Performance bounds for QC-MDPC codes decoders

Marco Baldi[1], Alessandro Barenghi[2], Franco Chiaraluce[1],
Gerardo Pelosi[2], and Paolo Santini[1]

[1] Università Politecnica delle Marche, Ancona, Italy
[2] Politecnico di Milano, Milano, Italy

{m.baldi, f.chiaraluce, p.santini}@univpm.it
{alessandro.barenghi, gerardo.pelosi}@polimi.it

Abstract. Quasi-Cyclic Moderate-Density Parity-Check (QC-MDPC) codes are receiving increasing attention for their advantages in the context of post-quantum asymmetric cryptography based on codes. However, a fundamentally open question concerns modeling the performance of their decoders in the region of a low decoding failure rate (DFR). We provide two approaches for bounding the performance of these decoders, and study their asymptotic behavior. We first consider the well-known Maximum Likelihood (ML) decoder, which achieves optimal performance and thus provides a lower bound on the performance of any sub-optimal decoder. We provide lower and upper bounds on the performance of ML decoding of QC-MDPC codes and show that the DFR of the ML decoder decays polynomially in the QC-MDPC code length when all other parameters are fixed. Secondly, we analyze some hard to decode error patterns for Bit-Flipping (BF) decoding algorithms, from which we derive some lower bounds on the DFR of BF decoders applied to QC-MDPC codes.

Keywords: QC-MDPC codes · Decoding failure rate · Bit-Flipping decoder · Maximum likelihood decoder · Error floor · Post-quantum cryptography · Code-based cryptography

1 Introduction

Code-based public-key cryptography is deemed as one of the most consolidated and promising areas in post-quantum cryptography. As the most remarkable example, we can mention the Classic McEliece scheme [2], which currently appears as a finalist in the NIST post-quantum standardization process [1, 27]. This scheme essentially consists of a highly optimized version of the original proposal by Robert McEliece [26] and, in particular, employs the same family of error correcting codes (namely, binary Goppa codes). Despite more than 40 years of cryptanalysis, the improvements in known attacks against the original McEliece scheme, which are substantially based on Information Set Decoding (ISD) algorithms, have been very limited (see [5] for a review of such algorithms, and [10, 12] for the state of the art). However, this robustness is somehow paid with very large public keys, a feature that has historically represented Achille's heel of code-based cryptography and, ultimately, has hindered its spreading in modern applications.

To overcome this issue, researchers have thoroughly investigated the possibility of replacing Goppa codes with other error correcting codes, and/or that of adding some geometrical structure to the employed codes, which may enable a more compact code representation. However, the majority of such attempts were unsuccessful, either because of algebraic attacks (such as [15, 38]), structural attacks (such as [3, 13, 25]), or a combination of them [20]. While algebraic code structures proved more difficult to hide and have lead to unbroken instances with moderate advantages in terms of public key size [9, 24], more important reductions in the key size can be achieved by resorting to random-based structured codes like Quasi-Cyclic Moderate-Density Parity-Check (QC-MDPC) codes [4, 30], which derive from the well-known family of Quasi-Cyclic Low-Density Parity-Check (QC-LDPC) codes [6]. However, low-complexity decoding of QC-MDPC codes, as well as QC-LDPC codes, is performed through iterative algorithms derived from Gallager's Bit Flipping (BF) decoder [21] and, differently from bounded distance decoders used for algebraic codes like Goppa codes, these algorithms are characterized by a non-null Decoding Failure Rate (DFR). This implies that an adversary performing a Chosen-Ciphertext Attack (CCA) can gather information about the secret key by exploiting decryption failures [19, 22, 31]. Formally, this translates into the fact that a non-zero DFR may prevent the cryptosystem from achieving Indistinguishability under Adaptively Chosen Ciphertext Attack (IND-CCA2), that is, resistance against active attackers, which is fundamental in many scenarios. To overcome this issue, it is required to guarantee that the decoding algorithm has a provably low DFR, namely, not higher than $2^{-\lambda}$, where λ is the target security level in bits [23]. Such small values of DFR are impossible to assess directly through numerical simulations; thus, finding theoretical models for the DFR of decoders for QC-MDPC codes is of paramount importance.

Related works. For a single-iteration BF decoder, the available analyses establish both the error correction capability [33, 39] and a provable, code-specific upper bound on the DFR [32]. However, for more than one decoder iteration, these models require some assumptions that result in a loose modeling of the decoder performance. Multiple iterations have been conservatively analyzed in [7], for a decoder that however processes the bits in a sequential manner and, consequently, is not efficient in practice. Following a completely different strategy, authors in [36] study the dependence of the DFR on the code length, and propose to extrapolate such a function in the region of low DFR values, based on its trend estimated through numerical simulations for smaller DFR values. Such an extrapolated performance is then used to adjust the code and decoder parameters [17, 18], as well as to design parameter sets for the BIKE cryptosystem [4]. A theoretical justification of this approach is provided in [35], where the authors claim that the logarithm of the DFR is a concave function of the code length, up to the point where the DFR is not larger than $2^{-\lambda}$. Under this assumption, extrapolation with an exponential decay in the code length yields a conservative DFR estimate. This is motivated in [35] through the assumption that, when the DFR is extremely low, the only relevant failure phenomena in a BF decoder are those due to input sequences for which the closest codeword is different from the transmitted one. Notice that, if such an assumption is true, then the BF decoder must approach the optimal (Maximum Likelihood (ML)) decoder in the region of low DFR.

Our contribution. We study the performance of decoders for QC-MDPC codes in the setting with a fixed number of errors. We start by analyzing the DFR of the optimum decoding strategy, corresponding to a complete ML decoder which additionally exploits the knowledge on the number of introduced errors. We show that, for some families of QC-MDPC codes (like those employed in BIKE), this decoder is characterized by a non-zero DFR that decays polynomially in the code length (assuming all the other parameters as fixed). Studying the performance of ML decoding allows us to obtain a lower bound on the DFR of any suboptimal decoder. In particular, through our analysis, we are able to formally and rigorously prove the existence of the *error floor* region, for the considered codes, as a function of the code length. The error floor is a well-known phenomenon when the codes are used in communication systems, for example affected by thermal noise, but its dependence on the code length has been rarely investigated in previous literature [29, 35].

More precisely, we consider BF decoders for QC-MDPC codes and show how to identify some error patterns that, with high probability, cannot be corrected. By doing this, we are able to compute a lower bound on the DFR of such decoders. With our results, we are able to provide evidence in contrast with the claims in [35], in particular showing that: i) a BF decoder is extremely far from being optimal, and ii) the most likely failure events are not those due to near-codewords. It must be noted that, in an independent and very recent work [42], Vasseur has come to similar conclusions with a thorough analysis of near-codewords and their impact on the DFR. We remark that our results do not directly imply that the parameters proposed in [4, 18, 35] do not achieve the claimed DFR. However, they suggest that finding exact models for the performance of a BF decoder still requires further investigations, especially in the regime, here of interest, of extremely low DFR.

The paper is organized as follows. In Section 2 we establish the notation used throughout the paper, and we provide basic concepts about coding theory and QC-MDPC codes. In Section 3 we analyze the ML decoder, and we employ the obtained results to prove the existence of the floor for specific families of QC-MDPC codes. In Section 4 we take into account BF decoding, and we describe how to pick hard to decode errors and how to use such vectors to find a lower bound on the DFR. Finally, in Section 5 we draw some concluding remarks. *With respect to the version published in the Proceedings of International Workshop on Code-Based Cryptography (CBCrypto 2021), this paper contains an additional appendix, namely, Appendix 5*

2 Notation and Background

We use bold uppercase letters to denote matrices, and bold lowercase letters to denote vectors. Given a matrix $\mathbf{A}$, we use $\mathbf{A}_{i,:}$ (resp, $\mathbf{A}_{:,i}$) to denote its i-th row (resp. column), while $a_{i,j}$ refers to its entry in the i-th row and j-th column. For a vector $\mathbf{a}$, we use a_i to refer to its i-th component. The null vector of length n is indicated as $\mathbf{0}_n$. The Hamming weight of vector $\mathbf{a}$ is denoted as $\mathrm{wt}(\mathbf{a})$, while $\mathrm{Supp}\left(\mathbf{a}\right)$ refers to its support, that is, the set of indexes pointing at non-null entries. Let $\mathbb{F}_2$ denote the binary finite field. For two vectors $\mathbf{a}$ and $\mathbf{b}$ with equal length, defined over $\mathbb{F}_2$, we denote as $\langle \mathbf{a}\ ;\ \mathbf{b}\rangle$ their integer inner product, that is, their inner product after lifting their entries from $\mathbb{F}_2$ to the ring of integers $\mathbb{Z}$.

For a set A, the expression $a \xleftarrow{\$} A$ means that a is uniformly picked among the elements of A; the cardinality of the set is denoted as $|A|$. We use $B_{n,w} \subset \mathbb{F}_2^n$ to denote the Hamming sphere with radius w, that is, the set of length-n vectors with Hamming weight w.

2.1 Error correcting codes

In the following we focus on linear block codes over $\mathbb{F}_2$.

Definition 1. *A linear code $\mathscr{C}$ of length n, dimension k and redundancy $r = n - k$ over $\mathbb{F}_2$ is a k-dimensional linear subspace of $\mathbb{F}_2^n$. We say that $\mathbf{G} \in \mathbb{F}_2^{k\times n}$ is a* generator matrix *for $\mathscr{C}$ if it is a basis of $\mathscr{C}$; a matrix $\mathbf{H} \in \mathbb{F}_2^{r\times n}$ is said to be a* parity-check matrix *for $\mathscr{C}$ if it is a basis of its null space.*

A crucial property of a linear code is that the sum of any number of codewords yields another codeword. Codes are normally endowed with a distance metric, that is, a function able to measure the distance between pairs of codewords; in this paper we only consider the Hamming metric, defined next.

Definition 2. *The* Hamming distance *in the vector space $\mathbb{F}_2^n$ is defined as the function* $\mathrm{dist} : \mathbb{F}_2^n \times \mathbb{F}_2^n \to \mathbb{N}$ *such that*

$$\mathrm{dist}(\mathbf{a}, \mathbf{b}) = |\mathrm{Supp}\left(\mathbf{a} + \mathbf{b}\right)| = \mathrm{wt}(\mathbf{a} + \mathbf{b}).$$

We finally recall the concepts of *weight distribution* and *minimum distance.*

Definition 3. *For a linear code $\mathscr{C} \subseteq \mathbb{F}_2^n$ and $w \in [0; n]$, we denote with A_w the number of codewords whose weight is w. Then, the* weight distribution *of $\mathscr{C}$ corresponds to the collection of all values A_w. The* minimum distance *of $\mathscr{C}$ is defined as the minimum $w > 0$ such that $A_w > 0$ or, equivalently, as*

$$d = \min\left\{\mathrm{dist}(\mathbf{c}, \mathbf{c}') \mid \mathbf{c}, \mathbf{c}' \in \mathscr{C},\ \ \mathbf{c} \neq \mathbf{c}'\right\} = \min\{\mathrm{wt}(\mathbf{c}) \mid \mathbf{c} \in \mathscr{C} \setminus \mathbf{0}_n\}.$$

Arguably, the most important application of linear codes is that of error correction over noisy channels; this is accomplished through decoding algorithms, i.e., techniques that, within certain limits, can identify the channel action on a received sequence and, consequently, reconstruct the transmitted codeword. In this work, we focus on the use of error correcting codes in the context of the McEliece cryptosystem. In such a setting, the message to be transmitted is first encoded as a codeword and then an *error vector* $\mathbf{e}$ of fixed weight t is added to it. To this end, we introduce the *McEliece channel*, whose action is described as

$$\mathbf{c} \mapsto \mathbf{x} = \mathbf{c} + \mathbf{e}, \;\; \mathbf{c} \in \mathbb{F}_2^n, \;\; \mathbf{e} \xleftarrow{\$} B_{n,t},$$

where $\mathbf{c}$ is the input sequence and $\mathbf{e}$ is the error introduced by the channel.

To provide a rigorous classification of decoders, we consider the following formal definition, which has been made specific to the McEliece channel.

Definition 4. *Let $\mathscr{C} \subseteq \mathbb{F}_2^n$ be a linear code of length n. We say that an algorithm* $\mathsf{Dec} : \mathbb{F}_2^n \to \mathbb{F}_2^n$ *has DFR ϵ for $\mathscr{C}$, in the McEliece channel with parameter t, if*

$$\Pr\left[\; \mathsf{Dec}(\mathbf{c}+\mathbf{e}) \neq \mathbf{c} \;\middle|\; \mathbf{c} \xleftarrow{\$} \mathscr{C}, \;\; \mathbf{e} \xleftarrow{\$} B_{n,t} \right] = \epsilon.$$

2.2 QC-MDPC codes

Let us recall the definition of MDPC codes, which were first introduced in [28] for the context of communications but, later on, received interest for the use in public-key cryptosystems [30].

Definition 5. *Let $\mathbf{H} \in \mathbb{F}_2^{r\times n}$ such that all of its rows have weight $w = O(\sqrt{n})$; then, we say that the code having $\mathbf{H}$ as parity-check matrix is an* MDPC code*.*

Namely, MDPC codes are analogous to LDPC codes, with the only difference that their parity-check matrices are denser than those of typical LDPC codes.

In particular, when used in cryptography, these codes are usually endowed with the QC structure, that is, the matrix $\mathbf{H}$ is formed by circulant blocks of size p. Note that, for a circulant matrix, all rows and columns have the same weight; thus, with some abuse of notation, we will use the term "weight of a circulant matrix" to refer to the weight of any of its rows/columns. From now on, we will focus on a particular class of QC-MDPC codes, which we formally define as follows.

Definition 6. *Let $\mathbf{H} = [\mathbf{H}_0, \cdots, \mathbf{H}_{n_0-1}]$, with $\mathbf{H}_i$ being a circulant matrix of size p and weight $v = O(\sqrt{p/n_0})$. Then, we say that the code $\mathscr{C}$ admitting $\mathbf{H}$ as parity-check matrix is an* n_0-QC-MDPC code*. Furthermore, we denote with $\mathcal{QC}$-$\mathcal{MDPC}(n_0, p, v)$ the collection of all such codes.*

Note that an n_0-QC-MDPC code has length $n = n_0p$, dimension $k = (n_0-1)p$ and redundancy $r = p$. A parity-check matrix as in Definition 6 has all columns with weight v, while all rows have weight $w = n_0v = O(\sqrt{n})$. In a cryptosystem, a user randomly and uniformly picks a code from $\mathcal{QC}$-$\mathcal{MDPC}(n_0, p, v)$, and uses its parity-check matrix as the secret key.

3 Maximum-Likelihood Decoding

In this section we analyze the optimal decoding strategy of QC-MDPC codes exploiting ML, and characterize its performance over the McEliece channel. Such a technique works by first testing the distance between each codeword and the received sequence, and then by outputting the codeword that minimizes such a distance. When there is more than one codeword at the same minimum distance from the received sequence, then the ML decoder can apply one of the following two strategies:

- *Complete* ML decoding: the decoder randomly outputs one of the codewords at minimum distance from the received sequence.
- *Incomplete* ML decoding: the decoder halts and reports a decoding failure.

In this paper we consider a complete ML decoder. We observe that the results we obtain can easily be adapted to the case of an incomplete ML decoder and, in general, no big difference exists between the two behaviors from a practical standpoint. However, since the complete ML decoder always returns a codeword, it is clear that its DFR is lower than that of the incomplete counterpart. Indeed, the two decoders behave differently only when there is more than one codeword at the same distance from the received sequence. In such a situation, the incomplete decoder will not try to decode (hence, according to Definition 4, it will fail), while with some non null probability the complete version will return the correct codeword.

Taking into account the fact that, in our case, there are exactly t errors affecting each transmitted codeword, we can modify the standard definition of complete ML decoding as follows.

Definition 7. *Let $\mathscr{C}$ be a linear code over $\mathbb{F}_2$ with length n. The complete* ML-decoder *is the algorithm $\mathsf{ML}:\mathbb{F}_2^n\rightarrow\mathscr{C}$ that, on input $\mathbf{x}\in\mathbb{F}_2^n$, returns $\mathbf{c}'\xleftarrow{\$}\mathscr{C}^{(\mathbf{x})}$, where*

$$\mathscr{C}^{(\mathbf{x})}=\left\{\mathbf{c}\in\mathscr{C}\ \text{ s.t. }\ \mathrm{dist}(\mathbf{x},\mathbf{c})=t\right\},$$

that is, $\mathscr{C}^{(\mathbf{x})}$ is the set of all the codewords of $\mathscr{C}$ which are exactly t away from $\mathbf{x}$ under Hamming distance.

Note that, when $\mathscr{C}^{(\mathbf{x})}$ contains only one codeword, obviously that codeword is the decoder output (so, no randomness is involved). We point out that the decoder we have defined above corresponds to the best decoder (in terms of DFR) one can dispose of, in the McEliece channel. Indeed, the decoder i) exploits knowledge on the number of errors, and ii) always returns a codeword. Because of these reasons, the study of its performances is meaningful since it allows us to derive the minimum DFR that can be reached. Complete ML decoding can also be performed when the decoder input is the syndrome of the received sequence; formally, we define such a procedure as follows.

Definition 8. *Let $\mathscr{C}$ be a linear code over $\mathbb{F}_2$ with length n and parity-check matrix $\mathbf{H}$. We define the* ML syndrome decoder *as the algorithm* $\mathsf{MLS}\colon \mathbb{F}_2^n \to \mathbb{F}_2^n$ *that, on input $\mathbf{x} \in \mathbb{F}_2^n$, returns $\mathbf{x} + \mathbf{e}'$, where $\mathbf{e}' \xleftarrow{\$} \mathcal{S}_{\mathbf{H}}^{(\mathbf{x})}$ and*

$$\mathcal{S}_{\mathbf{H}}^{(\mathbf{x})} = \left\{\mathbf{e} \in B_{n,t} \text{ s.t. } \mathbf{e}\mathbf{H}^\top = \mathbf{x}\mathbf{H}^\top\right\}.$$

Notice that the ML and MLS decoders are, in principle, different from each other. Indeed, the ML decoder always returns a codeword, while the MLS decoder may return a vector that does not belong to the code. Yet, in the following theorem we prove that the DFR of these algorithms coincide, and furthermore we provide explicit bounds for such a failure probability.

Theorem 1. *Let $\mathscr{C} \subseteq \mathbb{F}_2^n$ be a linear code of length n, dimension k and minimum distance d, and consider the transmission over the McEliece channel with parameter t. Then, the ML and MLS decoding algorithms have the same DFR, denoted as ϵ_{ML}, which is equal to*

$$\epsilon_{\mathsf{ML}} = 1 - \frac{1}{\binom{n}{t}} \sum_{\mathbf{e} \in B_{n,t}} \frac{1}{\left|\mathscr{C}^{(\mathbf{e})}\right|}.$$

Furthermore, it holds that $\epsilon_{\mathsf{ML}}^{(L)} \leq \epsilon_{\mathsf{ML}} \leq \epsilon_{\mathsf{ML}}^{(U)}$, where

$$\epsilon_{\mathsf{ML}}^{(U)} = \frac{1}{2\binom{n}{t}} \sum_{\substack{w \in [d;2t] \\ w \text{ even}}} A_w \binom{w}{w/2}\binom{n-w}{t-w/2},$$

$$\epsilon_{\mathsf{ML}}^{(L)} = \begin{cases} 0 & \textit{if } A_w = 0 \textit{ for all even } w \in [d;2t], \\ \max_{\substack{w \in [d;2t] \\ w \text{ even} \\ A_w > 0}} \left\{ \frac{\binom{w}{w/2}\binom{n-w}{t-w/2}}{2\binom{n}{t}} \right\} & \textit{otherwise.} \end{cases}$$

Proof. See Appendix A.

It is clear that the computational complexity of both ML and MLS decoders is intractable unless the code or the channel have trivial parameters (i.e., very low values of k and/or t). Indeed, a straightforward implementation of the ML decoder runs in time $O(2^{Rn})$ (since all codewords must be tested), being $R = k/n$ the code rate, while the MLS decoder takes time $O(n^t)$ (since it tests all vectors in $B_{n,t}$, of size $\binom{n}{t} = O(n^t)$). Furthermore, we recall that solving the decoding problem for a generic random linear code was proven to be NP-complete [11], as well as finding its minimum distance [41]. It is thus rather unlikely that efficient implementations of ML decoders are found. For this reason, one normally relies on sub-optimal decoding strategies. Hence, any such practical decoder is going to have a DFR higher than that of the ML decoder.

3.1 ML decoders for QC-MDPC codes

When QC-MDPC codes are employed in public-key cryptosystems [28, 4, 8], we have that both the secret and the public keys are representation of the same code $\mathscr{C}$, drawn at random from $\mathcal{QC}$-$\mathcal{MDPC}(n_0, p, v)$. In particular, the secret key corresponds to a sparse parity-check matrix, while the public key is either a dense generator or a dense parity-check matrix. Furthermore, we have that n_0 is normally chosen as a small integer, namely, $n_0 \in \{2, 3, 4\}$. Because of the QC structure, we can derive some common properties for these codes, as stated in the following proposition.

Proposition 1. *Let $\mathscr{C}$ be picked at random from $\mathcal{QC}$-$\mathcal{MDPC}(n_0, p, v)$. Then, the following properties hold:*

i) the minimum distance of $\mathscr{C}$ is not greater than $2v$;
ii) we have $A_{2v} \geq p\binom{n_0}{2}$.

Proof. See Appendix B.

When employed in a public-key cryptosystem, the parameters of a QC-MDPC code must satisfy some constrains in order to guarantee the desired security level λ. As it is well known, the best attacks against these schemes exploit Information Set Decoding (ISD) algorithms, which are techniques originally conceived for decoding arbitrary codes, when no efficient decoding algorithm is available. Given a code with length n and dimension k, an ISD algorithm can be used to decode an error vector of weight ω with a computational complexity that is well approximated [40] as

$$C_{\mathsf{ISD}}(n, k, \omega) \approx 2^{-\omega \log_2(1-k/n)}.$$

Note that the above complexity also corresponds to that of finding a specific codeword of weight ω in a code with the same parameters. In a public-key cryptosystem employing QC-MDPC codes, two main applications of ISD exist:

- decoding attacks, that aim at recovering the plaintext from an intercepted ciphertext, which can either be in the form of a syndrome or an error corrupted codeword. In both cases, the corresponding error has weight t, thus an adversary faces a complexity equal to $\frac{C_{\mathsf{ISD}}(n,k,t)}{\sqrt{p}}$, where the polynomial speed-up comes from quasi-cyclicity [34];
- key recovery attacks, that aim at finding low weight codewords in either the public code or its dual. The knowledge about these codewords will indeed reveal the structure of the sparse parity-check matrix used as the private key. In particular, it can be shown that searching for low weight codewords in the dual code corresponds to the optimal attack strategy [8, Section 2.3.1]. We can then assess the complexity of this kind of attacks as $\frac{C_{\mathsf{ISD}}(n_0p,p,n_0v)}{p}$.

To reach a security level of λ bits, we must guarantee that all successful attacks run in a time not lower than 2^λ. Hence, taking these considerations into account, we get that v and t must satisfy the following relationships

$$\begin{cases} v \approx \frac{\lambda+\log_2(p)}{n_0 \log_2\left(\frac{n_0}{n_0-1}\right)}, \\ t \approx \frac{\lambda+\frac{1}{2}\log_2(p)}{\log_2(n_0)}, \end{cases} \tag{1}$$

from which, with simple algebra, we get

$$t \approx vn_0 \left(1 - \frac{\log_2(n_0-1)}{\log_2(n_0)}\right) - \frac{\log_2(p)}{2\log_2(n_0)}. \tag{2}$$

QC-MDPC codes with $n_0 = 2$. To consider a case of practical interest, we focus on $n_0 = 2$; actually, this corresponds to the QC-MDPC codes that are considered in the BIKE cryptosystem [4] and other relevant works [36, 35, 18, 17]. Assuming $p \approx n_0 v^2$ (recall Definition 6), from (2) we have that

$$t \approx 2v - 0.5 - \log_2(v).$$

For security levels of practical interest, we always have $v < t$: since the resulting QC-MDPC$(2, p, v)$ code always contains codewords of even weight $\leq 2t$ (as stated in Proposition 1), applying Theorem 1 we get that the ML decoder has a provably non-zero DFR. Indeed, we can plug $w = 2v$ into the expression of $\epsilon_{\mathsf{ML}}^{(L)}$, and correspondingly obtain a lower bound on the DFR of the ML decoder as

$$\epsilon_{\mathsf{ML}}^{(L)} = \frac{\binom{2v}{v}\binom{2p-2v}{t-v}}{2\binom{2p}{t}}.$$

Notice that, for growing p and fixed v and t, we get $\epsilon_{\mathsf{ML}}^{(L)} = O\left(p^{-v}\right)$, which is polynomial in the circulant size p. To highlight such result, we encapsulate it in the following proposition.

Proposition 2. *Consider $\mathscr{C} \in \mathcal{QC}\text{-}\mathcal{MDPC}(2, p, v)$ used over a McEliece channel with parameter $t = 2v - 0.5 - \log_2(v)$. Then, ML decoding of $\mathscr{C}$ fails with a probability that decays asymptotically as $O(p^{-v})$.*

This result is foundational, since it proves that, when the parameters v and t are fixed, the DFR of the ML decoder decays polynomially with the circulant size (which is linear in the code length). This is the typical *floor* behavior: the DFR (seen as a function of the code length) starts with an exponential decay but, at some point, the slope changes and the DFR decay becomes only polynomial. To have a further insight on the lower bound of the ML decoder, and to especially highlight how it depends on the code parameters p and v, with simple approximations we elaborate the previous expression and get

$$\epsilon_{\mathsf{ML}}^{(L)} \approx 2^{1.5573v - v\log_2\left(\frac{p}{v}\right) - 0.5\log_2(v) - 1.3257}. \tag{3}$$

To see how such an estimate has been derived, see Appendix C. To have a graphical view of how $\epsilon_{\mathsf{ML}}^{(L)}$ evolves with the circulant size p, and also to have an evidence of the quality of the approximation in (3), we provide some numerical examples in Fig. 1.

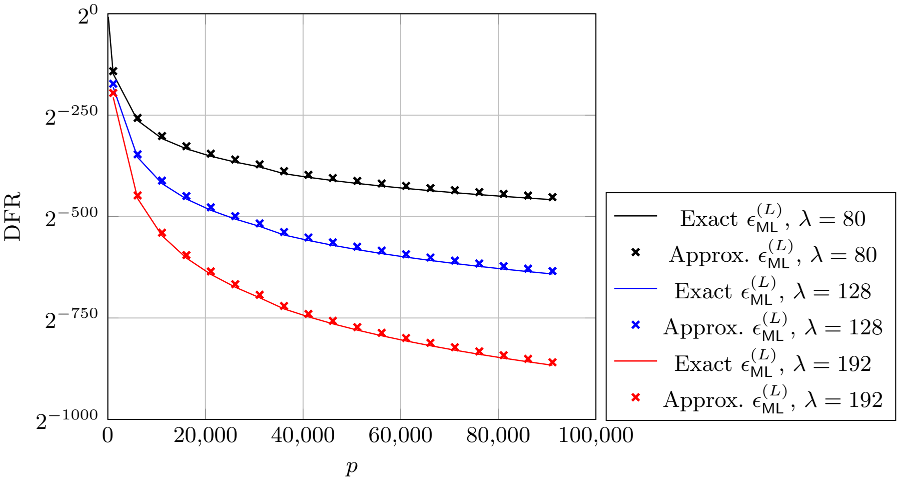


Fig. 1: Lower bound on the DFR of the ML decoder, for QC-MDPC codes with $n_0 = 2$ and parameters achieving different security levels. For each value of p, we have computed v and t through (1). The exact value of $\epsilon_{\mathsf{ML}}^{(L)}$ is computed as in Theorem 1, considering $w = 2v$, while the approximated one has been obtained through (3).

QC-MDPC codes with $n_0 \geq 4$. Interestingly, for $n_0 \geq 4$, (2) implies $t < v$. Recall that, due to sparsity, we expect that the minimum distance of a large majority of QC-MDPC codes is exactly $2v$. For all such codes, the upper bound $\epsilon_{\mathsf{ML}}^{(U)}$ expressed in Theorem 1 is null, and hence our analysis does not highlight the existence of the floor region.

4 Lower bounds for BF decoders

As mentioned before, the ML decoder is interesting from a theoretical perspective, since it can be used to derive a safe lower bound on the DFR of any decoder employed in practice. Yet, practical decoders in cryptosystems usually rely on completely different decoding strategies, which originate from the BF decoder first presented in [21]. In this section we propose a numerically-aided approach to compute a lower bound on the DFR of BF decoders. Based on Propositions 3 and 4, we will be able to find error vectors with a special structure by only looking at the code parity-check matrix, without needing any simulation. Then, starting from these error vectors, a lower bound on the DFR of BF decoding can be computed by exploiting some numerical simulations, as will be described in Proposition 5. For this reason, the lower bound we propose, which partially relies on simulations, is defined as numerically-aided.

A BF decoder performs the decoding procedure starting from an estimate of the value of $\mathbf{e}$, initially set to $\mathbf{0}_n$, and changes this estimate, flipping its bit values (hence the name) on the basis of a set of values computed starting from the syndrome, known as *counters*, which are defined as follows.

Definition 9. *Let* $\mathbf{H} \in \mathbb{F}_2^{r \times n}$ *and* $\mathbf{s} = \mathbf{x}\mathbf{H}^\top$*, for* $\mathbf{x} \in \mathbb{F}_2^n$*. We define the* i*-th counter* σ_i *as the number of set entries in* $\mathbf{s}$ *that are indexed by* $\mathrm{Supp}\left(\mathbf{H}_{:,i}\right)$ *or, equivalently, as the number of unsatisfied parity-check equations in which the* i*-th bit participates.*

It is immediately seen that almost all QC-MDPC decoders proposed in the literature (like those in [39, 32, 36, 18, 35, 17, 7]) include a stage in which error estimate bit flipping decisions are taken on the basis of counters. So, to encompass all such algorithms, we will generically speak of BF decoders.

Let $\mathbf{x} = \mathbf{c}+\mathbf{e}$, with $\mathbf{c}$ being a codeword and $\mathbf{e}$ being the error vector introduced by the channel. Any BF decoder follows a common procedure, which can be summarized as follows:

1. on input $\mathbf{x} \in \mathbb{F}_2^n$, compute the syndrome $\mathbf{s} = \mathbf{x}\mathbf{H}^\top$ and initialize the error estimate $\mathbf{e}' = \mathbf{0}_n$;
2. compute the counters σ_i, for $i = \{0, \dots, n-1\}$;
3. assume positions of $\mathbf{e}$ corresponding to high valued counters to be incorrectly estimated, and flip the corresponding entries in $\mathbf{e}$;
4. update the syndrome as $\mathbf{s} + \mathbf{e}'\mathbf{H}^\top$. If the new syndrome is null, complete the procedure outputting $\mathbf{x} + \mathbf{e}'$. If the new syndrome is not null and the maximum number of iterations has not been reached, restart from step 2, otherwise report the occurrence of a failure.

In particular, step 3 is implemented through a threshold criterion: positions associated to counters with values greater than or equal to some threshold $b \leq v$ are considered to be incorrectly estimated. When the decision on a bit is correct (i.e., when the current value of e'_i is different from e_i) we speak of *correct flip*, otherwise (i.e., when the current value of e'_i is equal to e_i) we speak of *wrong flip*. Notice that, in each iteration, we have that $\mathbf{s}$ corresponds to the syndrome of the vector $\mathbf{e}+\mathbf{e}'$. The value of b may be chosen in different ways (for instance, as a function of the iteration number and the syndrome weight), and is not expected to become lower than $v/2$. The reason for this claim is explained next. Indeed, any BF decoder treats as error affected the bits for which the number of unsatisfied involved parity-check equations exceeds that of the satisfied ones. Choosing $b < v/2$ implies that we contradict this criterion, hence we expect that the decoder ends up in performing a number of wrong flips which is larger than that of correct flips.

The counters values are related to the structure of $\mathbf{H}$, as well as to the support of the error vector; the exact relation is described in the next lemma.

Lemma 1. *Let* $\mathbf{H} \in \mathbb{F}_2^{r \times n}$ *and* $\mathbf{s} = \mathbf{e}\mathbf{H}^\top$ *for a vector* $\mathbf{e} \in \mathbb{F}_2^n$*. Let*

$$\gamma_{i,j} = \begin{cases} \left|\mathrm{Supp}\left(\mathbf{H}_{:,i}\right) \cap \mathrm{Supp}\left(\mathbf{H}_{:,j}\right)\right| & \text{if } i \neq j, \\ 0 & \text{if } i = j. \end{cases}$$

Let

$$\zeta_i^{(1)}(\mathbf{H},\mathbf{e}) = \sum_{j\in\mathrm{Supp}(\mathbf{e})\setminus\{i\}} \gamma_{i,j} - 2\sum_{\ell\in\mathrm{Supp}(\mathbf{H}_{:,i})} \left\lfloor \frac{\langle \mathbf{H}_{\ell,:}^{(i)}\ ;\ \mathbf{e}^{(i)}\rangle}{2} \right\rfloor,$$

$$\zeta_i^{(0)}(\mathbf{H},\mathbf{e}) = \sum_{j\in\mathrm{Supp}(\mathbf{e})} \gamma_{i,j} - 2\sum_{\ell\in\mathrm{Supp}(\mathbf{H}_{:,i})} \left\lfloor \frac{\langle \mathbf{H}_{\ell,:}\ ;\ \mathbf{e}\rangle}{2} \right\rfloor,$$

where $\mathbf{H}_{\ell,:}^{(i)}$ and $\mathbf{e}^{(i)}$ are the vectors obtained via puncturation of the i-th position. Then, for the i-th counter σ_i, the following relation holds

$$\sigma_i = \begin{cases} \mathrm{wt}(\mathbf{H}_{:,i}) - \zeta_i^{(1)}(\mathbf{H},\mathbf{e}) & \text{if } e_i = 1, \\ \zeta_i^{(0)}(\mathbf{H},\mathbf{e}) & \text{if } e_i = 0. \end{cases}$$

Proof. See Appendix D.

4.1 Hard to decode errors for QC-MDPC

In this section we rely on Lemma 1 to construct error patterns that, with high probability, cannot be corrected by a BF decoder. Namely, we consider the subset of $B_{n,t}$ formed by the vectors that have a large number of overlapping ones with a column of the parity-check matrix. We show that for such vectors decoding fails with a probability that is rather high, and use numerical simulations to find a lower bound for the DFR of the BF decoder.

Let $\mathscr{C}$ be a QC-MDPC(n_0, p, v) code, with parity-check matrix $\mathbf{H} \in \mathbb{F}_2^{p\times n_0 p}$ and $\mathbf{e} \in B_{n,t}$. As we have already said, a BF decoder takes decisions (i.e., decides which bits are correct and which are error affected) according to the counters values. We expect that high counters are associated to error positions, and low counters are associated to error free positions: if the counters behave in the opposite way (we speak of *bad counters*), then the decoder may make wrong choices. In particular, the decoder may potentially get stuck in a bad configuration when the number of bad counters is rather large. To better explain what we expect to happen in such a situation, let us start with some preliminary considerations.

- Let $\delta(\mathbf{e}) = \max\{\sigma_i \mid i \in \mathrm{Supp}\,(\mathbf{e})\}$. Clearly, a single iteration of a BF decoder with threshold set as $b > \delta(\mathbf{e})$ will never flip any of the set bits in $\mathbf{e}$.
- We expect the same phenomenon happens, with very high probability, even when considering multiple iterations, all employing thresholds larger than $\delta(\mathbf{e})$. Indeed, a flip among the set bits of $\mathbf{e}$ can happen only if the decoder, at some point, makes wrong flips and these flips trigger, in the subsequent iterations, correct flips among the positions indexed by $\mathrm{Supp}\,(\mathbf{e})$. Yet, this phenomenon should happen with extremely low probability. Indeed, when the decoder makes a wrong flip, it moves into a state characterized by more errors: it is very implausible that this somehow helps the decoding process.

- When $\delta(\mathbf{e})$ is particularly low (say, lower than $\lceil v/2 \rceil$), then it is reasonable that decoding fails, regardless of the employed thresholds. Indeed, to flip the set bits in $\mathbf{e}$, a threshold lower than $\lceil v/2 \rceil$ is required. However, with this choice, it becomes very likely that the number of wrong flips exceeds that of correct flips. Hence, the decoder simply increases the overall number of wrongly estimated bits.
- Analogous reasoning can be applied to the case in which an error vector is such that there is a large number of error free positions with high counters values. Indeed, in such a case, the decoder may wrongly flip some of the corresponding bits, and hence will end up in introducing errors.

As we argue in the remainder of this section, finding error vectors leading to bad counters is rather easy for QC-MDPC codes. We start with the following proposition (which can be trivially proven, taking into account that $\mathbf{H}$ is made of circulant blocks).

Proposition 3. *Let* $\mathbf{H} \in \mathbb{F}_2^{p \times n_0 p}$ *be the parity-check matrix of a QC-MDPC code. Then, for any* $\ell \in [0; n-1]$ *and any pair* $i, j \in \mathrm{Supp}\,(\mathbf{H}_{:,\ell})$, *we have* $\gamma_{i,j} \geq 1$.

Remember that, as stated in Lemma 1, high values of $\gamma_{i,j}$ have a bad influence on the counters. Hence, as a consequence of the above proposition, we expect that an error vector whose support is contained in the support of a column of $\mathbf{H}$ leads to large number of bad counters. To formalize this claim, we consider the following proposition.

Proposition 4. *Let* $\mathscr{C} \in \mathcal{QC}\text{-}\mathcal{MDPC}(n_0, p, v)$ *with parity-check matrix* $\mathbf{H}$. *Let* $\mathbf{e} \in \mathbb{F}_2^{n_0 p}$ *with weight* $\tilde{t} < v$, *and such that* $\mathrm{Supp}\,(\mathbf{e}) \subseteq \mathrm{Supp}\,(\mathbf{H}_{:,z})$ *for some* z. *Furthermore, assume that*

$$\begin{cases} \sum_{\ell \in \mathrm{Supp}(\mathbf{H}_{:,i})} \left\lfloor \frac{\langle \mathbf{H}_{\ell,:}^{(i)}\ ;\ \mathbf{e}^{(i)} \rangle}{2} \right\rfloor = 0 & \forall i \in \mathrm{Supp}\,(\mathbf{e}), \\ \\ \sum_{\ell \in \mathrm{Supp}(\mathbf{H}_{:,i})} \left\lfloor \frac{\langle \mathbf{H}_{\ell,:}\ ;\ \mathbf{e} \rangle}{2} \right\rfloor = 0 & \forall i \in \mathrm{Supp}\,(\mathbf{H}_{:,z}) \setminus \mathrm{Supp}\,(\mathbf{e}). \end{cases}$$

Then, the following relations hold

$$\begin{cases} \sigma_i \leq v + 1 - \tilde{t} & \text{if } i \in \mathrm{Supp}\,(\mathbf{e}), \\ \sigma_i \geq \tilde{t} & \text{if } i \in \mathrm{Supp}\,(\mathbf{H}_{:,z}) \setminus \mathrm{Supp}\,(\mathbf{e}). \end{cases}$$

Proof. The proof is a straightforward application of Lemma 1 and Proposition 3. We start with the case $i \in \mathrm{Supp}\,(\mathbf{e})$ and consider that, by hypothesis, we have $\zeta_i^{(1)}(\mathbf{H}, \mathbf{e}) = \sum_{j \in \mathrm{Supp}(\mathbf{e}) \setminus \{i\}} \gamma_{i,j}$. Since the support of $\mathbf{e}$ is contained in $\mathrm{Supp}\,(\mathbf{H}_{:,z})$, as a consequence of Proposition 3 we have $\gamma_{i,j} \geq 1$ for any pair of indexes $i, j \in \mathrm{Supp}\,(\mathbf{e})$, and hence $\zeta_i^{(1)}(\mathbf{H}, \mathbf{e}) = \sum_{j \in \mathrm{Supp}(\mathbf{e}) \setminus \{i\}} \gamma_{i,j} \geq \tilde{t} - 1$. Then, from Lemma 1 we get $\sigma_i = v - \zeta_i^{(1)}(\mathbf{H}, \mathbf{e}) \leq v + 1 - \tilde{t}$. Analogously, for the case $i \in \mathrm{Supp}\,(\mathbf{H}_{:,z}) \setminus \mathrm{Supp}\,(\mathbf{e})$, we have $\zeta_i^{(0)}(\mathbf{H}, \mathbf{e}) = \sum_{j \in \mathrm{Supp}(\mathbf{e})} \gamma_{i,j} \geq \tilde{t}$, and hence we get $\sigma_i = \zeta_i^{(0)}(\mathbf{H}, \mathbf{e}) \geq \tilde{t}$. □

As an application of the above proposition, we see that increasing $\tilde{t}$ will worsen the counters' behavior: namely, the counters values will become lower for error positions, and higher for the correct positions which are indexed by the column of $\mathbf{H}$ but not by the error vector. In particular, if we choose $\tilde{t} \geq \lceil \frac{v+3}{2} \rceil$, then we will get $\sigma_i \leq \lfloor v/2 \rfloor$ for all $i \in \mathrm{Supp}\left(\mathbf{e}\right)$, and $\sigma_i \geq \lceil \frac{v+3}{2} \rceil$ for all $i \in \mathrm{Supp}\left(\mathbf{H}_{:,z}\right) \setminus \mathrm{Supp}\left(\mathbf{e}\right)$. To flip the bits indexed by $\mathrm{Supp}\left(\mathbf{e}\right)$, we are going to need a threshold that is not higher than $\lceil v/2 \rceil$, but this will also trigger wrong flips for all positions $i \in \mathrm{Supp}\left(\mathbf{H}_{:,z}\right) \setminus \mathrm{Supp}\left(\mathbf{e}\right)$. Hence, in such a case, there does not exist a threshold that is sufficiently low to perform correct flips, but also high enough to guarantee that wrong flips do not happen. We point out that an important hypothesis in Proposition 4 is that the values of $\zeta_i^{(0)}(\mathbf{H}, \mathbf{e})$ and $\zeta_i^{(1)}(\mathbf{H}, \mathbf{e})$ only depend on the $\gamma_{i,j}$ values. In general, this is not true and one has to consider also the number of overlapping ones between the error vector and the rows of $\mathbf{H}$. Yet, as we show in the next section, the behavior of the counters remains somehow bad and these vectors cause failures with high probability.

Finally, we comment about the decoding of error vectors with weight $t > v$, but such that their support intersects with the support of a column in $\mathbf{H}$ in a sufficiently large number $\tilde{t}$ of positions. As a difference with the situation we have previously examined, the decoder must now correct more errors. In other words, we can write $\mathbf{e} = \hat{\mathbf{e}} + \check{\mathbf{e}}$, where $\hat{\mathbf{e}}$ and $\check{\mathbf{e}}$ have disjoint supports and $\hat{\mathbf{e}}$ is such that its support has size $\tilde{t}$ and is contained in the support of a column of $\mathbf{H}$. It is very unlikely that these additional errors (i.e., those due to $\check{\mathbf{e}}$) can improve the situation, up to the point that the decoder flips any of the bits in $\hat{\mathbf{e}}$. In the best case scenario, we expect that the decoder may be able to identify the error positions due to $\check{\mathbf{e}}$, but will not be able to flip any of the positions due to $\hat{\mathbf{e}}$. Hence, decoding will fail with very high probability also in this case. Notice that, with a simple counting argument, one finds that the number of errors with weight t and such that their supports intersect in $\tilde{t}$ elements with that of a column in $\mathbf{H}$ (say, the first one) is given by

$$\left|\left\{\mathbf{e} \in B_{n,t},\ \text{such that}\ \left|\mathrm{Supp}\left(\mathbf{e}\right) \cap \mathrm{Supp}\left(\mathbf{H}_{:,0}\right)\right| = \tilde{t}\right\}\right| = \binom{v}{\tilde{t}}\binom{n_0 p - v}{t - \tilde{t}}. \quad (4)$$

In general terms, the possibility to decode successfully depends on many factors (such as the decoder setting) which we have not considered yet. In other words, even if for a vector $\mathbf{e}$ we have $\delta(\mathbf{e}) > \lceil v/2 \rceil$, this does not imply that $\mathbf{e}$ can be corrected. Actually, we expect that a vector with a sufficiently large number of overlapping positions with a column of $\mathbf{H}$ is "*harder to decode*", with respect to a completely random vector. Hence, even moderately low values of $\tilde{t}$ may lead to rather high decoding failure probabilities. This in turn provides us with an operative method to generate error vector families which are expected to be harder to decode. As a consequence, through the use of numerical simulations to estimate the concrete DFR of these error families, we are able to obtain a lower bound on the DFR of any iterative BF-like decoder, as we state in the following proposition.

Proposition 5 (DFR lower bound).
Let $\mathscr{C} \in \mathcal{QC}$-$\mathcal{MDPC}(n_0, p, v)$, with parity-check matrix $\mathbf{H} \in \mathbb{F}_2^{p \times n_0 p}$. Let Dec be a BF-like decoder employed in the McEliece channel with parameter $t > v$, and consider the following procedure:

1. *for any $\tilde{t} \in [1, v]$, generate a large number of vectors with weight t and exactly $\tilde{t} \in [1; v]$ entries that overlap with $\mathbf{H}_{:,0}$;*
2. *simulate decoding of these vectors, and denote with $\tilde{\epsilon}_{\mathsf{Dec}}(\tilde{t})$ the estimated failure rate (that is, the ratio between the number of failure events and that of considered vectors);*
3. *compute*
$$\epsilon_{\mathsf{Dec}}^{(L)} = \sum_{\tilde{t}=1}^{v} \tilde{\epsilon}_{\mathsf{Dec}}(\tilde{t}) \frac{\binom{v}{\tilde{t}}\binom{n_0 p - v}{t - \tilde{t}}}{\binom{n_0 p}{t}}.$$

Then, $\epsilon_{\mathsf{Dec}}^{(L)}$ represents a lower bound for the DFR of Dec.

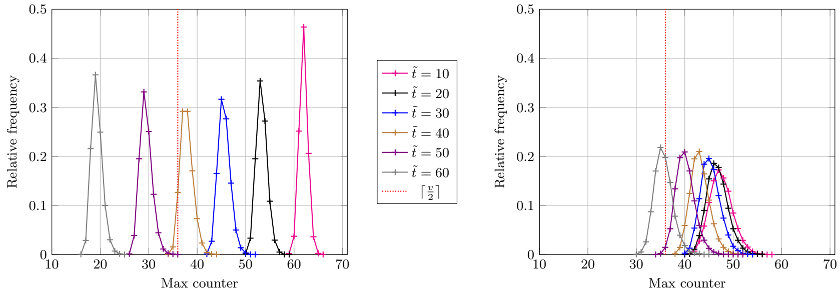
Proof. We consider error vectors with a special structure, that is, those intersecting with $\mathbf{H}_{:,0}$ in $\tilde{t} \in [1; v]$ positions. For each $\tilde{t}$, we rely on numerical simulations to estimate the probability that the decoder is not able to correct a vector of this kind, and call this probability $\tilde{\epsilon}_{\mathsf{Dec}}(\tilde{t})$. Assuming that $\tilde{\epsilon}_{\mathsf{Dec}}(\tilde{t})$ is a proper estimate of the failure probability, when considering only vectors $\mathbf{e} \in B_{n,t}$ such that $|\mathrm{Supp}\,(\mathbf{e}) \cap \mathrm{Supp}\,(\mathbf{H}_{:,0})| = \tilde{t}$, we have

$$\begin{aligned}
\epsilon_{\mathsf{Dec}}^{(L)} &= \sum_{\tilde{t}=1}^{v} \Pr\left[\mathsf{Dec}(\mathbf{e}) \neq \mathbf{0}_{\mathsf{n}}\right] \cdot \Pr\left[|\mathrm{Supp}\,(\mathbf{e}) \cap \mathrm{Supp}\,(\mathbf{H}_{:,0})| = \tilde{t} \mid \mathbf{e} \xleftarrow{\$} B_{n_0 p, t}\right] \\
&\approx \sum_{\tilde{t}=1}^{v} \tilde{\epsilon}_{\mathsf{Dec}}(\tilde{t}) \cdot \Pr\left[|\mathrm{Supp}\,(\mathbf{e}) \cap \mathrm{Supp}\,(\mathbf{H}_{:,0})| = \tilde{t} \mid \mathbf{e} \xleftarrow{\$} B_{n_0 p, t}\right] \qquad (5) \\
&= \sum_{\tilde{t}=1}^{v} \tilde{\epsilon}_{\mathsf{Dec}}(\tilde{t}) \cdot \frac{\binom{v}{\tilde{t}}\binom{n_0 p - v}{t - \tilde{t}}}{\binom{n_0 p}{t}},
\end{aligned}$$

where the last equality comes from (4). Finally, we claim that $\epsilon_{\mathsf{Dec}}^{(L)}$ is a lower bound on the DFR since there may be other vectors that cause a decoding failure. For instance, we are not considering vectors that do not intersect with $\mathbf{H}_{:,0}$, but intersect in a large number of positions with other columns of $\mathbf{H}$. For these vectors, we expect to have the same failure rates $\tilde{\epsilon}_{\mathsf{Dec}}^{(L)}$. □

Remark 1. The bound given in the above proposition is likely to be loose. For instance, we may consider the probability that a random $\mathbf{e} \in B_{n,t}$ intersects in $\tilde{t}$ positions with at least a generic column in $\mathbf{H}$. Assuming all columns of $\mathbf{H}$ behave as random vectors with weight v and length p, for rather large values of $\tilde{t}$ we get that such a probability corresponds to

$$1 - \left(1 - \frac{\binom{v}{\tilde{t}}\binom{n_0 p - v}{t - \tilde{t}}}{\binom{n_0 p}{t}}\right)^{n_0 p} \approx n_0 p \frac{\binom{v}{\tilde{t}}\binom{n_0 p - v}{t - \tilde{t}}}{\binom{n_0 p}{t}}.$$



(a) (b)

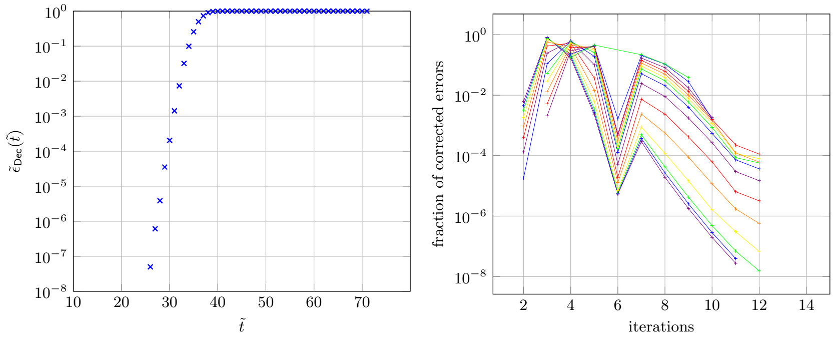
Fig. 2: Results of numerical simulations on 100 random codes, picked from the family $\mathcal{QC}\text{-}\mathcal{MDPC}(2, p, v)$ with $p = 12,323$ and $v = 71$. For each code, we have generated 100 error vectors intersecting with the first column of $\mathbf{H}$ in $\tilde{t}$ positions. Figures (a) and (b) report the measured distribution of $\max\{\sigma_i \mid i \in \mathrm{Supp}\,(\mathbf{e}) \cap \mathrm{Supp}\,(\mathbf{H}_{:,0})\}$. In (a), we have considered vectors with weight $\tilde{t}$, i.e., such that their support is fully contained in that of $\mathbf{H}_{:,0}$. In (b), we have considered vectors with weight $t = 134$ and support intersecting that of $\mathbf{H}_{:,0}$ in $\tilde{t}$ positions.

Using these probabilities in (5) (instead of the term $\binom{v}{\tilde{t}}\binom{n_0p-v}{t-\tilde{t}}/\binom{n_0p}{t}$), we would obtain an increase on the value of $\epsilon_{\mathsf{Dec}}^{(L)}$ by a factor n_0p. However, this approach leads to multiple counting of the same vectors. We expect that the obtained probabilities are not much higher than the actual ones, yet, using them would prevent us from claiming that $\epsilon_{\mathsf{Dec}}^{(L)}$ is a provable lower bound.

Remark 2. As anticipated in the Introduction, a similar analysis has been independently and concurrently performed by Vasseur in his PhD thesis [42, Chapter 16]. Namely, Vasseur has denoted as *near-codewords* the error patterns producing syndromes with unusually low weight. The effect of near-codewords on the counters distribution has been motivated by the results of numerical simulations, which are reported in [42, Table 16.2]. It can be easily seen that the error vectors we have considered in this section can be deemed as near-codewords, since with very high probability a rather large number of cancellations happen in the syndrome computation. However, as a significant difference with [42], in this paper we have provided a quantitative justification to for the counters behaviours, through Lemma 1 and Propositions 3 and 4.

4.2 Results for QC-MDPC$(2, p, v)$ codes

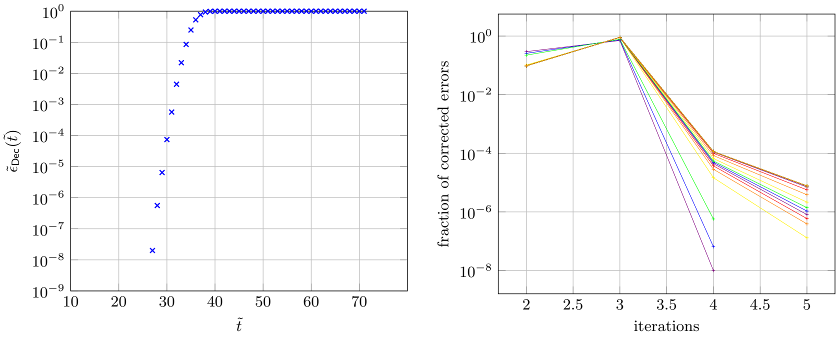
We first consider the counters distribution for error vectors whose support intersects that of a column of $\mathbf{H}$ in $\tilde{t}$ positions. As we have already said, due to overlapping ones with rows of $\mathbf{H}$, we expect the counters values to be slightly better than what we have considered in Proposition 4.



(a)

(b)

Fig. 3: Numerical simulations on the Backflip decoder as in the BIKE v3.2 specification, with maximum number of iterations set to 100. The sample DFR was estimated running either at least 10^8 decoding actions, or collecting at least 100 decoding failures, whichever event happened first. Figure (b) reports the number of iterations taken to decode an input, for all the inputs which were correctly decoded.



(a)

(b)

Fig. 4: Numerical simulations on the BGF decoder as in the BIKE v4.1 specification, with maximum number of iterations set to 5. The sample DFR was estimated running either at least 10^8 decoding actions, or collecting at least 100 decoding failures, whichever event happened first. Figure (b) reports the number of iterations taken to decode an input, for all the inputs which were correctly decoded.

Yet, due to sparsity, we expect that the number of such overlapping elements is low, so that the effect in the counters values is rather limited. To validate this assertion, we have run numerical simulations on the family of QC-MDPC codes with $n_0 = 2$, $p = 12,323$ and $v = 71$, employed in the McEliece channel with $t = 134$. Note that these parameters correspond to the ones of BIKE, version 4.1 [4], considered also in [18]. The obtained results are reported in Fig. 2. We notice that, regardless of the weight of the error vector, when the intersection between the error vector and a column of $\mathbf{H}$ increases, the maximum counter becomes lower. Hence, as a consequence, we expect that the failure probability increases, as well.

In order to validate the analysis reported in the previous section, we have applied Proposition 5 on two improved BF decoders, namely, the *backflip* proposed in [36] and the *Black Gray Flip (BGF)* proposed in [18], also used for decryption in BIKE [4]. Both decoders have been considered for codes with $v = 71$ and a McEliece channel with $t = 134$. We have analyzed both decoders for the values of p in the proposals of the BIKE cryptosystem [4], respectively in version 3.2 and 4.1, that is $p = 12,323$ for the BGF decoder and $p = 11,779$ for the backflip decoder.

We have performed numerical simulations to obtain the values of $\tilde{\epsilon}_{\mathsf{Dec}}(\tilde{t})$, stopping each simulation after having registered 100 decoding failures for each value of $\tilde{t}$ or having realized at least 100M decoding computations. In order to cope with the significant computation time requirements, we have parallelized the decoder calculus, distributing it through the OpenMP framework, thus resulting in some additional decoding computations beyond the 100M being occasionally performed. We tested 10 random codes with the same parameters detecting no relevant change in the results. For both decoders we report, in Figs. 3 and 4, the values of $\tilde{\epsilon}_{\mathsf{Dec}}(\tilde{t})$ as a function of $\tilde{t}$, and the number of iterations taken by the decoder whenever the error was correctly decoded. In the figures we additionally report the number of iterations taken by each decoder when a correct decoding computation took place, over the $\approx$ 330M decoded error vectors for each decoder; each colored line reports the data for a specific value of $\tilde{t}$ for which we have determined $\tilde{\epsilon}_{\mathsf{Dec}}(\tilde{t}) > 10^{-8}$.

Remark 3. An interesting experimental note regarding the computational efficiency and effectiveness of the decoding process of the BGF and backflip decoder concerns the number of iterations which they require to correct an error. While the BGF decoder employs all the 5 iterations for which it has been designed, all the iterations above the 12-th in the backflip decoder were useless in our simulations. Indeed, no error was corrected with a number of iterations between 13 and 100. This provides an interesting insight with respect to [37], where it is stated that adding iterations beyond the 20-th in a backflip decoder should significantly improve its expected DFR. Indeed, when considering the approach of [37], which extrapolates the low-DFR behavior of the decoder from experimentally simulable points, our results would imply that a 20 iteration backflip decoder behaves as a 100 iteration one (as the simulated results match). This in turn would lead to the DFR extrapolation of $2^{-97.65}$ being true also for the 100 iteration variant

Table 1: Summary of the DFR bounds found in this work, compared with the claimed values in [36, 18, 4].

Decoder	**Backflip** [36]	**BGF** [18, 4]
$(\mathbf{p}, \mathbf{v}, \mathbf{t})$	$(\mathbf{11779}, \mathbf{71}, \mathbf{134})$	$(\mathbf{12323}, \mathbf{71}, \mathbf{134})$
Claimed DFR	2^{-128}	2^{-128}
$\epsilon_{\mathsf{ML}}^{(L)}$	$2^{-425.86}$	$2^{-430.45}$
$\epsilon_{\mathsf{Dec}}^{(L)}$	$2^{-166.3}$	$2^{-168.06}$

of the backflip decoder. The non monotone trend in the number of iterations of the backflip decoder finds an explanation in the flipping Time-To-Live (TTL) of the procedure (which reverts a bit flip after a given TTL has expired): indeed the TTL during the overwhelming majority of our numerical simulations was found to be set to 5 for flips taking place in the first iteration.

Employing the results on $\tilde{\epsilon}_{\mathsf{Dec}}(\tilde{t})$ obtained through simulations in the expression of $\epsilon_{\mathsf{Dec}}^{(L)}$ given in Proposition 5, we are able to provide lower bounds on the DFR of these algorithms, which are shown in Table 1. The reported values differ from the ones of $\epsilon_{\mathsf{ML}}^{(L)}$ by a factor of $\approx 2^{259}$, in turn showing how a significant amount of failure events, at very low DFR values, are not due to near-codewords (as claimed in [35]). Indeed, our reported data are able to set a reliable lower bound on the DFR, through Proposition 5, in turn showing that iterative decoders perform significantly worse than the ML decoder. We note that the lower bounds we provide do not explicitly contradict the numerical claims on the DFR for both the backflip and the BGF decoder with the parameters at hand. We also note that obtaining concrete values for $\tilde{\epsilon}_{\mathsf{Dec}}(\tilde{t})$ for values of $\tilde{t} < 25$ may bring the value of our lower bound further up.

5 Conclusion

We have proposed two approaches for bounding the performance of iterative decoders derived from Gallager's BF, and used in decoding QC-MDPC codes in code-based cryptosystems. The first approach relies on modeling the ML decoder performance, which is an optimal decoder and hence provides an ultimate bound on the behavior of any sub-optimal decoder, such as the BF ones. This also allows to characterize the asymptotic DFR of these decoders, which has been shown to decay polynomially in the code length. The second approach exploits a numerically-aided procedure to provide a lower bound on the DFR of BF decoders: the approach relies on numerical estimations for the DFR of families of error vectors which are harder to decode for BF decoders. Through weighing the contribution to the total DFR of such error families with their size we achieve a lower bound on the DFR for the specific class of iterative decoders derived from BF. In particular, this second approach was shown to provide tighter lower bounds to the DFR by a factor of 2^{259} with respect to the bound obtained modeling the performance of the ML decoder, thus providing a preliminary quantitative assessment of the performance gap of the iterative BF decoders and their ideal ML counterpart on QC-MDPC parameters of interest in code-based cryptography.

References

[1] G. Alagic et al. *Status Report on the Second Round of the NIST Post-Quantum Cryptography Standardization Process.* `https://csrc.nist.gov/publications/detail/nistir/8309/final`.

[2] M. R. Albrecht et al. *Classic McEliece: Conservative Code-Based Cryptography.* `https://classic.mceliece.org/`.

[3] D. Apon et al. “Cryptanalysis of LEDAcrypt”. In: *Advances in Cryptology – CRYPTO 2020.* Ed. by D. Micciancio and T. Ristenpart. Cham: Springer International Publishing, 2020, pp. 389–418.

[4] N. Aragon et al. *BIKE: Bit Flipping Key Encapsulation.* `https://bikesuite.org`.

[5] M. Baldi et al. “A finite regime analysis of information set decoding algorithms”. In: *Algorithms* 12.10, 209 (2019).

[6] M. Baldi. *QC-LDPC Code-Based Cryptography.* SpringerBriefs in Electrical and Computer Engineering. Springer International Publishing, 2014.

[7] M. Baldi et al. “A failure rate model of bit-flipping decoders for QC-LDPC and QC-MDPC code-based cryptosystems”. In: *Proc. 17th International Joint Conference on e-Business and Telecommunications (ICETE), Secrypt 2020, 17th International Conference on Security and Cryptography.* Paris, France, 8-10 July 2020, pp. 238–249.

[8] M. Baldi et al. *LEDAcrypt: Low-dEnsity parity-check coDe-bAsed cryptographic systems.* `https://www.ledacrypt.org/`.

[9] M. Baldi et al. “Security of generalised Reed–Solomon code-based cryptosystems”. In: *IET Information Security* 13.4 (2019), pp. 404–410.

[10] A. Becker et al. “Decoding random binary linear codes in $2^{n/20}$: How $1 + 1 = 0$ improves information set decoding”. In: *Advances in Cryptology - EUROCRYPT 2012.* Ed. by D. Pointcheval and T. Johansson. Vol. 7237. Lecture Notes in Computer Science. Springer, 2012, pp. 520–536.

[11] E. R. Berlekamp, R. J. McEliece, and H. C. A. van Tilborg. “On the inherent intractability of certain coding problems”. In: *IEEE Transactions on Information Theory* 24.3 (1978), pp. 384–386.

[12] L. Both and A. May. “Decoding linear codes with high error rate and its impact for LPN Security”. In: *Post-Quantum Cryptography, PQCrypto 2018.* Ed. by T. Lange and R. Steinwandt. Vol. 10786. Lecture Notes in Computer Science. Springer, 2018, pp. 25–46.

[13] R. Canto-Torres and J. Tillich. “Speeding up decoding a code with a non-trivial automorphism group up to an exponential factor”. In: *Proc. IEEE International Symposium on Information Theory (ISIT 2019).* Paris, France, 7-12 July 2019, pp. 1927–1931.

[14] M.-S. Chen, T. Chou, and M. Krausz. “Optimizing BIKE for the Intel Haswell and ARM Cortex-M4”. In: *IACR Transactions on Cryptographic Hardware and Embedded Systems* 2021.3 (July 2021), pp. 97–124. DOI: `10.46586/tches.v2021.`

i3.97-124. URL: https://tches.iacr.org/index.php/TCHES/article/view/8969.

[15] A. Couvreur et al. “Distinguisher-based attacks on public-key cryptosystems using Reed-Solomon codes”. In: *Designs, Codes and Cryptography* 73.2 (2014), pp. 641–666.

[16] S. Das. “A brief note on estimates of binomial coefficients”. In: *URL: http://page. mi. fu-berlin. de/shagnik/notes/binomials. pdf* (2016).

[17] N. Drucker and S. Gueron. “A toolbox for software optimization of QC-MDPC code-based cryptosystems”. In: *Journal of Cryptographic Engineering* 9.4 (2019), pp. 341–357.

[18] N. Drucker, S. Gueron, and D. Kostic. “QC-MDPC decoders with several shades of gray”. In: *Post-Quantum Cryptography, PQCrypto 2020*. Ed. by J. Ding and J.-P. Tillich. Lecture Notes in Computer Science. Springer, Cham, 2020, pp. 35–50.

[19] E. Eaton et al. “QC-MDPC: A timing attack and a CCA2 KEM”. In: *Post-Quantum Cryptography, PQCrypto 2018*. Ed. by T. Lange and R. Steinwandt. Vol. 10786. Lecture Notes in Computer Science. Springer International Publishing, 2018, pp. 47–76.

[20] J.-C. Faugère et al. “Algebraic cryptanalysis of McEliece variants with compact keys”. In: *Advances in Cryptology – EUROCRYPT 2010*. Ed. by H. Gilbert. Vol. 6110. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2010, pp. 279–298.

[21] R. G. Gallager. *Low-Density Parity-Check Codes*. M.I.T. Press, 1963.

[22] Q. Guo, T. Johansson, and P. Stankovski. “A key recovery attack on MDPC with CCA security using decoding errors”. In: *Advances in Cryptology – ASIACRYPT 2016*. Ed. by J. H. Cheon and T. Takagi. Vol. 10031. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2016, pp. 789–815.

[23] D. Hofheinz, K. Hövelmanns, and E. Kiltz. “A modular analysis of the Fujisaki-Okamoto transformation”. In: *Theory of Cryptography,TCC 2017*. Ed. by Y. Kalai and L. Reyzin. Vol. 10677. Lecture Notes in Computer Science. Springer, Cham, 2017, pp. 341–371.

[24] K. Khathuria, J. Rosenthal, and V. Weger. “Encryption scheme based on expanded Reed-Solomon codes”. In: *Advances in Mathematics of Communications* 15.2 (2021), pp. 207–218.

[25] C. Löndahl et al. “Squaring attacks on McEliece public-key cryptosystems using quasi-cyclic codes of even dimension”. In: *Designs, Codes and Cryptography* 80.2 (2016), pp. 359–377.

[26] R. J. McEliece. “A public-key cryptosystem based on algebraic coding theory.” In: *DSN Progress Report* (1978), pp. 114–116.

[27] National Institute of Standards and Technology. *NIST Post-Quantum Standardization Process*. https://csrc.nist.gov/Projects/Post-Quantum-Cryptography. 2017.

[28] S. Ouzan and Y. Be’ery. *Moderate-density parity-check codes*. https://arxiv.org/abs/0911.3262. Sept. 2009.

[29] G. Poltyrev. “Bounds on the decoding error probability of binary linear codes via their spectra”. In: *IEEE Transactions on Information Theory* 40.4 (1994), pp. 1284–1292.

[30] R. Misoczki, J.-P. Tillich, N. Sendrier, P. S. L. M. Barreto. “MDPC-McEliece: New McEliece variants from moderate density parity-check codes”. In: *Proc.*

IEEE International Symposium on Information Theory (ISIT 2013). Istambul, Turkey, 7-12 July 2013, pp. 2069–2073.

[31] P. Santini et al. “Analysis of reaction and timing attacks against cryptosystems based on sparse parity-check codes”. In: *Code-Based Cryptography, CBC 2019*. Ed. by M. Baldi, E. Persichetti, and P. Santini. Vol. 11666. Lecture Notes in Computer Science. Springer, Cham, 2019, pp. 115–136.

[32] P. Santini et al. “Analysis of the error correction capability of LDPC and MDPC codes under parallel bit-flipping decoding and application to cryptography”. In: *IEEE Transactions on Communications* 68.8 (2020), pp. 4648–4660.

[33] P. Santini et al. “Hard-decision iterative decoding of LDPC codes with bounded error rate”. In: *Proc. IEEE International Conference on Communications (ICC 2019)*. Shanghai, China, 20-24 May 2019.

[34] N. Sendrier. “Decoding one out of many”. In: *Post-Quantum Cryptography, PQCrypto 2011*. Ed. by B.-Y. Yang. Vol. 7071. Lecture Notes in Computer Science. Springer, Berlin, Heidelberg, 2011, pp. 51–67.

[35] N. Sendrier and V. Vasseur. “About low DFR for QC-MDPC decoding”. In: *Post-Quantum Cryptography, PQCrypto 2020*. Ed. by J. Ding and J. Tillich. Vol. 12100. Lecture Notes in Computer Science. Springer, Cham, 2020, pp. 20–34.

[36] N. Sendrier and V. Vasseur. “On the decoding failure rate of QC-MDPC bit-flipping decoders”. In: *Post-Quantum Cryptography, PQCrypto 2019*. Ed. by J. Ding and R. Steinwandt. Vol. 11505. Lecture Notes in Computer Science. Springer, Cham, 2019, pp. 404–416.

[37] N. Sendrier and V. Vasseur. *On the existence of weak keys for QC-MDPC decoding*. Cryptology ePrint Archive, Report 2020/1232. `https://eprint.iacr.org/2020/1232`. 2020.

[38] V. M. Sidelnikov and S. O. Shestakov. “On insecurity of cryptosystems based on generalized Reed-Solomon codes”. In: *Discrete Mathematics and Applications* 2.4 (1992), pp. 439–444.

[39] J.-P. Tillich. “The decoding failure probability of MDPC codes”. In: *Proc. IEEE International Symposium on Information Theory (ISIT 2018)*. Vail, CO, USA, 17-22 June 2018, pp. 941–945.

[40] R. C. Torres and N. Sendrier. “Analysis of information set decoding for a sublinear error weight”. In: *Post-Quantum Cryptography, PQCrypto 2016*. Vol. 9606. Lecture Notes in Computer Science. Springer, Cham, 2016, pp. 144–161.

[41] A. Vardy. “The intractability of computing the minimum distance of a code”. In: *IEEE Transactions on Information Theory* 43.6 (1997), pp. 1757–1766.

[42] V. Vasseur. “Post-quantum cryptography: study on the decoding of QC-MDPC codes”. PhD thesis. Mar. 2021.

Appendix A: Proof of Theorem 1

We focus on ML decoding, and derive an analytical expression for its DFR. To this end, we consider an input $\mathbf{x} = \mathbf{c} + \mathbf{e} \in \mathbb{F}_2^n$, with $\mathbf{c} \xleftarrow{\$} \mathscr{C}$ and $\mathbf{e} \xleftarrow{\$} B_{n,t}$. The decoder first computes $\mathscr{C}^{(\mathbf{x})}$, that is, the set of all codewords that are t away from $\mathbf{x}$, and then outputs at random one of them. Given that, clearly, $\mathbf{c} \in \mathscr{C}^{(\mathbf{x})}$, and that decoding fails every time the decoder output is different from $\mathbf{c}$, we have that a failure happens with probability

$$\frac{\left|\mathscr{C}^{(\mathbf{x})}\right|-1}{\left|\mathscr{C}^{(\mathbf{x})}\right|}.$$

Note that, if there is only one codeword in $\mathscr{C}^{(\mathbf{x})}$, then this codeword must be $\mathbf{c}$; hence, in this case, we never have a failure. To obtain the DFR, which we denote as ϵ_{ML}, we average the above probability over all the possible errors $\mathbf{e} \in B_{n,t}$, added to all the codewords in $\mathscr{C}$. According to Definition 4, we assume uniform distributions for both $\mathbf{c}$ and $\mathbf{e}$, and hence obtain

$$\epsilon_{\mathsf{ML}} = \frac{1}{2^k\binom{n}{t}}\sum_{\mathbf{e}\in B_{n,t}}\sum_{\mathbf{c}\in\mathscr{C}}\frac{\left|\mathscr{C}^{(\mathbf{c}+\mathbf{e})}\right|-1}{\left|\mathscr{C}^{(\mathbf{c}+\mathbf{e})}\right|}.$$

Now we show that, due to linearity, we can consider that the transmitted codeword corresponds to $\mathbf{0}_n$. Indeed, for each codeword $\mathbf{c} \in \mathscr{C}$ and any $\mathbf{e} \in B_{n,t}$, we have

$$\begin{aligned}\mathscr{C}^{(\mathbf{c}+\mathbf{e})} &= \left\{\mathbf{a}\in\mathscr{C} \text{ s.t. } \mathrm{dist}(\mathbf{c}+\mathbf{e},\mathbf{a})=t\right\} = \left\{\mathbf{a}\in\mathscr{C} \text{ s.t. } \mathrm{dist}(\mathbf{c}+\mathbf{a},\mathbf{e})=t\right\}\\ &= \left\{\mathbf{a}'\in\mathscr{C} \text{ s.t. } \mathrm{dist}(\mathbf{a}',\mathbf{e})=t\right\} = \mathscr{C}^{(\mathbf{e})}.\end{aligned}$$

From this observation, we further obtain

$$\begin{aligned}\epsilon_{\mathsf{ML}} &= \frac{1}{2^k\binom{n}{t}}\sum_{\mathbf{e}\in B_{n,t}}\sum_{\mathbf{c}\in\mathscr{C}}\frac{\left|\mathscr{C}^{(\mathbf{c}+\mathbf{e})}\right|-1}{\left|\mathscr{C}^{(\mathbf{c}+\mathbf{e})}\right|} = \frac{1}{2^k\binom{n}{t}}\sum_{\mathbf{e}\in B_{n,t}}\sum_{\mathbf{a}'\in\mathscr{C}}\frac{\left|\mathscr{C}^{(\mathbf{e})}\right|-1}{\left|\mathscr{C}^{(\mathbf{e})}\right|}\\ &= \frac{1}{2^k\binom{n}{t}}\sum_{\mathbf{e}\in B_{n,t}}2^k\frac{\left|\mathscr{C}^{(\mathbf{e})}\right|-1}{\left|\mathscr{C}^{(\mathbf{e})}\right|} = \frac{1}{\binom{n}{t}}\sum_{\mathbf{e}\in B_{n,t}}\frac{\left|\mathscr{C}^{(\mathbf{e})}\right|-1}{\left|\mathscr{C}^{(\mathbf{e})}\right|}\\ &= 1-\frac{1}{\binom{n}{t}}\sum_{\mathbf{e}\in B_{n,t}}\frac{1}{\left|\mathscr{C}^{(\mathbf{e})}\right|}.\end{aligned}$$

We now proceed by proving the lower and upper bounds on the DFR. Based on the above aconsiderations, we consider the transmission of the null codeword over the McEliece channel. The output of the channel, which is given as input to the decoder, corresponds to a weight t vector, uniformly distributed over $B_{n,t}$. Decoding fails every time the decoder outputs a codeword which is not the null one. Clearly, for any $\mathbf{e} \in B_{n,t}$, we necessarily have $\mathbf{0}_n \in \mathscr{C}^{(\mathbf{e})}$: hence, a decoding failure may happen only when $\mathscr{C}^{(\mathbf{e})}$ contains at least two codewords. Notice that we can express the decoding failure rate as

$$\epsilon_{\mathsf{ML}} = \frac{1}{\binom{n}{t}}\sum_{\mathbf{e}\in B_{n,t}}\Pr\left[\mathsf{ML}(\mathbf{e})\neq\mathbf{0}_{\mathsf{n}}\right] = \frac{1}{\binom{n}{t}}\sum_{\mathbf{e}\in B_{n,t}}\sum_{\mathbf{c}\in\mathscr{C}\setminus\mathbf{0}_n}\Pr\left[\mathsf{ML}(\mathbf{e})=\mathbf{c}\right].$$

Consider that

$$\mathrm{dist}(\mathbf{c},\mathbf{e}) = \mathrm{wt}(\mathbf{c}) + t - 2\alpha \quad \text{and} \quad \mathrm{dist}(\mathbf{c},\mathbf{e}) \in [\mathrm{wt}(\mathbf{c}) - \mathrm{t}; \mathrm{wt}(\mathbf{c}) + \mathrm{t}],$$

where $\alpha = |\mathrm{Supp}\,(\mathbf{e}) \cap \mathrm{Supp}\,(\mathbf{c})|$ and, clearly, $0 \leq \alpha \leq \min\{t, \mathrm{wt}(\mathbf{c})\}$. In particular, $\mathbf{c}$ will be at distance t from $\mathbf{e}$ only when $2\alpha = \mathrm{wt}(\mathbf{c})$. Then, the following claims can be straightforwardly proven:

i) if $\mathrm{wt}(\mathbf{c})$ is odd, then $\mathbf{c} \notin \mathscr{C}^{(\mathbf{e})}$;

ii) if $\mathrm{wt}(\mathbf{c}) > 2t$, then $\mathrm{dist}(\mathbf{c},\mathbf{e}) > t$ and thus $\mathbf{c} \notin \mathscr{C}^{(\mathbf{e})}$;

iii) if $\mathrm{wt}(\mathbf{c})$ is even and $\leq 2t$, then, by a counting argument on the number of elements of $\mathrm{Supp}\,(\mathbf{e})$ and $\mathrm{Supp}\,(\mathbf{c})$ that coincide, we have that

$$|\{\mathbf{e} \in B_{n,t} \mid \mathrm{dist}(\mathbf{c},\mathbf{e}) = t\}| = \binom{\mathrm{wt}(\mathbf{c})}{\mathrm{wt}(\mathbf{c})/2}\binom{n - \mathrm{wt}(\mathbf{c})}{t - \mathrm{wt}(\mathbf{c})/2};$$

iv) if $\mathbf{e}$ is such that $\mathbf{c} \notin \mathscr{C}^{(\mathbf{e})}$, then $\Pr\left[\mathsf{ML}(\mathbf{e}) = \mathbf{c}\right] = 0$, otherwise

$$\Pr\left[\mathsf{ML}(\mathbf{e}) = \mathbf{c}\right] = \frac{1}{\left|\mathscr{C}^{(\mathbf{e})}\right|} \leq \frac{1}{2},$$

since $\mathscr{C}^{(\mathbf{e})}$ contains at least two codewords.

By putting everything together, we get

$$\begin{aligned}
\epsilon_{\mathsf{ML}} &= \frac{1}{\binom{n}{t}} \sum_{\mathbf{c} \in \mathscr{C} \setminus \mathbf{0}_n} \sum_{\mathbf{e} \in B_{n,t}} \Pr\left[\mathsf{ML}(\mathbf{e}) = \mathbf{c}\right] = \frac{1}{\binom{n}{t}} \sum_{\mathbf{c} \in \mathscr{C} \setminus \mathbf{0}_n} \sum_{\substack{\mathbf{e} \in B_{n,t} \\ \mathbf{c} \in \mathscr{C}^{(\mathbf{e})}}} \Pr\left[\mathsf{ML}(\mathbf{e}) = \mathbf{c}\right] \\
&= \frac{1}{\binom{n}{t}} \sum_{\mathbf{c} \in \mathscr{C} \setminus \mathbf{0}_n} \sum_{\substack{\mathbf{e} \in B_{n,t} \\ \mathbf{c} \in \mathscr{C}^{(\mathbf{e})}}} \frac{1}{\left|\mathscr{C}^{(\mathbf{e})}\right|} \leq \frac{1}{2\binom{n}{t}} \sum_{\mathbf{c} \in \mathscr{C} \setminus \mathbf{0}_n} |\{\mathbf{e} \in B_{n,t} \mid \mathrm{dist}(\mathbf{c},\mathbf{e}) = t\}| \\
&= \frac{1}{2\binom{n}{t}} \sum_{\substack{\mathbf{c} \in \mathscr{C} \setminus \mathbf{0}_n \\ \mathrm{wt}(\mathbf{c}) \in [d;2t] \\ \mathrm{wt}(\mathbf{c}) \text{ even}}} \binom{\mathrm{wt}(\mathbf{c})}{\mathrm{wt}(\mathbf{c})/2}\binom{n - \mathrm{wt}(\mathbf{c})}{t - \mathrm{wt}(\mathbf{c})/2} \\
&= \frac{1}{2\binom{n}{t}} \sum_{\substack{w \in [d;2t] \\ w \text{ even}}} A_w \binom{w}{w/2}\binom{n - w}{t - w/2} = \epsilon_{\mathsf{ML}}^{(U)},
\end{aligned}$$

where A_w is the number of codewords in $\mathscr{C}$ of weight w, and d is the minimum distance of $\mathscr{C}$.

In analogous way, we now derive a lower bound for the DFR of the ML-decoder; we start from

$$\begin{aligned}\epsilon_{\mathsf{ML}} &= \frac{1}{\binom{n}{t}}\sum_{\mathbf{e}\in B_{n,t}}\Pr\left[\mathsf{ML}(\mathbf{e})\neq \mathbf{0}_n\right] = \frac{1}{\binom{n}{t}}\sum_{\mathbf{e}\in B_{n,t}}\frac{\left|\mathscr{C}^{(\mathbf{e})}\right|-1}{\left|\mathscr{C}^{(\mathbf{e})}\right|}\\ &= \frac{1}{\binom{n}{t}}\sum_{\substack{\mathbf{e}\in B_{n,t}\\ \left|\mathscr{C}^{(\mathbf{e})}\right|\geq 2}}\frac{\left|\mathscr{C}^{(\mathbf{e})}\right|-1}{\left|\mathscr{C}^{(\mathbf{e})}\right|}\geq \frac{\left|\left\{\mathbf{e}\in B_{n,t}\;\middle|\;\left|\mathscr{C}^{(\mathbf{e})}\right|\geq 2\right\}\right|}{2\binom{n}{t}},\end{aligned}$$

where the inequality comes from the observation that, if $\left|\mathscr{C}^{(\mathbf{e})}\right|\geq 2$, we have $\frac{\left|\mathscr{C}^{(\mathbf{e})}\right|-1}{\left|\mathscr{C}^{(\mathbf{e})}\right|}\geq\frac{1}{2}$. In the above expression, we need to count the number of vectors $\mathbf{e}\in B_{n,t}$ for which $\mathscr{C}^{(\mathbf{e})}$ contains at least a codeword which is different from the null one. To avoid multiple counting of the same vector, we bound further such a quantity as follows. We have

$$\begin{aligned}&\left|\left\{\mathbf{e}\in B_{n,t}\mid \exists\mathbf{c}\in\mathscr{C}\setminus\mathbf{0}_n \text{ s.t. } \mathrm{dist}(\mathbf{c},\mathbf{e})=t\right\}\right|\\ &=\left|\bigcup_{\mathbf{c}\in\mathscr{C}\setminus\mathbf{0}_n}\{\mathbf{e}\in B_{n,t}\mid \mathrm{dist}(\mathbf{c},\mathbf{e})=t\}\right|\\ &=\left|\{\mathbf{e}\in B_{n,t}\mid \mathrm{dist}(\mathbf{c}^*,\mathbf{e})=t\}\cup\left(\bigcup_{\mathbf{c}\in\mathscr{C}\setminus\{\mathbf{0}_n,\mathbf{c}^*\}}\{\mathbf{e}\in B_{n,t}\mid \mathrm{dist}(\mathbf{c},\mathbf{e})=t\}\right)\right|\\ &\geq\left|\{\mathbf{e}\in B_{n,t}\mid \mathrm{dist}(\mathbf{c}^*,\mathbf{e})=t\}\right|,\end{aligned}$$

for any non null codeword $\mathbf{c}^*$. Notice that the above quantity depends only on the weight of the considered $\mathbf{c}^*$. Let $w=\mathrm{wt}(\mathbf{c}^*)$: if w is odd or $w\notin[d,2t]$, then $|\{\mathbf{e}\in B_{n,t}\mid \mathrm{dist}(\mathbf{c}^*,\mathbf{e})=t\}|=0$, otherwise

$$|\{\mathbf{e}\in B_{n,t}\mid \mathrm{dist}(\mathbf{c}^*,\mathbf{e})=t\}|=\binom{w}{w/2}\binom{n-w}{t-w/2}.$$

Since the above inequality holds for any codeword $\mathbf{c}^*$ of proper weight, we can write $\epsilon_{\mathsf{ML}}\geq\epsilon_{\mathsf{ML}}^{(L)}$, where

$$\epsilon_{\mathsf{ML}}^{(L)}=\max_{\substack{w\in[d,2t]\\ w\text{ even}\\ A_w>0}}\left\{\frac{\binom{w}{w/2}\binom{n-w}{t-w/2}}{2\binom{n}{t}}\right\}.$$

Notice that if $\mathscr{C}$ does not contain a codeword with even weight not larger than $2t$, then the expression of $\epsilon_{\mathsf{ML}}^{(L)}$ becomes meaningless (i.e., it becomes 0).

To conclude the proof, we show that the MLS decoder has the same DFR of the ML decoder. Let $\mathbf{x} = \mathbf{c} + \mathbf{e}$, with $\mathbf{c} \in \mathscr{C}$ and $\mathbf{e} \in B_{n,t}$, be the received sequence. The probability that ML, on input $\mathbf{x}$, outputs a codeword which is different from $\mathbf{c}$ is equal to $1 - \left|\mathscr{C}^{(\mathbf{e})}\right|^{-1}$. The MLS decoder, on input $\mathbf{s} = \mathbf{x}\mathbf{H}^\top$, fails with probability $1 - \left|\mathcal{S}_{\mathbf{H}}^{(\mathbf{x})}\right|^{-1}$. Note that $\mathscr{C}^{(\mathbf{e})}$ contains all codewords $\mathbf{c}' \neq \mathbf{c}$ such that $\mathrm{dist}(\mathbf{e}, \mathbf{c}') = t$; thus, we have that $\mathbf{e}' = \mathbf{c}' + \mathbf{e}$ has weight t and syndrome $\mathbf{e}'\mathbf{H}^\top = \mathbf{e}\mathbf{H}^\top = \mathbf{s}$, so that $\mathbf{e}' \in \mathcal{S}_{\mathbf{H}}^{(\mathbf{x})}$. Then, we have that $\left|\mathscr{C}^{(\mathbf{e})}\right| \leq \left|\mathcal{S}_{\mathbf{H}}^{(\mathbf{x})}\right|$. Now, for each $\mathbf{e}' \in \mathcal{S}_{\mathbf{H}}^{(\mathbf{x})}$, we have that $\mathbf{e}\mathbf{H}^\top = \mathbf{e}'\mathbf{H}^\top$, from which $(\mathbf{e}+\mathbf{e}')\mathbf{H}^\top = \mathbf{0}$; hence $\mathbf{c}'' = \mathbf{e} + \mathbf{e}' \in \mathscr{C}$. Now, consider that

$$\mathbf{x} + \mathbf{e}' = \mathbf{c} + \mathbf{e} + \mathbf{e}' = \mathbf{c} + \mathbf{c}'' = \hat{\mathbf{c}} \in \mathscr{C},$$

and that

$$\mathrm{dist}(\hat{\mathbf{c}}, \mathbf{x}) = \mathrm{wt}(\hat{\mathbf{c}} + \mathbf{x}) = \mathrm{wt}(\mathbf{c} + \mathbf{c}'' + \mathbf{c} + \mathbf{e}) = \mathrm{wt}(\mathbf{e} + \mathbf{e}' + \mathbf{e}) = \mathrm{wt}(\mathbf{e}') = t,$$

thus $\hat{\mathbf{c}} \in \mathscr{C}^{(\mathbf{e})}$. This shows that, for any candidate in $\mathcal{S}_{\mathbf{H}}^{(\mathbf{x})}$, we also have a candidate in $\mathscr{C}^{(\mathbf{e})}$, and vice versa: this proves that $\left|\mathscr{C}^{(\mathbf{e})}\right| = \left|\mathcal{S}_{\mathbf{H}}^{(\mathbf{x})}\right|$.

Appendix B: Proof of Proposition 1

Let $\mathscr{C} \in \mathcal{QC}\text{-}\mathcal{MDPC}(n_0, p, v)$, and denote with $\mathbf{H}$ its parity-check matrix formed by circulant blocks of weight v. Let $\mathbf{H}_i$ denote the i-th circulant block in $\mathbf{H}$. For $i_0, i_1 \in \{0, 1, \cdots, n_0 - 1\}$, with $i_0 \neq i_1$, and $\ell \in \{0, 1, \cdots, p-1\}$, consider a vector $\mathbf{c}^{(i_0,i_1,\ell)}$ in the form

$$\mathbf{c}^{(i_0,i_1,\ell)} = [\mathbf{c}_0^{(i_0,i_1,\ell)}, \mathbf{c}_1^{(i_0,i_1,\ell)}, \cdots, \mathbf{c}_{n_0-1}^{(i_0,i_1,\ell)}],$$

where

$$\mathbf{c}_j^{(i_0,i_1,\ell)} = \begin{cases} \mathbf{0}_p & \text{if } j \neq i_0, i_1, \\ \text{the transpose of the } \ell\text{-th column of } \mathbf{H}_{i_1} & \text{if } j = i_0, \\ \text{the transpose of the } \ell\text{-th column of } \mathbf{H}_{i_0} & \text{if } j = i_1. \end{cases}$$

It is easily seen that $\mathbf{c}^{i_0,i_1,\ell}\mathbf{H}^\top = \mathbf{0}_p$, hence $\mathbf{c}^{i_0,i_1,\ell} \in \mathscr{C}$. Furthermore, $\mathbf{c}^{i_0,i_1,\ell}$ has weight $2v$: this proves that $\mathscr{C}$ cannot have a minimum distance larger than $2v$. Consider now that the number of vectors $\mathbf{c}^{i_0,i_1,\ell}$ is given by the number of choices for i_0, i_1 and ℓ, which is equal to $p\binom{n_0}{2}$. This proves that $\mathscr{C}$ contains at least $p\binom{n_0}{2}$ codewords with weight $2v$. Clearly, we cannot exclude that there are more codewords with this weight (even if this is rather unlikely), so we can only claim that $A_{2v} \geq p\binom{n_0}{2}$.

Appendix C: Derivation of Equation (3)

We here show how (3) can be obtained. We start by specializing the expression of $\epsilon_{\mathsf{ML}}^{(L)}$ for the case of $n_0 = 2$. Remember that the code always contains codewords with weight $w = 2v$, so that we can write

$$\epsilon_{\mathsf{ML}}^{(L)} = \frac{\binom{2v}{v}\binom{2p-2v}{t-v}}{2\binom{2p}{t}}$$

For the binomials appearing in the above expression, we are going to use the following well known (for instance, see [16]) approximations

$$\binom{2v}{v} = \frac{2^{2v}}{\sqrt{\pi v}}\left(1 + o(1)\right), \tag{6}$$

$$\binom{2p-2v}{t-v} = \frac{1}{\sqrt{2\pi(t-v)}}\left(\frac{(2p-2v)e}{(t-v)}\right)^{t-v}\left(1 + o(1)\right), \tag{7}$$

$$\binom{2p}{t} = \frac{1}{\sqrt{2\pi t}}\left(\frac{2pe}{t}\right)^{t}\left(1 + o(1)\right), \tag{8}$$

where e is Euler's number. Neglecting the $o(1)$ terms and expressing (6) as a power of 2, we get

$$\binom{2v}{v} \approx 2^{2v-0.5\log_2(v)-0.8257}.$$

From (7) and (8), we obtain

$$\begin{aligned}\frac{\binom{2p-2v}{t-v}}{\binom{2p}{t}} &= \frac{1}{\sqrt{1-\frac{v}{t}}}\left(\frac{(2p-2v)e}{(t-v)}\right)^{t-v}\left(\frac{2pe}{t}\right)^{-t}\left(1+o(1)\right)\\ &= \frac{e^{-v}}{\sqrt{1-\frac{v}{t}}}\left(\frac{(2p-2v)}{(t-v)}\right)^{t-v}\left(\frac{2p}{t}\right)^{-t}\left(1+o(1)\right)\end{aligned}$$

To further simplify, we consider $t \approx 2v$, from which

$$\begin{aligned}\frac{\binom{2p-2v}{t-v}}{\binom{2p}{t}} &\approx \frac{e^{-v}}{\sqrt{0.5}}\left(\frac{(2p-2v)}{v}\right)^{v}\left(\frac{p}{v}\right)^{-2v}\\ &= \frac{e^{-v}}{\sqrt{0.5}}\left(\frac{2p}{v}-2\right)^{v}\left(\frac{p}{v}\right)^{-2v}\\ &= 2^{-1.4427v+0.5+v\log_2\left(\frac{2p}{v}-2\right)-2v\log_2\left(\frac{p}{v}\right)}.\end{aligned}$$

Since $\frac{2p}{v} \gg 2$, we further have

$$\begin{aligned}\frac{\binom{2p-2v}{t-v}}{\binom{2p}{t}} &\approx 2^{-1.4427v+0.5+v\log_2\left(\frac{2p}{v}\right)-2v\log_2\left(\frac{p}{v}\right)}\\ &= 2^{-0.4427v+0.5-v\log_2\left(\frac{p}{v}\right)}\end{aligned}$$

Putting everything together, we get

$$\frac{\binom{2v}{v}\binom{2p-2v}{t-v}}{2\binom{2p}{t}} \approx 2^{1.5573v - v\log_2\left(\frac{p}{v}\right) - 0.5\log_2(v) - 1.3257}$$

Appendix D: Proof of Lemma 1

To avoid confusion, in this proof we use " $\oplus$ " to indicate the sum in the binary finite field, and the operator " $+$ " to indicate the standard sum over the integers ring. We denote with c_j the value of the j-th parity-check equation $c_j = \bigoplus_{i=0}^{n-1} e_i h_{i,j}$, and we have

$$c_j = 1 \iff \langle \mathbf{H}_{:,j}\ ;\ \mathbf{e}\rangle \ \text{ is odd.}$$

Recalling that the i-th counter σ_i corresponds to the number of unsatisfied parity-check equations in which the i-th bit participates, that is

$$\sigma_i = \sum_{j \in \mathrm{Supp}(\mathbf{H}_{:,i})} c_j = \left|\{j \in \mathrm{Supp}\left(\mathbf{H}_{:,i}\right) \mid \langle \mathbf{H}_{j,:}\ ;\ \mathbf{e}\rangle \ \text{ is odd}\}\right|,$$

whenever $e_i = 1$ we have

$$\sigma_i = \mathrm{wt}(\mathbf{H}_{:,i}) - \left|\left\{j \in \mathrm{Supp}\left(\mathbf{H}_{:,i}\right) \mid \langle \mathbf{H}_{j,:}^{(i)}, \mathbf{e}^{(i)}\rangle \ \text{ is odd}\right\}\right|.$$

We notice that, for each non negative integer a, it results

$$a - 2\left\lfloor \frac{a}{2} \right\rfloor = \begin{cases} 1 & \text{if } a \text{ is odd,} \\ 0 & \text{if } a \text{ is even.} \end{cases}$$

Let $\hbar_{j,\ell}$ denote the lifted entry $h_{i,j}$ (i.e., with value in $\{0;1\} \subseteq \mathbb{Z}$), and consider the following chain of equalities

$$\begin{aligned}
\sigma_i &= \left|\left\{j \in \mathrm{Supp}\left(\mathbf{H}_{:,i}\right) \mid \langle \mathbf{H}_{j,:}^{(i)}\ ;\ \mathbf{e}^{(i)}\rangle \ \text{ is odd}\right\}\right| \\
&= \sum_{j \in \mathrm{Supp}\left(\mathbf{H}_{:,i}\right)} \langle \mathbf{H}_{j,:}^{(i)}; \mathbf{e}^{(i)}\rangle - 2\left\lfloor \frac{\langle \mathbf{H}_{j,:}^{(i)}; \mathbf{e}^{(i)}\rangle}{2} \right\rfloor \\
&= \sum_{j \in \mathrm{Supp}\left(\mathbf{H}_{:,i}\right)} \sum_{\ell \in \mathrm{Supp}\left(\mathbf{e}^{(i)}\right)} \hbar_{j,\ell} - 2\left\lfloor \frac{\langle \mathbf{H}_{j,:}^{(i)}; \mathbf{e}^{(i)}\rangle}{2} \right\rfloor \\
&= \left(\sum_{\ell \in \mathrm{Supp}(\mathbf{e}) \setminus \{i\}} \sum_{j \in \mathrm{Supp}\left(\mathbf{H}_{:,i}\right)} \hbar_{j,\ell} \right) - \sum_{j \in \mathrm{Supp}\left(\mathbf{H}_{:,i}\right)} 2\left\lfloor \frac{\langle \mathbf{H}_{j,:}^{(i)}\ ;\ \mathbf{e}^{(i)}\rangle}{2} \right\rfloor \\
&= \sum_{\ell \in \mathrm{Supp}(\mathbf{e}) \setminus \{i\}} \gamma_{i,\ell} - \sum_{j \in \mathrm{Supp}\left(\mathbf{H}_{:,i}\right)} 2\left\lfloor \frac{\langle \mathbf{H}_{j,:}^{(i)}\ ;\ \mathbf{e}^{(i)}\rangle}{2} \right\rfloor.
\end{aligned}$$

Putting all the previous inferences together, the thesis of the Lemma can be easily derived. When $e_i = 0$, the thesis of the Lemma can be proved with analogous reasoning.

Appendix E: Extended Simulation Results

Remark: The contents of this appendix are not included in the CBCrypto 2021 publication, and report the results of an extended simulation campaign on the numerical estimation of the DFR of the Black-Gray Flip decoder for BIKE v4.1 – Category 1 parameters.

These results were made possible by the publication of a highly optimized version of the BIKE v4.1 decoder [14], which in turn allowed us to extend our simulations beyond the ones for the CBCrypto 2021 paper. To this end, we integrated the aforementioned decoder in the software package provided at `https://bikesuite.org/`, and modified the error generating function so that $\tilde{t}$ error positions would be placed as per our analysis. To allow third party reproduction of our results, we employed as a seed of the PRNG the result of the POSIX `time` function call, i.e., the number of seconds since the midnight of 1970-01-01. We report the value of the seeds employed to allow results reproduction, and provide the employed codebase at `https://www.ledacrypt.org/DFR_additional_sim_CBCrypto.tar.gz`. The computation generating the data reported in Figure 5 ran in ≈ 69 core-days on a dual socket AMD Epyc 7551 server, while the computation generating the data in Figure 6 ran in ≈ 694 core-days on the same machine.

In the following, we further evaluate two points: the effect on $\tilde{\epsilon}_{\mathsf{Dec}}(\tilde{t})$ of picking different random QC-MDPC codes, and the effect of simply simulating a larger number of decoding actions, to the end of detecting failures for lower values of $\tilde{t}$.

Figure 5 reports the results of simulating 10 randomly selected QC-MDPC codes, showing that, while picking different QC-MDPC codes provides variations in the DFR against the error patterns we analyzed, such differences are contained in less than an order of magnitude in the DFR values in the 10^{-8}–10^{-7} range.

Figure 6 reports the results of more extensive simulations on three randomly selected QC-MDPC codes, one picked from the set of Figure 5, and two re-drawn at random. As it can be seen, the behaviour of more frequent failures we observed, is still consistent when considering further lower numbers of intersections between the error and the columns of the parity check matrix H.

Finally, we report the results of computing the value of $\epsilon^{(L)}_{\mathsf{Dec}}$ considering the methodology described in Section 4 in Table 2. Our results highlight how the lower bound on we obtain on the DFR is consistent across multiple codes, and tends to rise as more simulations are performed to estimate the DFR of error patterns having a lower value for $\tilde{t}$. Futhermore, we highlight that a cyclic shift of the error pattern does not change its behaviour with respect to the induction of decoding failures. As a consequence, for each error pattern found, all its p cyclic shifts will report the same measured failure rate. Taking into account this factor in the computation of $\epsilon^{(L)}_{\mathsf{Dec}}$ we obtain a higher estimate for our DFR bound, with respect to neglecting cyclic shift effects. We note that such an estimate does not consider the possibility for cyclic shifts of different errors to be matching, although this fact is expected to be negligible. Whilst considering

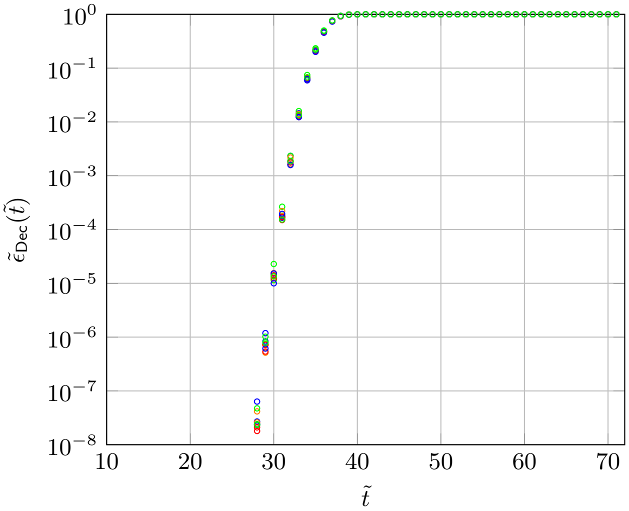


Fig. 5: Numerical simulations on the BGF decoder as in the BIKE v4.1 specification, with maximum number of iterations set to 5 and Category 1 parameters. The sample DFR was estimated running either at least 10^8 decoding actions, or collecting at least 100 decoding failures, whichever event happened first. The depicted results represent 10 randomly selected QC-MDPC codes, one for each marker colour.

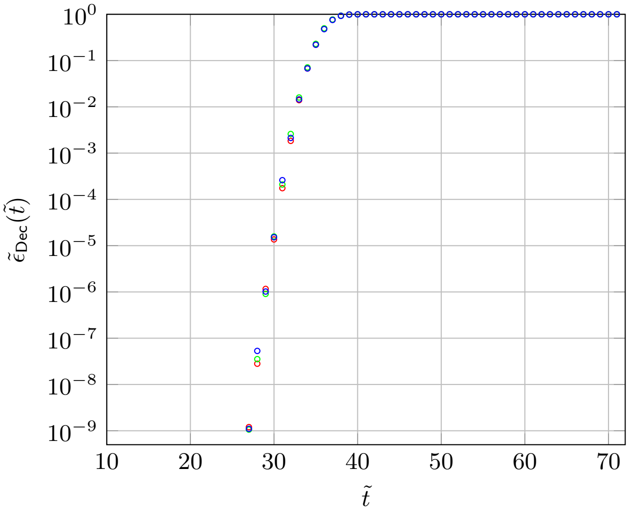


Fig. 6: Numerical simulations on the BGF decoder as in the BIKE v4.1 specification, with maximum number of iterations set to 5 and Category 1 parameters. The sample DFR was estimated running either at least 10^{10} decoding actions, or collecting at least 30 decoding failures, whichever event happened first. The depicted results represent 3 randomly selected QC-MDPC codes, one for each marker colour.

the aforementioned point, we note that the estimates for $\epsilon^{(L)}_{\mathsf{Dec}}$ which take into account the effects of quasi cyclicity exceed the claimed DFR of 2^{-128}. Finally, we observe that, in case the trend linking the failure rate and value of $\tilde{t}$ were to hold even for values of $\tilde{t}$ where currently no simulation results are available ($\tilde{t} < 27$), this would in turn imply an increase of the value of the lower bound for the DFR moving them in the range of $2^{-90} - 2^{-88}$ not considering the quasi cyclicity and in the $2^{-77} - 2^{-75}$ range considering quasi cyclicity.

Table 2: Lower bounds on the DFR values obtained as per the description in Section 4. The last column of the table takes into account the fact that, for each determined error pattern, all its cyclic shifts exhibit the same increased DFR behavior. Lower bounds above 2^{-128} are in boldface

PRNG seed	**Number of computed decodes**	$\epsilon_{\mathsf{Dec}}^{(L)}$ not considering quasi-cyclicity	$\epsilon_{\mathsf{Dec}}^{(L)}$ considering quasi-cyclicity
1633020036	10^8	$2^{-141.01}$	$\mathbf{2^{-127.42}}$
1633031832	10^8	$2^{-142.16}$	$2^{-128.57}$
1633056215	10^8	$2^{-141.59}$	$2^{-128.00}$
1633074398	10^8	$2^{-142.35}$	$2^{-128.76}$
1633098805	10^8	$2^{-141.94}$	$2^{-128.35}$
1633123135	10^8	$2^{-141.30}$	$\mathbf{2^{-127.71}}$
1633138995	10^8	$2^{-142.17}$	$2^{-128.58}$
1633163838	10^8	$2^{-142.37}$	$2^{-128.78}$
1633188683	10^8	$2^{-142.22}$	$2^{-128.63}$
1633213349	10^8	$2^{-142.08}$	$2^{-128.49}$
1632496983	10^{10}	$2^{-140.58}$	$\mathbf{2^{-126.99}}$
1632493786	10^{10}	$2^{-140.63}$	$\mathbf{2^{-127.04}}$
1633020036	10^{10}	$2^{-140.38}$	$\mathbf{2^{-126.79}}$