p_a = 0.05$. We see that the SC efficiently transforms $\Phi_v^{(\text{non-robust})}$ into a robust decoder. This conclusion holds even when the noise level p_{sc} is relatively high ($p_{sc} = 0.01$ was considered).

Figure 6 (b) gives the BER for the FD-Model, with $p_v = p_c = p_a = 0.02$. The same conclusions as for the SP-Model are obtained, despite the fact that for the FD-Model, $\Phi_v^{(\text{non-robust})}$ performs extremely poorly. In addition, when comparing Figure 6 (a) with Figure 6 (b), the decoder

seems to perform even better under the FD-Model than under the SP-Model. This result could seem surprising, even though $\Phi_v^{(\text{non-robust})}$ is different for each model. However, it can be explained by the combination of two effects. First, knowing that the FD-Model is potentially more harmful, we have considered lower decoder noise levels with the FD-Model (0.02 instead of 0.05). Second, under the FD-Model, a transient error can corrupt both the amplitude and the *sign* of a message, which makes SC particularly efficient to detect errors in this case. On the opposite, under the SP-Model, a transient error on a positive value can only give another positive value, or 0. Then, the newly obtained value 0 may possibly be transformed into a negative value by message computation or by the effect of another transient error. As a consequence, with the SP-Model, at least two operations are needed for the sign to be affected. This makes SC naturally less efficient on this model.

VII. Conclusion

In this paper, we performed an analysis of asymptotic performance of noisy FAIDs using noisy-DE. We introduced the functional threshold that enables to predict the asymptotic behavior of noisy FAIDs. From this asymptotic analysis, we illustrated the ~~large~~ variety of decoders robustness behaviors, and proposed a framework for the design of ~~naturally~~ robust decoders. Finite-length simulation illustrated the gain ~~at~~ consider~~ing~~ robust decoders, and noisy self-correction was shown to improve the performance of ~~the~~ noisy decoders.

In future work~~s~~, more accurate errors models will be considered for the faulty hardware. In particular, models that are not sign-preserving, ~~not symmetric~~, and ~~that~~ intervene at ~~a~~ boolean level inside the function computation ~~may~~ be considered. In addition, as self-correction appears to be a promising solution to overcome the noise introduced by the faulty hardware, there is a need for a theoretical performance analysis of self-corrected, even though the introduced memory makes difficult the use of density evolution. To finish, other applications may be considered, such as the construction of stable memories from LDPC codes.

REFERENCES

[1] J. V. Neumann, *Probabilistic Logics and the Synthesis of Reliable Organisms from Unreliable Components*, ser. Automata Studies. Princeton: Princeton University Press, 1956, pp. 43–98.

[2] P. Gács and A. Gál, "Lower bounds for the complexity of reliable boolean circuits with noisy gates," *IEEE Transactions on Information Theory*, vol. 40, no. 2, pp. 579–583, 1994.

[3] R. Dobrushin and S. Ortyukov, "Upper bound on the redundancy of self-correcting arrangements of unreliable functional elements," *Problemy Peredachi Informatsii*, vol. 13, no. 3, pp. 56–76, 1977.

[4] N. Pippenger, "On networks of noisy gates," in *26th Annual Symposium on Foundations of Computer Science*, 1985, pp. 30–38.

[5] M. Taylor, "Reliable information storage in memories designed from unreliable components," *Bell System Technical Journal*, vol. 47, pp. 2299–2337, 1968.

[6] A. Kuznetsov, "Information storage in a memory assembled from unreliable components," *Problems of Information Transmission*, vol. 9, pp. 254–264, 1973.

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

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

[9] L. Varshney, "Performance of LDPC codes under faulty iterative decoding," *IEEE Transactions on Information Theory*, vol. 57, no. 7, pp. 4427–4444, July 2011.

[10] C. Huang and L. Dolecek, "Analysis of finite-alphabet iterative decoders under processing errors," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2013, pp. 5085–5089.

[11] F. Leduc-Primeau and W. Gross, "Faulty Gallager-B decoding with optimal message repetition," in *50th Annual Allerton Conference on Communication, Control, and Computing*, 2012, pp. 549–556.

[12] C.-H. Huang, Y. Li, and L. Dolecek, "Gallager B LDPC decoder with transient and permanent errors," in *IEEE International Symposium on Information Theory Proceedings*, 2013, pp. 3010–3014.

[13] S. Yazdi, H. Cho, and L. Dolecek, "Gallager B decoder on noisy hardware," *IEEE Transactions on Communications*, vol. 61, no. 5, pp. 1660–1673, 2013.

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

[15] C. K. Ngassa, V. Savin, and D. Declercq, "Min-Sum-based decoders running on noisy hardware," in *IEEE Global Communications Conference*, Dec. 2013, pp. 1–10.

[16] S. Planjery, D. Declercq, L. Danjean, and B. Vasic, "Finite alphabet iterative decoders-part I: decoding beyond belief

propagation on the binary symmetric channel," *IEEE Transactions on Communications*, vol. 61, no. 10, pp. 4033–4045, 2013.

[17] T. J. Richardson and R. 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, Feb. 2001.

[18] E. Dupraz, D. Declercq, B. Vasic, and V. Savin, "Finite alphabet iterative decoders robust to faulty hardware: analysis and selection," in *8th International Symposium on Turbo Codes and Iterative Information Processing (ISTC) (accepted)*, 2014, pp. 1–10.

[19] C. K. Ngassa, V. Savin, and D. Declercq, "Unconventional behavior of the noisy min-sum decoder over the binary symmetric channel," in *Information Theory and Applications Workshop*, 2014, pp. 1–10.

[20] R. Tanner, D. Sridhara, A. Sridharan, T. Fuja, and D. Costello, "LDPC block and convolutional codes based on circulant matrices," *IEEE Trans. on Inf. Th.*, vol. 50, no. 12, pp. 2966–2984, 2004.

[21] V. Savin, "Self-corrected min-sum decoding of ~~ldpc~~ codes," in *IEEE International Symposium on Information Theory*. ~~IEEE~~, 2008, pp. 146–150.