Designs, Codes and Cryptography (2023) 91:4019–4058
https://doi.org/10.1007/s10623-023-01270-0

Coding and bounds for partially defective memory cells

Haider Al Kim[1,2] · **Sven Puchinger**[3] · **Ludo Tolhuizen**[4] · **Antonia Wachter-Zeh**[1]

Received: 2 July 2022 / Revised: 14 March 2023 / Accepted: 28 June 2023 /
Published online: 24 August 2023

Abstract
This paper considers coding for so-called *partially stuck (defect)* memory cells. Such memory cells can only store partial information as some of their levels cannot be used fully due to, e.g., wearout. First, we present new constructions that are able to mask u partially stuck cells while correcting at the same time t random errors. The process of "masking" determines a word whose entries coincide with writable levels at the (partially) stuck cells. For $u > 1$ and alphabet size $q > 2$, our new constructions improve upon the required redundancy of known constructions for $t = 0$, and require less redundancy for masking partially stuck cells than former works required for masking fully stuck cells (which cannot store any information). Second, we show that treating some of the partially stuck cells as erroneous cells can decrease the required redundancy for some parameters. Lastly, we derive Singleton-like, sphere-packing-like, and Gilbert–Varshamov-like bounds. Numerical comparisons state that our constructions match the Gilbert–Varshamov-like bounds for several code parameters, e.g., BCH codes that contain all-one word by our first construction.

Keywords Flash memories · Non-volatile memories · Defective memory · (Partially) Stuck cells · Error-correcting codes · Gilbert-Varshamov bound

Mathematics Subject Classification 94B20 · 94B65 · 94B60

Communicated by C. Mitchell.

This work has received funding from the German Research Foundation (Deutsche Forschungsgemeinschaft, DFG) under Grant No. WA3907/1-1, the German Academic Exchange Service (Deutscher Akademischer Austauschdienst, DAAD) under the support program ID 57381412, and the European Union's Horizon 2020 research and innovation program through the Marie Sklodowska-Curie under Grant No. 713683. This article was presented in part at the 2019 International Symposium Problems of Redundancy in Information and Control Systems (Redundancy), at the 17th International Workshop on Algebraic and Combinatorial Coding Theory (ACCT2020), and also as an extended abstract at the 2021 and 2023 annual Non-Volatile Memories Workshop (NVMW).

✉ Haider Al Kim
haider.alkim@tum.de

Extended author information available on the last page of the article

1 Introduction

The demand for reliable memory solutions and in particular for non-volatile memories such as flash memory and *phase change memories* (PCMs) for different applications is steadily increasing. These multi-level devices provide permanent storage and a rapidly extendable capacity. Recently developed devices exploit an increased number of cell levels while at the same time the physical size of the cells was decreased. Therefore, coding and signal processing solutions are essential to overcome reliability issues. The key characteristic of PCM cells is that they can switch between two main states: an amorphous state and a crystalline state. PCM cells may become *defect* (also called *stuck*) [8, 13, 17, 20] if they fail in switching their states. This occasionally happens due to the cooling and heating processes of the cells. Therefore, cells can only hold a single phase [8, 20]. In multi-level PCM cells, failure may occur at a position in either of extreme states or in the partially programmable states of crystalline.

The work [29] investigates codes that mask so-called *partially stuck* (partially defective) cells, i.e., cells which cannot use all levels. For multi-level PCMs, the case in which the partially stuck level $s = 1$ is particularly important since this means that a cell can reach all crystalline sub-states, but cannot reach the amorphous state.

Figure 1 depicts the general idea of reliable and (partially) defective memory cells. It shows two different cell level representations: Representation 1 forms the binary extension field $\mathbb{F}_{2^2}$ and Representation 2 forms the set of integers modulo $q = 4$, i.e., $\mathbb{Z}/4\mathbb{Z}$.

1.1 Related Work

Coding for memories with stuck cells, also known as *defect-correcting codes* for *memories with defects*, dates back to the 1970 s, cf. the work by Kuznetsov and Tsybakov [15]. They proposed binary defect-correcting codes in finite and asymptotic regimes whose required redundancy is at least the number of defects. Later works [2–7, 12, 14, 16, 19, 25–27] investigated the problem of defective cells under various aspects: binary and non-binary, only defect-correcting coding and error-and-defect-correcting coding, and finite and asymptotic length analysis.

In binary defect-correcting coding models, e.g. [2–5, 16, 19, 26], the authors dealt with masking stuck cells without considering additional substitution errors. In these studies, it is unclear if the proposed constructions are optimal in terms of their required redundancy. The works [12, 14, 27] considered masking stuck memory cells while at the same time correcting potential random errors. In [12], so-called partitioned cyclic code and partitioned BCH codes were proposed for this task.

The asymptotic model of stuck-cell-masking codes also received considerable attention in the previously mentioned papers. Moreover, there is work devoted to asymptotically optimal codes for a fixed number of defects [6] or for a number of defects proportional to the codeword length [7]. The proposed constructions, for example [7, Section 4] and its extended version in [7, Section 5] that can additionally correct substitution errors, show that u check symbols are sufficient for masking u defects. However, they use codes with a property that is not well studied in coding theory. Therefore, we do not dwell on [6] and [7], and also our goal is to obtain code constructions for finite code length n.

The recent work [29] considers *partially* stuck memory cells (see Figure 1. C), and improves upon the redundancy necessary for masking compared to all prior works for classical stuck cells. However, the paper does not consider error correction in addition to masking.

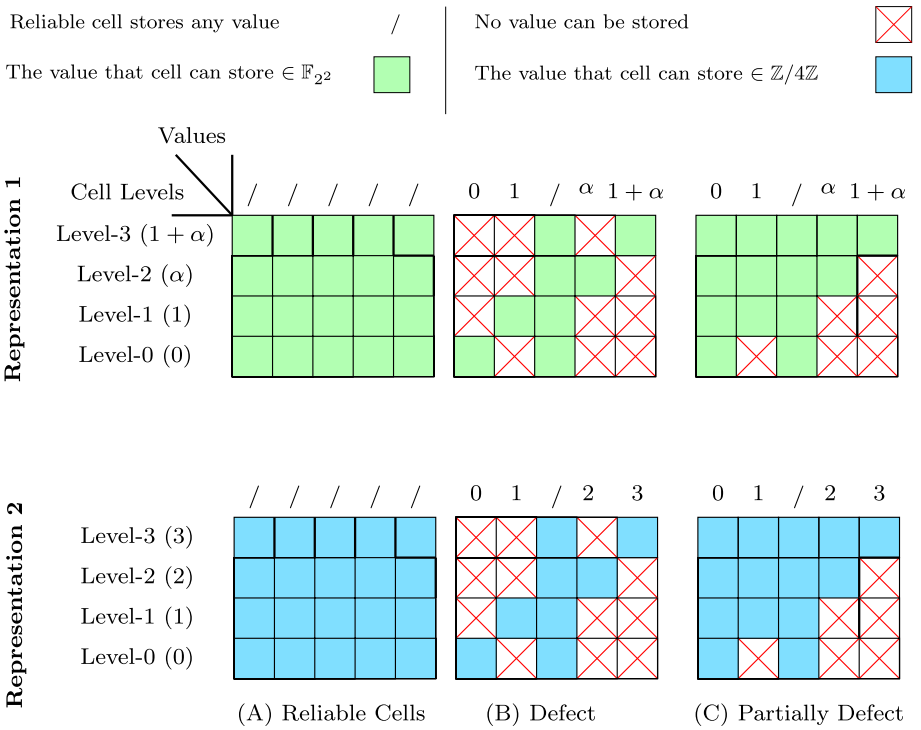

Fig. 1 Illustration of reliable and (partially) defective memory cells. In this figure, there are $n = 5$ cells with $q = 4$ possible levels. The cell levels $\in \mathbb{F}_{2^2}$ are mapped to $(0, 1, \alpha$ or $1 + \alpha)$ shown in Representation 1 or $\in \mathbb{Z}/4\mathbb{Z}$ are mapped to $(0, 1, 2$ or $3)$ shown in Representation 2. Case (A) illustrates fully reliable cells which can store any of the four values in both representations. In the stuck scenario as shown in case (B), the defective cells can store only the exact stuck level s. Case (C) is more flexible (partially defective scenario). Partially stuck cells at level $s \geq 1$ can store level s or higher

1.2 Our contribution

In this paper, we consider the problem of combined error correction and masking of partially stuck cells. Compared to the conventional stuck-cell case in [12], we reduce the redundancy necessary for masking, similar to the results in [29], and even reduce further compared to [29, Construction 5].

If cells are partially stuck at level 1, we can simply use a $(q-1)$-ary error correcting code as mentioned in [29, Section III]. However, this approach could require too much redundancy if a cell is partially stuck at different levels rather than 1. For instance, using $(q-s)$-ary codes for $2 \leq s \leq q-1$ reduces the cardinality of the code because exempting s out of the available q levels is quite expensive. Further, for relatively few partially stuck-at-1 cells, even a $(q-1)$-ary error correcting code is not a competitor to our constructions (cf. Figure 6). Therefore, considering sophisticated coding schemes is favorable.

We provide code constructions for any number of partially stuck cells; see Table 3 for an overview of our constructions and their required redundancies. For the error-free case, where only masking is necessary, our redundancies coincide with those from [29] or are even smaller.

Our paper also investigates a technique where the encoder, after a first masking step, introduces errors at some partially stuck positions of a codeword in order to satisfy the stuck-at constraints. The decoder uses part of the error-correcting capability to correct these introduced errors.

We also derive bounds on our code constructions, namely a Singleton-type, sphere-packing-type, and Gilbert-Varshamov-type bounds. We provide a numerical analysis by comparing our code constructions and the derived bounds with other trivial codes and known limits.

Our focus is on long codes over small alphabets, i.e., the code length n is larger than the field size q. Otherwise, one could instead mask by a code of length $n < q$ (by using, e.g., [24]).

The remainder of the paper is arranged as follows. In Sect. 2, we provide notations and define the models of joint errors and partially defective cells examined in this study. Our code constructions along with their encoding and decoding algorithms are presented in Sect. 3 and 4. Section 5 generalizes the previous constructions to mask partially stuck cells at any arbitrary level and correct errors additionally. Section 6 investigates exchanging error correction capability toward more partially stuck cells masking possibility. Upper- and lower-like bounds on our constructions are derived in Sect. 7 and 8, respectively. In Sect. 9, we provide numerical and analytical comparisons. Finally, Sect. 10 concludes this work.

2 Preliminaries

2.1 Notations

For a prime power q, let $\mathbb{F}_q$ denote the finite field of order q and $\mathbb{F}_q[x]$ be the set of all univariate polynomials with coefficients in $\mathbb{F}_q$. For $g, f \in \mathbb{Z}_{>0}$, denote $[f] = \{0, 1, \ldots, f-1\}$ and $[g, f] = \{g, g+1, \ldots, f-1\}$.

As usual, an $[n, k, d]_q$ code is a linear code over $\mathbb{F}_q$ of length n, dimension k and minimum (Hamming) distance d. The (Hamming) weight $\mathrm{wt}(\boldsymbol{x})$ of a vector $\boldsymbol{x} \in \mathbb{F}_q^n$ equals its number of non-zero entries.

In order to simplify notation, we sometimes identify $x \in \mathbb{F}_q$ with the number of field elements not larger than x, that is, with the integer $q - |\{y \in \mathbb{F}_q \mid x \geq y\}|$. The meaning of x will be clear from the context. Figure 1 depicts the two representations that are equivalent in this sense. Finally, we denote the q-ary entropy function by h_q, that is

$$h_q(0) = 0, h_q(1) = \log_q(q-1), \text{ and } h_q(x) = -x\log_q(x) - (1-x)\log_q(1-x) + x\log_q(q-1) \text{ for } 0 < x < 1.$$

2.2 Definitions

2.2.1 Defect and partially defect cells

A cell is called defect (*stuck-at level s*), if it can only store the value s. A cell is called partially defect (*partially-stuck-at level s*), if it can only store values which are at least s. Note that a cell that is partially defect at level 0 is a non-defect cell which can store any of the q levels and a cell that is partially defect at level $q-1$ is a (fully) defect cell.

2.2.2 $(n, M)_q(\Sigma, t)$ PSMC

For $\Sigma \subset \mathbb{F}_q^n$ and non-negative integer t, a q-ary (Σ, t)-*partially-stuck-at-masking code* $\mathcal{C}$ of length n and size M is a coding scheme consisting of a message set $\mathcal{M}$ of size M, an encoder $\mathcal{E}$ and a decoder $\mathcal{D}$ satisfying:

1. The encoder $\mathcal{E}$ is a mapping from $\mathcal{M} \times \Sigma$ to $\mathbb{F}_q^n$ such that
$$\text{for each } (\boldsymbol{m}, \boldsymbol{s}) \in \mathcal{M} \times \Sigma, \quad \mathcal{E}(\boldsymbol{m}, \boldsymbol{s}) \geq \boldsymbol{s},$$
2. For each $(\boldsymbol{m}, \boldsymbol{s}) \in \mathcal{M} \times \Sigma$ and each $\boldsymbol{e} \in \mathbb{F}_q^n$ such that
$$\mathrm{wt}(\boldsymbol{e}) \leq t \text{ and } \mathcal{E}(\boldsymbol{m}, \boldsymbol{s}) + \boldsymbol{e} \geq \boldsymbol{s},$$
it holds that
$$\mathcal{D}(\mathcal{E}(\boldsymbol{m}, \boldsymbol{s}) + \boldsymbol{e}) = \boldsymbol{m}.$$

2.2.3 $(n, M)_q(u, 1, t)$ PSMC

A q-ary $(u, 1, t)$ PSMC of length n and cardinality M is a q-ary (Σ, t) PSMC of length n and size M where
$$\Sigma = \{\boldsymbol{s} \in \{0, 1\}^n \mid \mathrm{wt}(\boldsymbol{s}) \leq u\}.$$
In this special case, the partially stuck-at condition means that the output of the encoder is non-zero at each position of the support $\boldsymbol{\phi}$ of $\boldsymbol{s}$.

3 Code construction for masking up to $q - 1$ partially stuck-at-1 Cells

3.1 Code construction

In this section, we present a coding scheme over $\mathbb{F}_q$ that can mask up to $q - 1$ partially stuck-at-1 cells and additionally can correct errors. We adapt the construction from [29], which allows to mask up to $q - 1$ partially stuck-at-1 ($s_i = 1$ for all i) cells with only a single redundancy symbol, but cannot correct any substitution errors.

Construction 1 *Assume that there is an $[n, k, d]_q$ code $\mathcal{C}$ with a $k \times n$ generator matrix of the form*
$$\boldsymbol{G} = \begin{bmatrix} \boldsymbol{G}_1 \\ \boldsymbol{G}_0 \end{bmatrix} = \begin{bmatrix} \boldsymbol{0}_{(k-1)\times 1} & \boldsymbol{I}_{k-1} & \boldsymbol{P}_{(k-1)\times(n-k)} \\ 1 & \boldsymbol{1}_{k-1} & \boldsymbol{1}_{n-k} \end{bmatrix},$$
where $\boldsymbol{I}_{k-1}$ is the $(k-1) \times (k-1)$ identity matrix, $\boldsymbol{P} \in \mathbb{F}_q^{(k-1)\times(n-k)}$, and $\boldsymbol{1}_\ell$ is the all-one vector of length ℓ. From the code $\mathcal{C}$, a PSMC can be obtained, whose encoder and decoder are shown in Algorithm 1 and Algorithm 2.

Theorem 1 *The coding scheme in Construction 1 is a $(q - 1, 1, \lfloor \frac{d-1}{2} \rfloor)$ PSMC of length n and cardinality q^{k-1}.*

Algorithm 1: Encoding

Input:
- Message: $\boldsymbol{m} = (m_0, m_1, \ldots, m_{k-2}) \in \mathbb{F}_q^{k-1}$
- Positions of partially stuck-at-1 cells: $\boldsymbol{\phi}$

Compute $\boldsymbol{w} = (w_1, w_2, \ldots, w_{n-1}) = \boldsymbol{m} \cdot \boldsymbol{G}_1$
Find $v \in \mathbb{F}_q \setminus \{w_i \mid i \in \boldsymbol{\phi}\}$
Compute $\boldsymbol{c} = \boldsymbol{w} - v \cdot \boldsymbol{G}_0$
Output: Codeword $\boldsymbol{c} \in \mathbb{F}_q^n$

Algorithm 2: Decoding

Input:
- Retrieve $\boldsymbol{y} = \boldsymbol{c} + \boldsymbol{e}$, $\boldsymbol{y} \in \mathbb{F}_q^n$

$\hat{\boldsymbol{c}} \leftarrow$ decode $\boldsymbol{y}$ in $\mathcal{C}$
$\hat{v} \leftarrow$ first entry of $\hat{\boldsymbol{c}}$
$\hat{\boldsymbol{w}} = (\hat{w}_0, \hat{w}_1, \cdots, \hat{w}_{n-1}) \leftarrow (\hat{\boldsymbol{c}} - \hat{v} \cdot \boldsymbol{G}_0)$
$\hat{\boldsymbol{m}} \leftarrow (\hat{w}_1, \ldots, \hat{w}_{k-1})$
Output: Message vector $\hat{\boldsymbol{m}} \in \mathbb{F}_q^{k-1}$

Proof To mask the partially stuck-at-1 positions, the codeword has to fulfill:

$$c_i \geq 1, \text{ for all } i \in \boldsymbol{\phi}. \tag{1}$$

Since $\mid \boldsymbol{\phi} \mid < q$, there is at least one value $v \in \mathbb{F}_q$ such that $w_i \neq v$, for all $i \in \boldsymbol{\phi}$. Thus, $c_i = (w_i - v) \neq 0$ and (1) is satisfied.

The decoder (Algorithm 2) gets $\boldsymbol{y}$, which is $\boldsymbol{c}$ corrupted by at most $\lfloor \frac{d-1}{2} \rfloor$ substitution errors. The decoder of $\mathcal{C}$ can correct these errors and obtain $\boldsymbol{c}$.

Due to the structure of $\boldsymbol{G}$, the first position of $\boldsymbol{c}$ equals $-v$. Hence, we can compute $\hat{\boldsymbol{w}} = \boldsymbol{w}$ (cf. Algorithm 2) and $\hat{\boldsymbol{m}} = \boldsymbol{m}$. □

Corollary 1 *If there is an* $[n, k, d]_q$ *code containing a word of weight n, then there is a* q*-ary* $(q-1, 1, \lfloor \frac{d-1}{2} \rfloor)$ *PSMC of length n and size* q^{k-1}.

3.2 Comparison to the conventional stuck-cell scenario

Theorem 1 combines [12, Theorem 1] and [29, Theorem 4] to provide a code construction that can mask partially stuck cells and correct errors. The required redundancy is a single symbol for masking plus the redundancy for the code generated by the upper part of $\boldsymbol{G}$, needed for the error correction. In comparison, [12, Theorem 1] requires at least

$$\min\{n - k \,:\, \exists\, [n, k, d]_q \text{ code with } d > u\} \geq u$$

redundancy symbols to mask u stuck cells, where the inequality follows directly from the Singleton bound.

Throughout this work, we use q-ary cyclic codes, in particular BCH codes, to describe our coding methods and to show some examples. In the following, we present Tables 1 and 2 to compare ternary cyclic codes of length $n = 8$ for masking partially stuck cells to masking stuck cells [12], both with error correction. Here, the defining set, denoted by D_c, is the set of indices of zeros of the code, that is, $\boldsymbol{c}$ is a codeword if and only if $\boldsymbol{c}(\alpha^b) = 0$ for all b in D_c, where $\alpha \in \mathbb{F}_q$ is the primitive n^{th} root of unity in $\mathbb{F}_q$. These code classes are unique,

Table 1 Ternary Codes for **Partially** Stuck-at-1 Memory Cell for $n = 8$

Cardinality	Overall redundancy	u	t	The defining set D_c
3^7	1	2	0	{4}
3^6	2	2	0	{4}
3^5	3	2	0	{5, 7}
3^4	4	2	1	{4, 5, 7}
3^3	5	2	1	{1, 2, 3, 6}
3^2	6	2	1	{1, 2, 3, 4, 6}
3^2	6	2	1	{1, 3, 4, 5, 7}
3	7	2	1	{1, 2, 3, 5, 6, 7}

practical, and explicit in their applications. The minimum distance d can be bounded from below by the BCH bound, which is the number of consecutive elements in D_c plus one, or more involved bounds such as the Hartmann-Tzeng bound [11] or the Roos bound [21].

The tables show that masking partially stuck cells requires less redundancy than masking stuck cells, both with and without additional error correction. The reason is that there is only one forbidden value in each partially stuck-at-1 cell, while there are $q - 1$ forbidden values in each stuck-at cell.

3.3 Remarks on construction 1

Remark 1 The special case of Theorem 1 with $n < q$ was used in [24] for constructing a $(q-1)$-ary error-correcting code from a q-ary Reed-Solomon code, which can be of interest if $q - 1$ is not the power of a prime.

Remark 2 The code constructions in Theorem 1 also work over the ring of integers modulo q ($\mathbb{Z}/q\mathbb{Z}$) in which q is not necessarily a prime power, similar to the construction for $u < q$ in [29].

Remark 3 According to [29, Construction 3], it is possible to further decrease the required redundancy for masking u partially stuck-at-1 cells to $1 - \log_q \lfloor \frac{q}{u+1} \rfloor$. We can use the same strategy here. Let $z = \lfloor (\frac{q}{u+1}) \rfloor$. We choose disjoint sets $A_1, A_2, .., A_z$ of size $u + 1$ in $\mathbb{F}_q$. As additional information, the encoder picks $j \in \{1, 2, \ldots, z\}$. In Step 2 of Algorithm 1, it selects v from A_j. As the decoder acquires v, it can obtain j as well.

4 Code constructions for masking more than $q - 1$ partially stuck-at-1 cells

The masking technique in the previous section only guarantees successful masking up to a number of $q - 1$ partially stuck-at-1 cells. In this section, we present two code constructions for simultaneous masking and error correction when $q \leq u < n$. One is based on the masking-only construction in [29, Construction 4] and the other is based on [29, Section VI], which are able to mask $u \geq q$ partially stuck positions, but cannot correct any errors. We generalize these constructions to be able to cope with errors. The latter construction may lead to larger code dimensions for a given pair (u, t), in a similar fashion as [29, Construction 5]

Table 2 Ternary Codes for Stuck-at Memory [12] for $n = 8$

Cardinality	Overall Redundancy	u	t	The defining set D_c
3^7	1	1	0	{0}
3^6	2	1	0	{0}
3^5	3	1	0	{5, 7}
3^4	4	1	1	{0, 1, 3}
3^3	5	1	1	{1, 2, 3, 6}
3^2	6	1	2	{0, 1, 2, 3, 6}
3^2	6	2	1	{0, 1, 3}
3	7	2	1	{1, 2, 3, 6}

improves upon [29, Construction 4]. Further, taking $t = 0$ it achieves larger codes sizes than [29, Construction 5] if the all-one word is in the code.

4.1 Code construction over $\mathbb{F}_q$ for masking up to $q + d_0 - 3$ partially stuck cells

We recall that [29, Construction 4] can mask more than $q - 1$ partially stuck-at-1 cells and it is a generalization of the all-one vector construction [29, Theorem 4]. Hence, replacing the $\mathbf{1}_n$ vector in Theorem 1 by a parity-check matrix as in [29, Construction 4] allows masking of q or more partially stuck-at 1 cells, and correct t errors.

Construction 2 *Suppose that there is an $[n, k, d]_q$ code $\mathcal{C}$ with a $k \times n$ generator matrix of the following form:*

$$\boldsymbol{G} = \begin{bmatrix} \boldsymbol{G}_1 \\ \boldsymbol{H}_0 \end{bmatrix}$$

where $\boldsymbol{H}_0 \in \mathbb{F}_q^{l \times n}$ is a parity-check matrix of an $[n, n - l, d_0]$ code $\mathcal{C}_0$. From the code $\mathcal{C}$, a PSMC can be obtained, whose encoder and decoder are shown in Algorithm 3 and Algorithm 4.

Theorem 2 *The coding scheme in Construction 2 is a $(d_0 + q - 3, 1, \lfloor \frac{d-1}{2} \rfloor)$ PSMC of length n and cardinality q^{k-l}.*

Algorithm 3: Encoding

Input:
- Message: $\boldsymbol{m} \in \mathbb{F}_q^{k-l}$
- Positions of partially stuck-at-1 cells: $\boldsymbol{\phi}$

1 Compute $\boldsymbol{w} = (w_1, w_2, \ldots, w_{n-1}) = \boldsymbol{m} \cdot \boldsymbol{G}_1$

2 Find $\left\{ \boldsymbol{z} = (z_0, \ldots, z_{l-1}) \in \mathbb{F}_q^l \;\middle|\; (\boldsymbol{w} + \boldsymbol{z}\boldsymbol{H}_0)_i \neq 0, \text{ for all } i \in \boldsymbol{\phi} \right\}$.

3 Compute $\boldsymbol{c} = \boldsymbol{w} + \boldsymbol{z} \cdot \boldsymbol{H}_0$

Output: Codeword $\boldsymbol{c} \in \mathbb{F}_q^n$

Algorithm 4: Decoding

Input: $\boldsymbol{y} = \boldsymbol{c} + \boldsymbol{e} \in \mathbb{F}_q^n$

1 $\hat{\boldsymbol{c}} \leftarrow$ decode $\boldsymbol{y}$ in the code $\mathcal{C}$

2 Determine $\hat{\boldsymbol{m}} \in \mathbb{F}_q^{k-l}$ and $\hat{\boldsymbol{z}} \in \mathbb{F}_q^l$ such that $\hat{\boldsymbol{c}} = \hat{\boldsymbol{m}}\boldsymbol{G}_1 + \hat{\boldsymbol{z}}\boldsymbol{H}_0$.

Output: Message vector $\hat{\boldsymbol{m}} \in \mathbb{F}_q^{k-l}$

Proof For the masking part, the proof is a simple modification of [29, Theorem 7]. In Sect. 5, we give a full proof of Proposition 2, which generalizes Construction 2. The error correction part of the proof follows the proof of Theorem 1. □

The gain of Theorem 2 in the number of partially stuck cells that can be masked comes at the cost of larger redundancy. However, the redundancy is still smaller than the redundancy of the construction for masking stuck-at cells and error correction in [12]. In particular, let $\mathcal{C}$ be an $[n, k, d \geq 2t + 1]$ code containing an $[n, l]_q$ subcode $\mathcal{C}_0$ for which $\mathcal{C}_0^{\perp}$ has minimum distance d_0. With Theorem 2, we obtain a $(d_0 + q - 3, 1, \lfloor \frac{d-1}{2} \rfloor)$ PSMC of length n and cardinality q^{k-l}. The construction in [12] yields a coding scheme with equal cardinality, allowing for masking up to $d_0 - 1$ fully stuck cells and correcting $\lfloor \frac{d-1}{2} \rfloor$ errors since the minimum distance d_1 defined in [12] for the error correction capability of the code is d in our notation. Hence, exactly $q - 2$ more cells that are partially-stuck-at levels 1 than classical stuck cells can be masked.

Example 1 We apply Construction 2 to mask up to $u = 4$ partially stuck cells over $\mathbb{F}_4$ and $\boldsymbol{m} \in \mathbb{F}_4^9$. Let α be a primitive element in $\mathbb{F}_{16}$ and let $\mathcal{C}$ be the $[15, 12, 3]_4$ code with zeros α^0 and α^1. Let $\mathcal{C}_0$ be the $[15, 3]$ subcode of $\mathcal{C}$ be the BCH code with zeros $\{\alpha^i \mid 0 \leq i \leq 14\} \backslash \{\alpha^5, \alpha^6, \alpha^9\}$. As $\mathcal{C}_0^{\perp}$ is equivalent to the $[15, 12, 3]_4$ code with zeros $\alpha^5, \alpha^6, \alpha^9$, it has minimum distance $d_0 = 3$. Hence, we obtain a $(4, 1, 1)$ PSMC code of cardinality 4^9. □

4.2 Code construction over $\mathbb{F}_{2^\mu}$ for masking up to $2^{\mu-1}(d_0 + 1) - 1$ partially stuck cells

We generalize [29, Section VI] to be able to cope with errors. Unlike [29, Section VI] that could be over any prime power q, the following code construction works over the finite field $\mathbb{F}_q$ where $q = 2^\mu$ in order to describe a 2^μ-ary partially stuck cells code construction. This is because binary sub-field subcodes that are required in this construction are not linear subspace for codes over any prime power q. We denote by $\beta_0 = 1, \beta_1, \ldots, \beta_{\mu-1}$ a basis of $\mathbb{F}_{2^\mu}$ over $\mathbb{F}_2$. That is, any element $a \in \mathbb{F}_{2^\mu}$ can be uniquely represented as $a = \sum_{i=0}^{\mu-1} a_i \beta_i$ where $a_i \in \mathbb{F}_2$ for all i. In particular, $a \in \mathbb{F}_2$ if and only if $a_1 = \cdots = a_{\mu-1} = 0$. This is a crucial property of $\mathbb{F}_{2^\mu}$ that we will use in Construction 3.

Construction 3 *Let $\mu > 1$. Suppose $\boldsymbol{G}$ is a $k \times n$ generator matrix of an $[n, k, d]_{2^\mu}$ code $\mathcal{C}$ of the form*

$$\boldsymbol{G} = \begin{bmatrix} \boldsymbol{H}_0 \\ \boldsymbol{G}_1 \\ \boldsymbol{x} \end{bmatrix} \tag{2}$$

where

1. $\boldsymbol{H}_0 \in \mathbb{F}_2^{l\times n}$ *is a parity check matrix of an* $[n, n-l, d_0]_2$ *code* $\mathcal{C}_0$,
2. $\boldsymbol{G}_1 \in \mathbb{F}_{2^\mu}^{(k-l-1)\times n}$,
3. $\boldsymbol{x} \in \mathbb{F}_{2^\mu}^{1\times n}$ *has Hamming weight* n.

From the code $\mathcal{C}$*, a PSMC can be obtained, whose encoder and decoder are shown in Algorithm 5 and Algorithm 6.*

Algorithm 5: Encoding $(\boldsymbol{m}; \boldsymbol{m}'; \boldsymbol{\phi})$

Input:
- Message:
  $(\boldsymbol{m}', \boldsymbol{m}) \in \mathcal{F}^l \times \mathbb{F}_{2^\mu}^{k-l-1}$, where
  $\mathcal{F} = \{\sum_{i=1}^{\mu-1} x_i\beta_i \mid (x_1, \ldots, x_{\mu-1}) \in \mathbb{F}_2^{\mu-1}\}$.
- Positions of partially stuck-at-1 cells: $\boldsymbol{\phi}$
- Notations introduced in Construction 3.

1 $\boldsymbol{w} \leftarrow \boldsymbol{m}' \cdot \boldsymbol{H}_0 + \boldsymbol{m} \cdot \boldsymbol{G}_1 + z \cdot \boldsymbol{x}$ where $z \in \mathbb{F}_{2^\mu}$ is chosen such that $|\{i \in \boldsymbol{\phi} \mid w_i \in \mathbb{F}_2\}| \le d_0 - 1$.
2 Choose $\boldsymbol{\gamma} \in \mathbb{F}_2^l$ such that $(\boldsymbol{\gamma}\boldsymbol{H}_0)_i = 1 - \boldsymbol{w}_i$ for all $i \in \boldsymbol{\phi}$ for which $\boldsymbol{w}_i \in \mathbb{F}_2$.

Output: $\boldsymbol{c} = \boldsymbol{w} + \boldsymbol{\gamma} \cdot \boldsymbol{H}_0 \in \mathcal{C}$

Algorithm 6: Decoding

Input:
- $\boldsymbol{y} = \boldsymbol{c} + \boldsymbol{e} \in \mathbb{F}_{2^\mu}^n$, where $\boldsymbol{c}$ is a valid output of Algorithm 5 and $\boldsymbol{e}$ is an error of Hamming weight at most t.
- Notations introduced in Construction 3.

1 $\hat{\boldsymbol{c}} \leftarrow$ decode $\boldsymbol{y}$ in the code $\mathcal{C}$
2 Obtain $\boldsymbol{a} \in \mathbb{F}_{2^\mu}^l, \hat{\boldsymbol{m}} \in \mathbb{F}_{2^\mu}^{k-l-1}, \hat{z} \in \mathbb{F}_{2^\mu}$ such that $\hat{\boldsymbol{c}} = \boldsymbol{a}\boldsymbol{H}_0 + \hat{\boldsymbol{m}}\boldsymbol{G}_1 + \hat{z}\boldsymbol{x}$.
3 Obtain $\hat{\boldsymbol{m}'} \in \mathcal{F}^{k-l-1}$ and $\hat{\boldsymbol{\gamma}} \in \mathbb{F}_2^{k-l-1}$ such that $\boldsymbol{a} = \hat{\boldsymbol{m}'} + \hat{\boldsymbol{\gamma}}$.

Output: $(\hat{\boldsymbol{m}}, \hat{\boldsymbol{m}'})$

Theorem 3 *The coding scheme in Construction 3 is a* 2^μ*-ary* $(2^{\mu-1}d_0 - 1, 1, \lfloor\frac{d-1}{2}\rfloor)$ *PSMC of length* n *and cardinality* $2^{\mu(k-l-1)}2^{l(\mu-1)}$.

Proof Let $\boldsymbol{\phi} \subset [n]$ have size $u \le 2^{\mu-1}d_0 - 1$.
We first show the existence of z from Step 1. For each $i \in \boldsymbol{\phi}$, we have that $\boldsymbol{x}_i \neq 0$, so there are exactly two elements $z \in \mathbb{F}_{2^\mu}$ such that $(\boldsymbol{m}' \cdot \boldsymbol{H}_0 + \boldsymbol{m} \cdot \boldsymbol{G}_1)_i + z\boldsymbol{x}_i \in \mathbb{F}_2$. As a result,

$$2u = 2|\boldsymbol{\phi}| = |\{(i, z) \in \boldsymbol{\phi} \times \mathbb{F}_{2^\mu} \mid (\boldsymbol{m}' \cdot \boldsymbol{H}_0 + \boldsymbol{m} \cdot \boldsymbol{G}_1)_i + z\boldsymbol{x}_i \in \mathbb{F}_2\}|.$$

As $u < 2^{\mu-1}d_0$, there is a $z \in \mathbb{F}_{2^\mu}$ such that the condition in Step 1 is satisfied.

As $\boldsymbol{H}_0$ is the parity check matrix of a code with minimum distance d_0, any $d_0 - 1$ columns of $\boldsymbol{H}_0$ are independent, so an appropriate $\boldsymbol{\gamma}$ exists. Now we show that $c_i \neq 0$ for all $i \in \boldsymbol{\phi}$. Indeed, if $\boldsymbol{w}_i \notin \mathbb{F}_2$, then $\boldsymbol{c}_i = \boldsymbol{w}_i + (\boldsymbol{\gamma}\boldsymbol{H}_0)_i \in \{\boldsymbol{w}_i, \boldsymbol{w}_i + 1\}$, so $\boldsymbol{c}_i \notin \mathbb{F}_2$. By Step 2 in Algorithm 5, for $\boldsymbol{w}_i \in \mathbb{F}_2$, we have that $\boldsymbol{c}_i = 1$. Hence, for all $i \in \boldsymbol{\phi}$, $\boldsymbol{c}_i$ is either 1 or is in $\mathbb{F}_{2^\mu} \setminus \mathbb{F}_2$, i.e., $c_i \neq 0$.

Decoding: As $\boldsymbol{c} \in \mathcal{C}$, $\hat{\boldsymbol{c}} = \boldsymbol{c}$. As $\boldsymbol{G}$ has full rank, and

$$\boldsymbol{c} = (\boldsymbol{m}' + \boldsymbol{\gamma})\boldsymbol{H}_0 + \boldsymbol{m}\boldsymbol{G}_1 + z\boldsymbol{x},$$

it holds that $\boldsymbol{a} = \hat{\boldsymbol{m}}' + \hat{\boldsymbol{\gamma}}$, $\hat{\boldsymbol{m}} = \boldsymbol{m}$ and $\hat{z} = z$. As $\hat{\boldsymbol{m}}' \in \mathcal{F}^l$ and $\hat{\boldsymbol{\gamma}} \in \mathbb{F}_2^l$, we can retrieve $\hat{\boldsymbol{m}}' = \boldsymbol{m}'$ from $\boldsymbol{a} = \hat{\boldsymbol{m}}' + \hat{\boldsymbol{\gamma}}$. □

We show next two minor extensions of Theorem 3 for the special case that $\boldsymbol{x}$ is the all-one vector.

Proposition 1 *If $\boldsymbol{x}$ is the all-one vector in Theorem 3, then the coding scheme in Construction 3 can be modified to produce a 2^μ-ary $(2^{\mu-1}d_0 - 1, 1, \lfloor \frac{d-1}{2} \rfloor)$ PSMC of length n and cardinality $2 \times 2^{\mu(k-l-1)}2^{l(\mu-1)}$.*

Proof For $\boldsymbol{x} = \mathbf{1}$, if $\boldsymbol{m}'\boldsymbol{H}_0 + \boldsymbol{m}\boldsymbol{G}_1 + z\mathbf{1}$ has at most $d_0 - 1$ binary entries, then so has $\boldsymbol{m}'\boldsymbol{H}_0 + \boldsymbol{m}\boldsymbol{G}_1 + (z+1)\mathbf{1}$. Hence, there is a $z_0 \in \mathcal{F}$ such that $\boldsymbol{w} + z_0\mathbf{1}$ has at most $d_0 - 1$ binary entries, and we can encode

$$\boldsymbol{w} = \boldsymbol{m}'\boldsymbol{H}_0 + \boldsymbol{m}\boldsymbol{G}_1 + (z_0 + \zeta)\mathbf{1},$$

where $\zeta \in \{0, 1\}$ is an additional message bit so that the cardinality from Theorem 3 is doubled. As $z_0 \in \mathcal{F}$ and $\zeta \in \{0, 1\}$, the pair (z_0, ζ) can be retrieved from $z_0 + \zeta$. □

Construction 3.A (Extension of Construction 3) Let $\boldsymbol{G}$ be a $k \times n$ generator matrix of an $[n, k, d]_{2^\mu}$ code $\mathcal{C}$ of the form

$$\boldsymbol{G} = \begin{bmatrix} \boldsymbol{H}_0 \\ \boldsymbol{G}_1 \\ \mathbf{1} \end{bmatrix}, \text{ where}$$

1) $\mathbf{1}$ is the all-one vector of length n
2) $\boldsymbol{G}_1 \in \mathbb{F}_q^{k-l-1 \times n}$
3) $\begin{bmatrix} \boldsymbol{H}_0 \\ \mathbf{1} \end{bmatrix}$ is the parity-check matrix of an $[n, n-l-1, d_e]_2$ code.

Theorem 4.A *If the conditions of Construction 3.A hold, then Construction 3 can be modified to produce a 2^μ-ary $(2^{\mu-1}d_e, 1, \lfloor \frac{d-1}{2} \rfloor)$ PSMC of length n and cardinality $2^{\mu(k-l-1)}2^{l(\mu-1)}$.*

Proof In Step 1 of Algorithm 5, the encoder determines z such that $|\{i \in \boldsymbol{\phi} \mid w_i \in \mathbb{F}_2\}| \leq d_e - 1$; the existence of such a z is proved as in the proof of Theorem 3. Next, the encoder determines $\boldsymbol{\gamma} \in \{0, 1\}^l$ and $\gamma_0 \in \{0, 1\}$ such that

$$\boldsymbol{v} = (\boldsymbol{\gamma}, \gamma_0) \cdot \begin{bmatrix} \boldsymbol{H}_0 \\ \mathbf{1} \end{bmatrix}$$

is such that $v_i = 1 - w_i$ for all $i \in \boldsymbol{\phi}$ for which $w_i \in \{0, 1\}$. The encoding output $\boldsymbol{c} = \boldsymbol{v} + \boldsymbol{w} = (\boldsymbol{m}' + \boldsymbol{\gamma})\boldsymbol{H}_0 + \boldsymbol{m}\boldsymbol{G}_1 + (z_0 + \gamma_0)\mathbf{1}$ thus is in $\mathcal{C}$ and has no zeros in the positions of $\boldsymbol{\phi}$.

In decoding, from $\boldsymbol{c}$ both $(\boldsymbol{m}' + \boldsymbol{\gamma})$ and $\boldsymbol{m}$ can be retrieved, and so, as $\boldsymbol{m}' \in \mathcal{F}^l$ and $\boldsymbol{\gamma} \in \{0, 1\}^l$, $\boldsymbol{m}'$ can be retrieved as well. □

Proposition 1 doubles the size of the PSMC as compared to Theorem 3 (by using ζ as additional message bit), while masking the same number of partially stuck-at-errors and correcting the same number of substitution errors. Theorem 4.A, as compared to Theorem 3,

results in a PSMC of the same size and error correction capabilities, but increases the number of cells that can be masked from $2^{\mu-1}d_0 - 1$ to $2^{\mu-1}d_e - 1$. If d_0 is odd, then this increment is at least $2^{\mu-1}$.

Now, we show an example using nested BCH codes, allowing to store more symbols compared to Theorem 3 for the same code parameters.

Example 2 Let α be a primitive 15^{th} root of unity in $\mathbb{F}_{16}$, and let $\mathcal{C}$ be the $[15, 12, 3]_4$ BCH code with zeros α^5, α^6 and α^9. Let the $[15, 4]_2$ subcode $\mathcal{C}_0^{\perp}$ of $\mathcal{C}$ be defined as

$$\mathcal{C}_0^{\perp} = \left\{ (x_0, \ldots, x_{14}) \in \mathbb{F}_2^{15} \;\middle|\; \sum_{i=0}^{14} x_i \alpha^{ij} = 0 \text{ for } j \in \{0, \ldots, 14\} \setminus \{7, 11, 13, 14\} \right\}.$$

As $\mathbf{1} \in \mathcal{C} \backslash \mathcal{C}_0^{\perp}$, the code $\mathcal{C}$ has a generator matrix of the form given in Construction 3, namely

$$\boldsymbol{G}' = \begin{bmatrix} \boldsymbol{H}_0 \\ \boldsymbol{G}_1 \\ \boldsymbol{x} \end{bmatrix},$$

where $\boldsymbol{H}_0$ is a generator matrix for $\mathcal{C}_0^{\perp}$ and $\boldsymbol{G}_1$ has $12-4-1 = 7$ rows. The code $\mathcal{C}_0 = (\mathcal{C}_0^{\perp})^{\perp}$ is equivalent to the $[15, 11]_2$

We stipulate that $\alpha^4 = \alpha + 1$ to obtain explicit $\boldsymbol{G}'$ as below,

$$\boldsymbol{G}' = \begin{pmatrix}
1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 \\
\omega & \omega & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \omega & \omega & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \omega & \omega & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & \omega & \omega & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & \omega & \omega & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & \omega & \omega & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \omega & \omega & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1
\end{pmatrix},$$

where $\mathbb{F}_4$ has elements $\{0, 1, \omega, \omega^2\}$ with $\omega = \alpha^5$. Note that the top row of $\boldsymbol{H}_0$ corresponds to the generator polynomial for $\mathcal{C}_0^{\perp}$, and the top row of $\boldsymbol{G}_1$ corresponds to the coefficients of $(x+\alpha^5)(x+\alpha^6)(x+\alpha^9)$ which is the generator polynomial of $\mathcal{C}$. Application of Proposition 1 yields a (5, 1, 1) PSMC over $\mathbb{F}_{2^2}$ of length 15 and size $2 \times 4^7 2^4 = 2^{19}$, whereas application of Construction 3.A gives a (7, 1, 1) PSMC over $\mathbb{F}_{2^2}$ with cardinality $2^{2(7+4)-4} = 2^{18}$. Note that application of Construction 3 yields a (5, 1, 1) PSMC over $\mathbb{F}_{2^2}$ of length 15 and size $4^7 2^4 = 2^{18}$.

Finally, we note that application of Theorem 2 to $\mathcal{C}$ yields a (4, 1, 1) PSMC of size 4^8, which has worse parameters than the three PSMC mentioned before. □

Example 2 clearly shows that for the same code parameters, Construction 3, Proposition 1 and Construction 3.A significantly improve upon Construction 2.

Remark 4 For masking only, choose $n - k = 0$ in Construction 3 and therefore,

$$\boldsymbol{G}_1 = \left[\mathbf{0}_{(n-l-1)\times(l+1)} \; \boldsymbol{I}_{(n-l-1)} \; \mathbf{0}_{(n-l-1)\times 1} \right],$$

and we can store $n - l - 1$ information symbols. Thus, Proposition 1 for masking only improves upon [29, Construction 5]. For example if $l = 4$, then $n - l - 1 = 10$ in [29, Example 7] and the size of the code is $2^{2(n-l-1)} \cdot 2^l = 2^{24}$, while $n - l - 1 = 10$ in Proposition 1 for $\boldsymbol{x} = \mathbf{1}$ and the cardinality is $2 \cdot 2^{2n-l-1} = 2^{25}$.

We summarize in Table 3 our constructions and compare them with some of the previous works, namely with the construction for masking classical stuck cells in [12] and constructions for partially stuck cells without errors in [29]. Table 3 clearly shows that more information can be stored with partially stuck-at errors than with classical stuck cells.

5 Generalization of the constructions to arbitrary partially defective levels

So far, we have considered the *important* case for $s_i = 1$ for all $i \in \boldsymbol{\phi}$. In this section, we present error correction and masking codes constructions that can mask partially stuck cells at any level $\boldsymbol{s} \in \mathbb{F}_q^n$ of weight $\mathrm{wt}(\boldsymbol{s}) \leq |\boldsymbol{\phi}|$ and correct errors additionally.

We fix throughout the paper a total ordering "$\geq$" of the elements of $\mathbb{F}_q$ such that $a \geq 1 \geq 0$ for all $a \in \mathbb{F}_q \backslash \{0\}$. So 0 is the smallest element in $\mathbb{F}_q$, and 1 is the next smallest element in $\mathbb{F}_q$. We extend the ordering on $\mathbb{F}_q$ to $\mathbb{F}_q^n$: for $\boldsymbol{x} = (x_0, \ldots, x_{n-1}) \in \mathbb{F}_q^n$ and $\boldsymbol{y} = (y_0, \ldots, y_{n-1}) \in \mathbb{F}_q^n$, we say that $\boldsymbol{x} \geq \boldsymbol{y}$ if and only if $x_i \geq y_i$ for all $i \in [n]$.

5.1 Generalization of Theorem 1

Here, we give only the main theorem without adding the exact encoding and decoding processes because it follows directly as a consequence of Construction 1.

Theorem 4 (Generalization of Theorem 1) *Let* $\Sigma = \{\boldsymbol{s} \in \mathbb{F}_q^n \mid \sum_{i=0}^{n-1} s_i \leq q - 1\}$. *Assume there is an* $[n, k, d]_q$ *code* $\mathcal{C}$ *of a generator matrix as specified in Theorem 1. Then there exists a* $(\Sigma, \lfloor \frac{d-1}{2} \rfloor)$ *PSMC over* $\mathbb{F}_q$ *of length n and cardinality* q^{k-1}.

Proof We follow the generalization for the masking partially-stuck-at any arbitrary levels in [29, Theorem 10]. Hence, for $\boldsymbol{s} \in \Sigma$, we modify Step 2 in Algorithm 1 such that $w_i - v \geq s_i$ for all $i \in [n]$. Such a v exists as each cell partially-stuck-at level s_i excludes s_i values for v, and $\sum_{i=0}^{n-1} s_i < q$. The rest of the encoding steps and the decoding process are analogous to Algorithms 1 and 2. As the output from the encoding process is a codeword, we can correct $\lfloor \frac{d-1}{2} \rfloor$ errors. □

5.2 Generalization of construction 2

In the following, we generalize Construction 2 to arbitrary $\boldsymbol{s}$ stuck levels.

Proposition 2 *Let*

$$\Sigma = \left\{ \boldsymbol{s} \in \mathbb{F}_q^n \;\middle|\; \min \left\{ \sum_{i \in \Psi} s_i \;\middle|\; \Psi \subseteq [n], |\Psi| = n - d_0 + 2 \right\} \leq q - 1 \right\}$$

then the coding scheme in Construction 2 can be modified to produce a $(\Sigma, \lfloor \frac{d-1}{2} \rfloor)$ *PSMC of length n and size* q^{k-l}.

Table 3 Comparison among [12], [29], and this work. We denote by d the minimum distance required to correct errors and d_0 to mask (partially) stuck cells. A positive integer $\mu > 1$ is defined in Construction 3. Other Notation: See Sect. 2.1

	(Partially) Stuck Cells u	Distance d_0	Errors $\lfloor \frac{d-1}{2} \rfloor$	Redundancy	Cardinality
Construction 1	$\leq q-1$	irrelevant	Yes	$n-k+1$	q^{k-1}
Construction 2	$\leq n$	$\geq u-q+3$	Yes	$n-k+l$	q^{k-l}
Construction 3	$\leq n$	$\geq \lfloor \frac{2u}{2^\mu} \rfloor + 1$	Yes	$n-k+1+\frac{l}{\mu}$	$q^{k-1-\frac{l}{\mu}}$
Proposition 1	$\leq n$	$\geq \lfloor \frac{2u}{2^\mu} \rfloor + 1$	Yes	$n-k+1+\frac{l-1}{\mu}$	$q^{k-1+\frac{1-l}{\mu}}$
Construction 3.A	$\leq n$	$\geq \lfloor \frac{2u}{2^\mu} \rfloor$ if d_0 is odd	Yes	$n-k+1+\frac{l}{\mu}$	$2^{2(k-1-\frac{l}{\mu})}$
[29, Construction 2]	$\leq q-1$	irrelevant	No	1 (since $n-k=0$)	q^{n-1}
[29, Construction 4]	$\leq n$	$\geq u-q+3$	No	l (since $n-k=0$)	q^{n-l}
[29, Construction 5]	$\leq n$	$\geq \lfloor \frac{2u}{q} \rfloor + 1$	No	$1+l(1-\log_q \lfloor \frac{q}{2} \rfloor)$ (since $n-k=0$), and $1+\frac{l}{\mu}$ (for $q=2^\mu$)	$q^{n-1-l(1-\log_q \lfloor \frac{q}{2} \rfloor)}$, and $2^{2(n-1-\frac{l}{\mu})}$ (for $q=2^\mu$)
Proposition 1 (masking only)	$\leq n$	$\geq \lfloor \frac{2u}{2^\mu} \rfloor + 1$	No	$1+\frac{l-1}{\mu}$ (since $n-k=0$)	$2^{2(n-1-\frac{l-1}{\mu})}$
[12, Theorem 1]	$\leq n$	$\geq u+1$	Yes	$n-k+l$	q^{k-l}

Proof To avoid cumbersome notation, we assume without loss of generality that $s_0 \geq s_1 \geq \cdots \geq s_{n-1}$. As the $d_0 - 2$ leftmost columns of $\boldsymbol{H}_0$ are independent, there is an invertible $\boldsymbol{T} \in \mathbb{F}_q^{l \times l}$ such that the matrix $\boldsymbol{Y} = \boldsymbol{T}\boldsymbol{H}_0$ has the form

$$\boldsymbol{Y} = \begin{bmatrix} I_{d_0-2} & \boldsymbol{A} \\ \boldsymbol{0} & \boldsymbol{B} \end{bmatrix},$$

where I_{d_0-2} is the identity matrix of size $d_0 - 2$, $\boldsymbol{0}$ denotes the $(l - d_0 + 2) \times (d_0 - 2)$ all-zero matrix, $\boldsymbol{A} \in \mathbb{F}_q^{(d_0-2)\times(n-d_0+2)}$ and $\boldsymbol{B} \in \mathbb{F}_q^{(l-d_0+2)\times(n-d_0+2)}$. As $\boldsymbol{T}$ is invertible, and any $d_0 - 1$ columns of $\boldsymbol{H}_0$ are independent, any $d_0 - 1$ columns of $\boldsymbol{Y}$ are independent as well.

For $0 \leq i \leq l - 1$, we define

$$L_i = \{j \in [n] \mid Y_{i,j} \neq 0 \text{ and } Y_{m,j} = 0 \text{ for } m > i\}. \tag{3}$$

Clearly, $L_0, \ldots, L_{l-1}$ are pairwise disjoint. Moreover, for each $j \in \{d_0-2, d_0-1, \ldots, n-1\}$, column j of $\boldsymbol{Y}$ is independent from the $(d_0 - 2)$ leftmost columns of $\boldsymbol{Y}$, and so there is an $i \geq d_0 - 2$ such that $Y_{i,j} \neq 0$. Consequently,

$$\bigcup_{i=d_0-2}^{l-1} L_i = \{d_0 - 2, \ldots, n - 1\}. \tag{4}$$

By combining (4) and the form of $\boldsymbol{Y}$, we infer that

$$L_k = \{k\} \text{ for all } k \in [d_0 - 2]. \tag{5}$$

Let $\boldsymbol{w} \in \mathbb{F}_q^n$ be the vector to be masked, i.e. the vector after Step 1 in Algorithm 3. The encoder successively determines the coefficients $z_0, \ldots, z_{l-1}$ of $\boldsymbol{z} \in \mathbb{F}_q^l$ such that $\boldsymbol{w} + \boldsymbol{z}\boldsymbol{Y} \geq \boldsymbol{s}$, as follows.

For $j \in [d_0 - 2]$, the encoder sets $z_j = s_j - w_j$.

Now let $d_0 - 2 \leq i \leq l - 1$ and assume that $z_0, \ldots, z_{i-1}$ have been obtained such that

$$w_j + \sum_{k=0}^{i-1} z_k Y_{k,j} \geq s_j \text{ for all } j \in \bigcup_{k=0}^{i-1} L_k. \tag{6}$$

It follows from combination of (5) and the choice of $z_0, \ldots, z_{d_0-3}$ that (6) is satisfied for $i = d_0 - 2$.

For each $j \in L_i$, we define F_j as

$$F_j = \left\{ x \in \mathbb{F}_q \;\middle|\; w_j + \sum_{k=0}^{i-1} z_k Y_{k,j} + x Y_{i,j} < s_j \right\}.$$

Clearly, $|F_j| = s_j$ as $Y_{i,j} \neq 0$, and so

$$\left| \bigcup_{j \in L_i} F_j \right| \leq \sum_{j \in L_i} |F_j| = \sum_{j \in L_i} s_j \leq \sum_{j=d_0-2}^{n-1} s_j \leq q - 1,$$

where the last inequality follows from the assumption of Σ in the proposition statement and the ordering of the components of $\boldsymbol{s}$. Hence, $\bigcup_{j \in L_i} F_j \neq \mathbb{F}_q$. The encoder chooses

$z_i \in \mathbb{F}_q \setminus \bigcup_{j \in L_i} F_j$. We claim that

$$w_j + \sum_{k=0}^{i} z_k Y_{k,j} \geq s_j \text{ for all } j \in \bigcup_{k=0}^{i} L_k. \tag{7}$$

For $j \in L_i$, (7) follows from the definition of F_j. For $j \in \bigcup_{k=0}^{i-1} L_k$, (7) follows from (6) and the fact that $Y_{i,j} = 0$.

By using induction on i, we infer that

$$w_j + \sum_{k=0}^{l-1} z_k Y_{k,j} \geq s_j \text{ for all } j \in \cup_{k=0}^{l-1} L_k = [n]. \tag{8}$$

That is, with $\boldsymbol{z} = (z_0, \ldots, z_{l-1})$, we have that $\boldsymbol{w} + \boldsymbol{z}\boldsymbol{Y} \geq \boldsymbol{s}$. As $\boldsymbol{Y} = \boldsymbol{T}\boldsymbol{H}_0$, it follows that $\boldsymbol{z} := \boldsymbol{z}\boldsymbol{T}$ is such that

$$\boldsymbol{w} + \boldsymbol{z}\boldsymbol{H}_0 \geq \boldsymbol{s}.$$

The decoding process remains as in Algorithm 4. □

We give an alternative *non-constructive* proof for Proposition 2 in Appendix A.

Remark 5 The proof of Proposition 2 shows that $(d_0 - 2)$ cells can be set to any desired value, while the remaining $(n - d_0 + 2)$ cells can be made to satisfy the partial stuck-at conditions, provided that the sum of the stuck-at levels in these $(n - d_0 + 2)$ cells is less than q.

Example 3 Let $\mathcal{C}$ be the $[13, 10, 3]_3$ generated by

$$\boldsymbol{G} = \begin{bmatrix} \boldsymbol{G}_1 \\ \boldsymbol{H}_0 \end{bmatrix} = \begin{pmatrix} 1&0&0&0&0&0&0&0&0&0&1&2&0 \\ 0&1&0&0&0&0&0&0&0&0&0&1&2 \\ 0&0&1&0&0&0&0&0&0&0&1&0&2 \\ 0&0&0&1&0&0&0&0&0&0&1&1&1 \\ 0&0&0&0&1&0&0&0&0&0&1&1&2 \\ 0&0&0&0&0&1&0&0&0&0&2&0&2 \\ 0&0&0&0&0&0&1&0&0&0&1&2&1 \\ 1&0&1&1&0&1&0&1&1&1&0&1&1 \\ 0&1&1&2&0&0&1&1&2&0&1&1&2 \\ 0&0&0&0&1&1&1&1&1&2&2&2&2 \end{pmatrix} \in \mathbb{F}_3^{10\times 13}$$

Let $\mathcal{C}_0$ be the code with parity check matrix $\boldsymbol{H}_0$. Let

$$\boldsymbol{s} = \begin{pmatrix} 2&0&0&0&0&1&1&0&0&0&0&0&0 \end{pmatrix} \in \mathbb{F}_3^{13}.$$

Notations are as introduced in Construction 2 and Proposition 2.

Observe that $\mathcal{C}_0$ has minimum distance $d_0 = 3$. Note that $\boldsymbol{H}_0$ has the form

$$\boldsymbol{H}_0 = \begin{bmatrix} I_{d_0-2} & \boldsymbol{A} \\ \boldsymbol{0} & \boldsymbol{B} \end{bmatrix} = \begin{pmatrix} 1&0&1&1&0&1&0&1&1&1&0&1&1 \\ 0&1&1&2&0&0&1&1&2&0&1&1&2 \\ 0&0&0&0&1&1&1&1&1&2&2&2&2 \end{pmatrix},$$

in Proposition 2 we can take $\boldsymbol{T}$ the identity matrix, and $\boldsymbol{Y} = \boldsymbol{H}_0$.

For a random message vector $\boldsymbol{m} = \begin{pmatrix} 1&1&1&2&1&1&0 \end{pmatrix} \in \mathbb{F}_3^7$,

$$\boldsymbol{w} = \boldsymbol{m}\boldsymbol{G}_1 = \begin{pmatrix} 1&1&1&2&1&1&0&0&0&0&1&0&2 \end{pmatrix} \in \mathbb{F}_3^{13}.$$

To mask the $d_0 - 2$ largest positions of the vector $\boldsymbol{s}$ (corresponding to the $(d_0 - 2)$ leftmost columns of $\boldsymbol{Y}$), namely s_0 (the leftmost value highlighted in blue in $\boldsymbol{s}$), the encoder sets $z_j = s_j - w_j$ for $j \in [d_0 - 2]$. Thus, $z_0 = 2 - 1 = 1$. Note that

$$\boldsymbol{w} + \begin{pmatrix} z_0 & 0 & 0 \end{pmatrix} \boldsymbol{H}_0 = \begin{pmatrix} 2 & 1 & 2 & 0 & 1 & 2 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \end{pmatrix}.$$

Next, the encoder determines z_1 and z_2 so as to mask in positions 5 and 6. As $\boldsymbol{H}_0$ has non-zero entries in the bottom row of columns 5 and 6, we have that $L_1 = \emptyset$ and $L_2 = \{5, 6\}$. The encoder can thus take any value for z_1, so let us say it takes 0. The coefficient z_2 is chosen in such a way that

$$\boldsymbol{w} + \begin{pmatrix} z_0 & 0 & 0 \end{pmatrix} \boldsymbol{H}_0 + \begin{pmatrix} 0 & 0 & z_2 \end{pmatrix} \boldsymbol{H}_0 \geq \boldsymbol{s}.$$

This inequality is satisfied if and only if $2 + z_2 \neq 0$ (position 5) and $0 + z_2 \neq 0$ (position 6), so if and only if $z_2 = 2$.

Note that, in this example, there are $\mathrm{wt}(\boldsymbol{s}) = 3$ partially stuck cells. As $\mathcal{C}$ has minimum distance 3, it can correct a single error. □

Corollary 2 (Generalization of Theorem 2) *Let $s \in \mathbb{F}_q$ and let*

$$\Sigma = \left\{ \boldsymbol{s} \in \mathbb{F}_q^n \mid \mathrm{wt}(\boldsymbol{s}) \leq d_0 + \left\lceil \frac{q}{s} \right\rceil - 3 \text{ and } \max\{s_i \mid i \in [n] \leq s\} \right\}.$$

The coding scheme in Construction 2 is a $(\Sigma, \lfloor \frac{d-1}{2} \rfloor)$ PSMC scheme of length n and size q^{k-l}.

Proof Let $\boldsymbol{s} \in \Sigma$ have weight $u \leq d_0 + \lceil \frac{q}{s} \rceil - 3$. Let $\Psi \subseteq [n]$ of size $n - d_0 + 2$ be such that the number of non-zero components of $\boldsymbol{s}$ in $[n] \setminus \Psi$ equals $\min(d_0 - 2, u)$. Then $\boldsymbol{s}$ has $u - \min(d_0 - 2, u)$ non-zero components in Ψ. As a consequence, if $u \leq d_0 - 2$, then $\sum_{i \in \Psi} s_i = 0$, and if $u > d_0 - 2$, then

$$\sum_{i \in \Psi} s_i \leq s(u - d_0 + 2) \leq s(\lceil \frac{q}{s} \rceil - 1) < s(\frac{q}{s} + 1 - 1) = q.$$

Hence in both cases, $\sum_{i \in \Psi} s_i \leq q - 1$. The corollary thus follows from Proposition 2. In particular, if $s = 1$, the corollary agrees with Theorem 2. □

We do not generalize Construction 3 as it is tailored to the *special* case where $s_i = 1$ for all $i \in \boldsymbol{\phi}$.

6 Trading partially stuck cells with errors

In the constructions shown so far, the encoder output $\boldsymbol{c}$ is a word from an error correcting code $\mathcal{C}$. If $\boldsymbol{c}$ does not satisfy the partially-stuck-at conditions in j positions, the encoder could modify it in these j positions to obtain a word $\boldsymbol{c}' = \boldsymbol{c} + \boldsymbol{e}'$ satisfying the partially-stuck-at constrains, while $\mathrm{wt}(\boldsymbol{e}') = j$. If $\mathcal{C}$ can correct t errors, then it still is possible to correct $t - j$ errors in $\boldsymbol{c}'$. This observation was also made in [12, Theorem 1]. The above reasoning shows that the following proposition holds.

Proposition 3 *If there is an $(n, M)_q(u, 1, t)$ PSMC, then for any j with $0 \leq j \leq t$, there is an $(n, M)_q(u + j, 1, t - j)$ PSMC.*

In the remainder of this section, we generalize the above proposition to general Σ (Theorem 5). We also provide variations on the idea of the encoder introducing errors to the result of a first encoding step in order that the final encoder output satisfies the partially-stuck-at conditions.

Theorem 5 (Partial Masking PSMC) *Let $\Sigma \subset \mathbb{F}_q^n$, and assume that there exists an $(n, M)_q(\Sigma, t)$ PSMC $\mathcal{C}$. For any $j \in [t]$, there exists an $(n, M)_q(\Sigma^{(j)}, t - j)$ PSMC $\mathcal{C}_j$, where*

$$\Sigma^{(j)} = \left\{ \boldsymbol{s}' \in \mathbb{F}_q^n \mid \exists \boldsymbol{s} \in \Sigma \left[d(\boldsymbol{s}, \boldsymbol{s}') \leq j \text{ and } \boldsymbol{s}' \geq \boldsymbol{s} \right] \right\}.$$

Algorithm 7: Encoding

Input: $(\boldsymbol{m}, \boldsymbol{s}') \in \mathcal{M} \times \Sigma^{(j)}$.
1 Determine $\boldsymbol{s} \in \Sigma$ such that $d(\boldsymbol{s}, \boldsymbol{s}') \leq j$ and $\boldsymbol{s}' \geq \boldsymbol{s}$.
2 Let $\boldsymbol{c} = \mathcal{E}(\boldsymbol{m}, \boldsymbol{s})$.
3 Define $\boldsymbol{c}' = \mathcal{E}'_j(\boldsymbol{m}, \boldsymbol{s}')$ as $c'_i = \max(c_i, s'_i)$ for $i \in [n]$.
Output: Codeword $\boldsymbol{c}'$.

Algorithm 8: Decoding

Input: Received $\boldsymbol{y} = \boldsymbol{c}' + \boldsymbol{e}$ where $\mathrm{wt}(\boldsymbol{e}) \leq t - j$ and $\boldsymbol{y} \geq \boldsymbol{s}'$
1 Message $\boldsymbol{m} = \mathcal{D}(\boldsymbol{y})$
Output: Message vector $\boldsymbol{m}$

Proof Let the encoder $\mathcal{E}_j$ and the decoder $\mathcal{D}_j$ for $\mathcal{C}_j$ be Algorithm 7 and Algorithm 8, respectively. By definition, $\boldsymbol{c}' \geq \boldsymbol{s}'$. Moreover, if $s_i = s'_i$, then $c_i \geq s_i = s'_i$, so $c_i = c'_i$. As a result, $d(\boldsymbol{c}, \boldsymbol{c}') \leq j$.

In Algorithm 8, the decoder $\mathcal{D}$ of $\mathcal{C}$ is directly used for decoding $\mathcal{C}_j$. As $\boldsymbol{y} \geq \boldsymbol{s}'$, surely $\boldsymbol{y} \geq \boldsymbol{s}$. Moreover, we can write $\boldsymbol{y} = \boldsymbol{c} + (\boldsymbol{c}' - \boldsymbol{c} + \boldsymbol{e})$. As shown above, $\mathrm{wt}(\boldsymbol{c}' - \boldsymbol{c}) \leq j$, and so $\mathrm{wt}(\boldsymbol{c} - \boldsymbol{c}' + \boldsymbol{e}) \leq t$. As a consequence, $\mathcal{D}(\boldsymbol{y}) = \boldsymbol{m}$. □

We can improve on Theorem 5 for Construction 3 giving Lemma 1.

Lemma 1 *Given an $[n, k, d]_q$ code as defined in Construction 3, then for any j such that $0 \leq j \leq \lfloor \frac{d-1}{2} \rfloor$, there is a 2^μ-ary $(2^{\mu-1}(d_0 + j) - 1, 1, \lfloor \frac{d-1}{2} \rfloor - j)$ PSMC of length n and size q^{k-l-1}.*

Proof Let $\boldsymbol{\phi} \subset [n]$ has size $u \leq 2^{\mu-1}(d_0 + j) - 1$. We use the notation from Algorithm 5. After Step 1, $\boldsymbol{w}$ has at most $u_0 = \lfloor \frac{2u}{2^\mu} \rfloor \leq d_0 + j - 1$ binary entries in the positions from $\boldsymbol{\phi}$. After Step 2, at least $d_0 - 1$ of these entries in $\boldsymbol{c}$ differ from 0. By setting the at most j other binary entries in the positions from $\boldsymbol{\phi}$ equal to 1, the encoder introduces at most j errors, and guarantees that the partially-stuck-at conditions are satisfied. □

In Lemma 2, we use another approach for introducing errors in order to satisfy the stuck-at conditions.

Lemma 2 *Given an $[n, k, d]_q$ code containing a word of weight n, for any j with $0 \leq j \leq \lfloor \frac{d-1}{2} \rfloor$, there is a q-ary $(q - 1 + qj, 1, \lfloor \frac{d-1}{2} \rfloor - j)$ PSMC of length n and size q^{k-1}.*

Proof We use the notation from Construction 1.

Let $\boldsymbol{\phi} \subset [n]$ have size $u \leq q - 1 + qj$. Let $\boldsymbol{x}$ be a codeword of weight n. For each $i \in \boldsymbol{\phi}$, there is exactly one $v \in \mathbb{F}_q$ such that $w_i + vx_i = 0$, and so

$$\sum_{v \in \mathbb{F}_q} | \{i \in \boldsymbol{\phi} \mid w_i + vx_i = 0\} | = u.$$

As a consequence, there is $v \in \mathbb{F}_q$ such that $\boldsymbol{c} = \boldsymbol{w} + v\boldsymbol{x}$ has at most $\lfloor \frac{u}{q} \rfloor \leq j$ entries in $\boldsymbol{\phi}$ equal to zero. By setting these entries of $\boldsymbol{c}$ to a non-zero value, the encoder introduces at most j errors. As $\mathcal{C}$ can correct up to $\lfloor \frac{d-1}{2} \rfloor$ errors, it can correct these j errors and additionally up to $\lfloor \frac{d-1}{2} \rfloor - j$ substitution errors. □

Example 4 Consider a $[15, 9, 5]_4$ code $\mathcal{C}$ containing the all-one word, e.g. the BCH code with zeroes $\alpha, \alpha^2, \alpha^3$, where α is a primitive element in $\mathbb{F}_{16}$. Let $u \leq 7$ and $t = 1$. We use the all-one word for partial masking, ensuring that 0 occurs in at most $\lfloor \frac{u}{4} \rfloor \leq 1$ position indexed by $\boldsymbol{\phi}$. We set the codeword value in this position to 1, introducing one error. We can correct this introduced error and one additional random error as $\mathcal{C}$ has minimum distance 5. Hence, we have obtained a 4-ary (7, 1, 1) PSMC of length 15 and cardinality 4^8. □

We show in Example 5 how applying Lemma 2 for Construction 3 outperforms Lemma 1.

Example 5 Given $d_0 = 3$, $u = 15$ and $q = 2^2$ and let α be a primitive element in $\mathbb{F}_4$ and take $\boldsymbol{x} = \mathbf{1}$. Assume we have

$$\begin{aligned} \boldsymbol{w}^{(\boldsymbol{\phi})} &= (\boldsymbol{m}' \cdot \boldsymbol{H}_0 + \boldsymbol{m} \cdot \boldsymbol{G}_1) + z \cdot \mathbf{1} \\ &= (0, 1, \alpha, 1 + \alpha, 0, 1, \alpha, 1 + \alpha, 0, 1, \alpha, 1 + \alpha, 0, 1, \alpha) \\ &\quad + z \cdot \mathbf{1}, \end{aligned}$$

then choosing $z = 1 + \alpha$ minimizes the number of binary values in $\boldsymbol{w}^{(\boldsymbol{\phi})}$, we get:

$$\boldsymbol{w}^{(\boldsymbol{\phi})} = (1 + \alpha, \alpha, 1, 0, 1 + \alpha, \alpha, 1, 0, 1 + \alpha, \alpha, 1, 0, 1 + \alpha, \alpha, 1).$$

Following Step 2 in Algorithm 5 and since $d_0 = 3$, we can mask at most $d_0 - 1$ binary values highlighted in the vector $\boldsymbol{w}^{(\boldsymbol{\phi})}$ that leaves us, in this example, with at most $\lfloor \frac{2u}{2^2} \rfloor - d_0 + 1 = 5$ zeros that remain unmasked.

However, applying Lemma 2 instead for Construction 3 gives a better result. Choosing $\boldsymbol{\gamma} = \mathbf{0}$ in Step 2 of Algorithm 5, we obtain $\boldsymbol{c}^{(\boldsymbol{\phi})} = \boldsymbol{w}^{(\boldsymbol{\phi})}$ with ($\lfloor \frac{u}{q} \rfloor = 3$) zeros highlighted in blue above that we can directly trade. □

Remark 6 As the code from Construction 3 has a word of weight n, Lemma 2 implies the existence of an $(u, 1, t)$ PSMC of cardinality q^{k-1} under the condition that $2(t + \lfloor \frac{u}{q} \rfloor) < d$. Lemma 1 shows the existence of an $(u, 1, t)$ PSMC of smaller cardinality, *viz.* q^{k-l}, under the condition that $2(t + \max(0, \lfloor \frac{2u}{2^\mu} \rfloor - d_0 + 1) < d$. As a consequence, Lemma 1 can only improve on Lemma 2 if $d_0 - 1 > \lfloor \frac{2u}{2^\mu} \rfloor - \lfloor \frac{u}{2^\mu} \rfloor$.

We can generalize Lemma 2 as follows.

Lemma 3 *Given an $[n, k, d]_q$ code containing a word of weight n. Let $0 \leq j \leq \lfloor \frac{d-1}{2} \rfloor$, and let*

$$\Sigma = \left\{ \boldsymbol{s} \in \mathbb{F}_q^n \;\middle|\; \sum_i s_i \leq q - 1 + qj \right\}.$$

There is a q-ary $(\Sigma, \lfloor \frac{d-1}{2} \rfloor - j)$ PSMC of length n and size q^{k-1}.

Proof We use the notation from Theorem 4. For simplicity, we assume that the code contains the all-one word. We wish to choose the multiplier $v \in \mathbb{F}_q$ such that $\boldsymbol{c} = \boldsymbol{w} - v \cdot \mathbf{1}$ satisfies $c_i \geq s_i$ for as many indices i as possible. For each index i, there are $q - s_i$ values for v such that this inequality is met. Hence, there is a $v \in \mathbb{F}_q$ such that $c_i \geq s_i$ for at least $\lceil \frac{1}{q} \sum_i (q - s_i) \rceil = n - \lfloor \frac{1}{q} \sum_i s_i \rfloor$ indices i. The encoder thus needs to introduce errors only in the at most $\lfloor \frac{1}{q} \sum_i s_i \rfloor$ positions for which the inequality is not satisfied. □

Lemma 4 *Assume there exists a matrix as in Proposition 2. Let* $0 \leq j \leq \lfloor \frac{d-1}{2} \rfloor$*, and let*

$$\Sigma = \left\{ \boldsymbol{s} \in \mathbb{F}_q^n \;\middle|\; \exists \Psi \subset [n] : \; |\, \Psi \,| = n - d_0 + 2 \left[\sum_{i \in \Psi} s_i \leq q - 1 + qj \right] \right\}.$$

Then exists a q*-ary* $(\Sigma, \lfloor \frac{d-1}{2} \rfloor - j)$ *PSMC of length* n *and size* q^{k-l}*.*

Proof Let $\boldsymbol{s} \in \Sigma$. In order to simplify notation, we assume without loss of generality that $\sum_{i=d_0-2}^{n-1} s_i \leq q-1+qj$. We use the same argument as in the alternative proof of Proposition 2 (see Appendix A). Clearly,

$$n - d_0 + 2 - \left\lfloor \frac{1}{q} \sum_{i=d_0-2}^{n-1} s_i \right\rfloor \geq n - d_0 + 2 - j.$$

So we infer that for at least $n - j$ indices $i \in [n]$,

$$w_i + \Big((z, \boldsymbol{\eta}) \boldsymbol{T} \boldsymbol{H}_0 \Big)_i \geq s_i.$$

□

Remark 7 The proof of Lemma 4 shows that the encoder output in fact can be made equal to $\boldsymbol{s}$ in the $d_0 - 2$ largest entries of $\boldsymbol{s}$. In fact, it shows that the scheme allows for masking $d_0 - 2$ stuck-at errors, masking partial stuck errors in the remaining cells, and correcting $\lfloor \frac{d-1}{2} \rfloor - j$ substitution errors, provided that the sum of the partially stuck-at levels in the $n - d_0 + 2$ remaining cells is less than $(j + 1)q$.

7 Upper bounds on PSMC codes

The output of an encoder has restrictions on the values in the partially-stuck-at cells; in the other cells, it can attain all values. So the set of all encoder outputs is a poly-alphabetic code [23]. To be more precise, the following proposition holds.

Proposition 4 *Let* $\mathcal{C}$ *be an* $(n, M)_q(\Sigma, t)$ *partially-stuck-at-masking code with encoder* $\mathcal{E}$*. For any* $\boldsymbol{s} \in \Sigma$*, let*

$$\mathcal{C}_{\boldsymbol{s}} = \{ \mathcal{E}(\boldsymbol{m}, \boldsymbol{s}) \mid \boldsymbol{m} \in \mathcal{M} \}.$$

Then $\mathcal{C}_{\boldsymbol{s}}$ *is a code with minimum distance at least* $2t + 1$ *and* $|\mathcal{M}|$ *words, and*

$$\mathcal{C}_{\boldsymbol{s}} \subset Q_0 \times Q_1 \times \cdots \times Q_{n-1}, \text{ where}$$

$Q_i = \{x \in \mathbb{F}_q \mid x \geq s_i\}$.

Proof By our error model, errors in stuck-at cells result in values still satisfying the stuck-at constraints. Therefore, t errors can be corrected if and only if $\mathcal{C}_{\boldsymbol{s}}$ has minimum Hamming distance at least $2t + 1$. The rest of the proposition is obvious. □

As a result of Proposition 4, upper bounds on the size of poly-alphabetic codes [23] are also upper bounds on the size of partially-stuck-at codes. Hence, we give the following corollaries to state Singleton-type and sphere-packing-type bounds for error-correcting and partially-stuck-at-masking codes.

Corollary 3 (Singleton-type bound) *Let $\mathcal{C}$ be a q-ary (Σ, t) PSMC of length n and size M. Then for any $\boldsymbol{s} = (s_0, \ldots, s_{n-1}) \in \Sigma$,*

$$M \leq \min \left\{ \prod_{j \in J} (q - s_j) \;\middle|\; J \subset [n], |J| = n - 2t \right\}.$$

Proof Combination of Proposition 4 and [23, Theorem 2]. □

Corollary 4 (Sphere-packing-type bound) *Let $\mathcal{C}$ be a q-ary (Σ, t) PSMC of length n and size M. Then for any $\boldsymbol{s} = (s_0, \ldots, s_{n-1}) \in \Sigma$*

$$M \leq \frac{\prod_{i=0}^{n-1} (q - s_i)}{V_t^{(b)}},$$

where $V_t^{(b)}$, the volume of a ball of radius t, satisfies

$$V_t^{(b)} = \sum_{r=0}^{t} V_r^{(s)},$$

where the volume $V_r^{(s)}$ of the sphere with radius r is given by

$$\begin{aligned} V_0^{(s)} &= 1, \\ V_r^{(s)} &= \sum_{1 \leq i_1 < \ldots < i_r \leq n} (q - 1 - s_{i_1}) \cdots (q - 1 - s_{i_r}). \end{aligned}$$

Proof Combination of Proposition 4 and [23, Theorem 3]. □

Remark 8 The difference between poly-alphabetic codes and partially-stuck-at-masking codes is that in the former, the positions of stuck-at cells and the corresponding levels are known to both encoder and decoder, whereas in the latter, this information is known to the encoder only.

Figure 2 compares our derived sphere-packing-like bound to the amount of storable information symbols for a completely reliable memory (i.e., no stuck cells, no errors that can be seen at $u = 0$ in the solid line) and the upper bound on the cardinality of an only-masking PSMC (only stuck cells, no errors) derived in [29] as depicted in the solid curve. At $u = 0$, the derived sphere-packing-type bound (dotted and dashed-dotted plots) matches the usual sphere-packing bound ("only errors") case. The more u partially-stuck-at cells, the less amount of storable information, i.e. only $q - 1$ levels can be utilized. Hence, the dotted and dashed-dotted lines are declining while u is growing. On the other hand, the more errors (e.g., $t = 25$ in the dashed-dotted plot), the higher overall required redundancy and the lower storable information for the aforementioned curve.

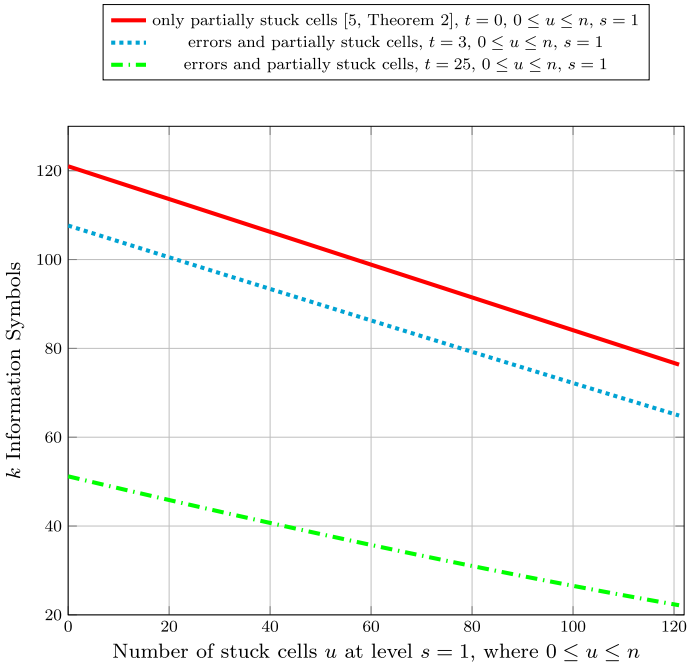

Fig. 2 **Sphere-packing bounds**: Comparison for k information symbols for ("only partially stuck cells [29, Theorem 2]") and our sphere-packing-like ("errors and partially stuck cells") bounds. The classical sphere-packing bound ("only errors") can read at $u = 0$ in our sphere-packing-like bounds curves. The chosen parameters are $\mu = 5$ and $q = 3$, and $n = ((q^\mu - 1)/(q - 1))$

8 Gilbert–Varshamov (GV) bound

8.1 Gilbert–Varshamov (GV) bound: finite length

We have provided various constructions of $(u, 1, t)$ PSMCs based on q-ary t-error correcting codes with additional properties. In this section, we first employ GV-like techniques to show the existence of $(u, 1, t)$ PSMCs. Next, we study the asymptotic of the resulting GV bounds.

We start with a somewhat refined version of the Gilbert bound, that should be well-known, but for which we did not find an explicit reference.

Lemma 5 *Let q be a prime power, and assume there is an $[n, s]_q$ code $\mathcal{C}_s$ with minimum distance at least d. If $k \ge s$ is such that*

$$\sum_{i=0}^{d-1} \binom{n}{i} (q-1)^i < q^{n-k+1},$$

then there is an $[n, k]_q$ code $\mathcal{C}_k$ with minimum distance at least d that has $\mathcal{C}_s$ as a subcode.

Proof By induction on k. For $k = s$, the statement is obvious. Now let $\kappa \ge s$ and let $\mathcal{C}_\kappa$ be an $[n, \kappa]_q$ code with minimum distance at least d that has $\mathcal{C}_s$ as a subcode. If $q^\kappa \sum_{i=0}^{d-1} \binom{n}{i} (q-1)^i < q^n$, then the balls with radius $d - 1$ centered at the words of $\mathcal{C}_\kappa$ do not cover $\mathbb{F}_q^n$, so there is a word $\boldsymbol{x}$ at distance at least d from all words in $\mathcal{C}_\kappa$. As shown in the proof of [28,

Theorem. 5.1.8], the $[n, \kappa+1]_q$ code $\mathcal{C}_{\kappa+1}$ spanned by $\mathcal{C}_\kappa$ and $\boldsymbol{x}$ has minimum distance at least d. □

8.1.1 Finite GV bound based on Lemma 2

Theorem 6 *Let q be a prime power. Let n, k, t, u be non-negative integers such that*

$$\sum_{i=0}^{2(t+\lfloor \frac{u}{q} \rfloor)} \binom{n}{i}(q-1)^i < q^{n-k+1}.$$

There exists a q-ary $(u, 1, t)$ PSMC of length n and size q^{k-1}.

Proof Let $\mathcal{C}_1$ be the $[n, 1, n]_q$ code generated by the all-one word. Lemma 5 implies that there is an $[n, k]_q$ code with minimum distance at least $2(t + \lfloor \frac{u}{q} \rfloor) + 1$ that contains the all-one word. Lemma 2 shows that $\mathcal{C}_k$ can be used to construct a PSMC with the claimed parameters. □

Remark 9 GV bound from Theorem 1 is a special case of Theorem 6 for $u \leq q-1$.

8.1.2 Finite GV bound based on Construction 2

Lemma 6 *Let q be a prime power, and let $1 \leq k < n$. Let $E \subset \mathbb{F}_q^n \backslash \{0\}$. The fraction of $[n, k]_q$ codes with non-empty intersection with E is less than $|E|/q^{n-k}$.*

Proof Let $\mathcal{C}$ be the set of all $[n, k]_q$ codes. Obviously,

$$\left|\{C \in \mathcal{C} \mid C \cap E \neq \emptyset\}\right| \leq \sum_{C \in \mathcal{C}: C \cap E \neq \emptyset} |\, C \cap E \,| = \sum_{C \in \mathcal{C}} |\, C \cap E \,|.$$

It follows from [18, Lemma 3] that

$$\frac{1}{|\,\mathcal{C}\,|} \sum_{C \in \mathcal{C}} |\, C \cap E \,| = \frac{q^k - 1}{q^n - 1}|E| < \frac{|E|}{q^{n-k}}.$$

□

Remark 10 If E has the additional property that $\lambda \boldsymbol{e} \in E$ for all $\boldsymbol{e} \in E$ and $\lambda \in \mathbb{F}_q \backslash \{0\}$, then the upper bound in Lemma 6 can be reduced to $|E|/(q-1)q^{n-k}$.

Lemma 7 *Let k, n, d and $d^\perp$ be integers such that*

$$\sum_{i=0}^{d-1} \binom{n}{i}(q-1)^i < \frac{1}{2}q^{n-k} \text{ and}$$

$$\sum_{i=0}^{d^\perp-1} \binom{n}{i}(q-1)^i < \frac{1}{2}q^k.$$

There exists a q-ary $[n, k]$ code $\mathcal{C}$ with minimum distance at least d such that $\mathcal{C}^\perp$ has minimum distance at least $d^\perp$.

Proof Let $\mathcal{C}$ denote the set of all $[n,k]_q$ codes. By applying Lemma 6 with $E = \{\boldsymbol{e} \in \mathbb{F}_q^n \mid 1 \leq \mathrm{wt}(\boldsymbol{e}) \leq d-1\}$ and using the first condition of the lemma, we see that more than half of the codes in $\mathcal{C}$ have empty intersection with E, that is, have minimum distance at least d. Similarly, more than half of all q-ary $[n, n-k]$ codes have minimum distance at least $d^\perp$, and so more than half of the codes in $\mathcal{C}$ have a dual with minimum distance at least $d^\perp$. We conclude that $\mathcal{C}$ contains a code with both desired properties. □

Theorem 7 (Gilbert-Varshamov-like bound by Construction 2) *Let q be a prime power. Suppose the positive integers u, t, n, k, l with $u, t \leq n$ and $l < k$ satisfy*

$$\sum_{i=0}^{2t} \binom{n}{i} (q-1)^i < \frac{1}{2} q^{n-l}, \tag{9}$$

$$\sum_{i=0}^{u-q+2} \binom{n}{i} (q-1)^i < \frac{1}{2} q^{l}, \tag{10}$$

$$\sum_{i=0}^{2t} \binom{n}{i} (q-1)^i < q^{n-k+1}. \tag{11}$$

Then there is a q-ary $(u, 1, t)$ PSMC of length n and cardinality q^{k-l}.

Proof According to Lemma 7, (9) and (10) imply the existence of an $[n,l]_q$ code $\mathcal{C}_0$ with minimum distance at least $2t+1$ for which the dual code has minimum distance at least $u-q+3$. Lemma 5 shows that $\mathcal{C}_0$ can be extended to an $[n,k]_q$ code $\mathcal{C}$ with minimum distance at least d. As $\mathcal{C}$ has a generator matrix of the form required by Construction 2, the theorem follows. □

8.1.3 Finite GV bound based on Proposition 1

In this section, we give sufficient conditions for the existence of matrices satisfying the conditions of Proposition 1. We start with Lemma 8 and 9, then we prove the main theorem (Theorem 8).

Lemma 8 *Let $\boldsymbol{G}$ be a $k \times n$ matrix over $\mathbb{F}_q$. For $s \geq 1$, let*

$$d_s = \min\{\mathrm{wt}(\boldsymbol{m}\boldsymbol{G}) \mid \boldsymbol{m} \in \mathbb{F}_{q^s}^k \setminus \{\boldsymbol{0}\}\}.$$

Then $d_s = d_1$.

Proof Let $s \geq 1$. As $\mathbb{F}_q \subseteq \mathbb{F}_{q^s}$, it is clear that $d_1 \geq d_s$.

To show the converse, we use the trace function T defined as $T(x) = \sum_{i=0}^{s-1} x^{q^i}$. As is well-known, T is a non-trivial mapping from $\mathbb{F}_{q^s}$ to $\mathbb{F}_q$, and

$$T(ax+by) = aT(x) + bT(y), \tag{12}$$

for all $x, y \in \mathbb{F}_{q^s}$ and $a, b \in \mathbb{F}_q$. We extend the trace function to vectors by applying it coordinate-wise.

Let $\boldsymbol{m} \in \mathbb{F}_{q^s}^k \setminus \{\boldsymbol{0}\}$. We choose $\lambda \in \mathbb{F}_{q^s}$ such that $T(\lambda \cdot \boldsymbol{m}) \neq \boldsymbol{0}$. As $T(0) = 0$, we infer that $\mathrm{wt}(\boldsymbol{m}\boldsymbol{G}) = \mathrm{wt}(\lambda \cdot \boldsymbol{m}\boldsymbol{G}) \geq \mathrm{wt}(T(\lambda \cdot \boldsymbol{m}\boldsymbol{G}))$. As all entries from $\boldsymbol{G}$ are in $\mathbb{F}_q$, it follows from (12) that $T(\lambda \cdot \boldsymbol{m}\boldsymbol{G}) = T(\lambda \cdot \boldsymbol{m})\boldsymbol{G}$. As a consequence,

$$\mathrm{wt}(\boldsymbol{m}\boldsymbol{G}) \geq \mathrm{wt}(T(\lambda \cdot \boldsymbol{m}\boldsymbol{G})) = \mathrm{wt}(T(\lambda \cdot \boldsymbol{m})\boldsymbol{G}) \geq d_1,$$

where the last inequality holds as $T(\lambda \cdot \boldsymbol{m}) \in \mathbb{F}_q^k \setminus \{\boldsymbol{0}\}$. □

Now we introduce Lemma 9 which is the binary version of Lemma 7 but with an extra restriction on the weight of the words.

Lemma 9 *Let k, n, d and $d^\perp$ be integers such that*

$$\sum_{i=0}^{d-1} \binom{n}{i} < \frac{1}{4} \cdot 2^{n-k} \text{ and } \sum_{i=0}^{d^\perp-1} \binom{n}{i} < \frac{1}{2} \cdot 2^k.$$

There exists a binary $[n, k]$ code $\mathcal{C}$ with minimum distance at least d without a word of weight more than $n - d + 1$ such that $\mathcal{C}^\perp$ has minimum distance at least $d^\perp$.

Proof Similar to the proof of Lemma 7. Let $\mathcal{C}$ denote the set of all binary $[n, k]$ codes. By applying Lemma 6 with $E = \{\boldsymbol{e} \in \mathbb{F}_2^n \mid 1 \leq \mathrm{wt}(\boldsymbol{e}) \leq d - 1 \text{ or } \mathrm{wt}(\boldsymbol{e}) \geq n - d + 1\}$, we infer that the first inequality implies that more than half of the codes in $\mathcal{C}$ contain no element from E. Similarly, the second inequality implies that more than half of the binary $[n, n - k]$ codes have minimum weight at least $d^\perp$, and so more than half of the codes in $\mathcal{C}$ have a dual with minimum distance at least $d^\perp$. We conclude that there is a code in $\mathcal{C}$ having both desired properties. □

Theorem 8 (Gilbert-Varshamov-like bound by Construction 3) *Let n, k, l, u, t, μ be positive integers with $u \leq n$, $2t < n$ and $l < k$ be such that*

$$\sum_{i=0}^{2t} \binom{n}{i} < \frac{1}{4} \cdot 2^{n-l}, \tag{13}$$

$$\sum_{i=0}^{\lfloor \frac{u}{2^{\mu-1}} \rfloor} \binom{n}{i} < \frac{1}{2} \cdot 2^l, \tag{14}$$

$$\sum_{i=0}^{2t} \binom{n}{i} (2^\mu - 1)^i < 2^{\mu(n-k+1)}, \tag{15}$$

Then there exists a $(u, 1, t)$ PSMC of length n over $\mathbb{F}_{2^\mu}$ with cardinality $2 \cdot 2^{\mu(k-l-1)} 2^{l(\mu-1)}$.

Proof By Lemma 9, there exists a binary $[n, l]$ code $\mathcal{C}_0$ with minimum distance at least $2t + 1$ for which $\mathcal{C}_0^\perp$ has minimum distance at least $\lfloor \frac{u}{2^{\mu-1}} \rfloor + 1$ with the following additional property: if $\boldsymbol{H}_0 \in \mathbb{F}_2^{l \times n}$ is a generator matrix for $\mathcal{C}_0$, then the binary code $\mathcal{C}_\mu$ with generator matrix $\begin{bmatrix} \boldsymbol{H}_0 \\ \mathbf{1} \end{bmatrix}$ has minimum distance at least $2t + 1$. According to Lemma 8, the code $\mathcal{C}_\mu$ over $\mathbb{F}_{2^\mu}$ with this generator matrix has minimum distance at least $2t + 1$ as well. Lemma 5 implies that $\mathcal{C}_\mu$ can be extended to an $[n, k]_{2^\mu}$ code with minimum distance at least $2t + 1$. The $[n, k]$ code has a generator matrix of the form $\boldsymbol{G} = \begin{bmatrix} \boldsymbol{H}_0 \\ \boldsymbol{G}_1 \\ \mathbf{1} \end{bmatrix}$. Application of Proposition 1 yields the claim. □

8.1.4 Finite GV bound from trivial construction

Clearly, a $(q - 1)$-ary code of length n with minimum distance at least $2t + 1$ is a q-ary $(u, 1, t)$ PSMC of length n with $u = n$. Combining this observation with the Gilbert bound for a $(q - 1)$-ary alphabet, we obtain the following corollary.

Corollary 5 *Let $q \geq 3$, and let*

$$M = \left\lceil \frac{(q-1)^n}{\sum_{i=0}^{2t} \binom{n}{i}(q-2)^i} \right\rceil.$$

There is a q-ary $(n, 1, t)$ PSMC of length n and cardinality M.

So far we have covered the GV-like bounds for our code constructions for finite length n.

8.2 Asymptotic GV bound on PSMCs

In this section, we present the asymptotic version of the GV bounds from the previous section. That is, we provide lower bounds on the achievable rates of a q-ary $(u, 1, t)$ PSMCs in the regime that the code length n tends to infinity, and the number u of partially-stuck-at cells and the number t of random errors both grow linearly in n.

We recall the well-known following lemma that estimates the volume of a Hamming ball using the q-ary entropy function.

Lemma 10 *For positive integers n, $q \geq 2$ and real δ, $0 \leq \delta \leq 1 - \frac{1}{q}$,*

$$\mathrm{Vol}_q(n, \delta n) \leq q^{h_q(\delta)n},$$

where $\mathrm{Vol}_q(n, r) = \sum_{j=0}^{r} \binom{n}{j}(q-1)^j$ denotes the volume of a Hamming ball with radius r.

Proof The proof of Lemma 10 has been stated in many references including [22, p.105] and [10, Proposition 3.3.1]. □

8.2.1 Asymptotic bound for Theorem 6

Theorem 9 *Let q be a prime power. Let $0 \leq \tau, \upsilon < 1$ be such that*

$$2\left(\tau + \frac{\upsilon}{q}\right) < 1 - \frac{1}{q}.$$

For sufficiently large n, there exists an $(\lfloor \upsilon n \rfloor, 1, \lfloor \tau n \rfloor)$ PSMC of length n and rate at least

$$1 - h_q\left(2\left(\tau + \frac{\upsilon}{q}\right)\right) - \frac{2}{n}.$$

Proof Let n be a positive integer such that $\lceil nh_q(2(\tau + \frac{\upsilon}{q})) \rceil < n$. Let $t = \lfloor \tau n \rfloor$ and $u = \lfloor \upsilon n \rfloor$. Take $k = n - \lceil nh_q(2\tau + 2\frac{\upsilon}{q}) \rceil$. Lemma 10 implies that $\mathrm{Vol}_q(n, 2t + 2\lfloor \frac{u}{q} \rfloor) \leq q^{n-k}$, and so, according to Theorem 6, there is a q-ary $(u, 1, t)$ PSMC of length n with rate $\frac{k-1}{n} \geq 1 - h_q(2(\tau + \frac{\upsilon}{q})) - \frac{2}{n}$. □

8.2.2 Asymptotic GV bound from Construction 2

Theorem 10 (Asymptotic Gilbert-Varshamov-like bound from Theorem 7) *Let q be a prime power. Let υ, τ be such that*

$$0 < \upsilon, 2\tau < 1 - \frac{1}{q} \text{ and } h_q(\upsilon) + h_q(2\tau) < 1.$$

For sufficiently large n, there exists a q-ary $(\lfloor \upsilon n \rfloor, 1, \lfloor \tau n \rfloor)$ *PSMC of length n and rate at least*

$$1 - h_q(2\tau) - h_q(\upsilon) - \frac{4\log_q(2) + 2}{n}.$$

Proof Let n be a positive integer. Write $u = \lfloor \upsilon n \rfloor$ and $t = \lfloor \tau n \rfloor$. Then $\mathrm{Vol}_q(n, u - q + 2) \leq \mathrm{Vol}_q(n, u)$. Hence, by setting

$$l = \lceil n h_q(\upsilon) + 2\log_q(2) \rceil,$$

Lemma 10 implies that (10) is satisfied.

Similarly, by setting

$$k = n - \lceil n h_q(2\tau) + 2\log_q(2) \rceil,$$

it is ensured that (11) is satisfied.
According to Theorem 7, there is a q-ary $(u, 1, t)$ PSMC of length n and size q^{k-l}, so with rate $k - l$. The choices for k and l show that the theorem is true. □

Remark 11 Theorem 10 in fact holds for classical stuck-at cells instead of stuck-at-1 errors, as follows from considering the generalization of Theorem 7 in Proposition 2, i.e., Heegard's construction [12].

8.2.3 Asymptotic GV bound from Construction 3

Theorem 11 (Asymptotic Gilbert-Varshamov-like bound from Theorem 8) *Let* μ *be a positive integer, and let* υ *and* τ *be such that*

$$0 \leq \frac{\upsilon}{2^{\mu-1}} < \frac{1}{2}, 0 < 2\tau < \frac{1}{2}, \text{ and } h_2\left(\frac{\upsilon}{2^{\mu-1}}\right) + h_2(2\tau) < 1.$$

For sufficiently large n there is a 2^μ*-ary* $(\lfloor \upsilon n \rfloor, 1, \lfloor \tau n \rfloor)$ *PSMC of length n and rate at least*

$$1 - h_{2^\mu}(2\tau) - \frac{1}{\mu} h_2\left(\frac{\upsilon}{2^{\mu-1}}\right) - \frac{2}{n} - \frac{3}{\mu n}.$$

Proof For notational convenience, we set $\upsilon_0 = \frac{\upsilon}{2^{\mu-1}}$ and $\eta = 1 - h_2(2\tau) - h_2(\upsilon_0)$. Note that $\eta > 0$.
Let n be a positive integer satisfying $n \geq \frac{7}{\eta}$, and let $u = \lfloor \upsilon n \rfloor$, $u_0 = \lfloor \frac{u}{2^{\mu-1}} \rfloor$ and $t = \lfloor \tau n \rfloor$. We set

$$l = \lceil n h_2(\upsilon_0) \rceil + 3.$$

Lemma 10 implies that (14) is satisfied. Moreover, as

$$n - l - 3 \geq n - n h_2(\upsilon_0) - 7 = n h_2(2\tau) + n\eta - 7 \geq n h_2(2\tau),$$

Lemma 10 implies that (13) is satisfied.

We set

$$k = n - \lceil n h_{2^\mu}(2\tau) \rceil.$$

Lemma 10 implies that (15) is satisfied.

According to [10, Corollary 3.3.4], we have that $h_{2^\mu}(2\tau) \leq h_2(2\tau)$, and so

$$k - l \geq n - n h_2(2\tau) - 1 - n h_2(\upsilon_0) - 4 = n\eta - 5 \geq 2.$$

Theorem 8 implies the existence of a 2^μ-ary $(u, 1, t)$ PSMC of length n with size $2 \cdot 2^{\mu(k-1)} 2^{-l}$, i.e., its rate is

$$\frac{k-1}{n} - \frac{l-1}{\mu n} \geq 1 - h_{2^\mu}(2\tau) - \frac{1}{\mu} h_2(\upsilon_0) - \frac{2}{n} - \frac{3}{\mu n}.$$

□

Theorem 12 (Asymptotic Gilbert-Varshamov bound from Corollary 5) *Let $q \geq 3$. For each positive integer n and each τ with $0 \leq 2\tau < 1 - \frac{1}{q-1}$, there exists a q-ary $(n, 1, \lfloor \tau n \rfloor)$ PSMC of length n and rate at least*

$$(1 - h_{q-1}(2\tau)) \cdot \log_q(q-1).$$

Proof Let $t = \lfloor \tau n \rfloor$. Corollary 5 implies the existence of a q-ary $(n, 1, t)$ PSMC of length n and cardinality M satisfying

$$M \geq \frac{(q-1)^n}{V_{q-1}(n, 2t)} \geq (q-1)^{n(1-h_{q-1}(2\tau))},$$

where the last inequality holds by Lemma 10. □

9 Comparisons

We provide different comparisons between our code constructions and the existence of the code based on Theorems 6, 7 and 8. Next, we also compare to the known limits and investigate the trade-off between masking and error-correction as described in Sect. 6.

9.1 Comparison of Theorem 6 for $u \leq q-1$ to other Bounds

Figure 3 illustrates the rates of a $(q-1, 1, t)$ PSMC obtained from Theorem 1 (applying Theorem 6 for the special case $u \leq q-1$) for $n = 114$, $q = 7$ and $0 \leq t \leq 56$. We show how close explicit BCH codes that contain the all-one word of certain rates R and that can correct designed distances $d \geq 2t+1$ to the achieved rates from Theorem 1. We note that the solid red graph matches the dashed-dotted green plot for a few code parameters and overpasses it for $t = 39$. We also compare to the classical q-ary GV bound (in dashed black) as well as to reduced alphabet $(q-1)$-ary GV bound (in dashed-dotted blue). To this end, we show upper bounds on the rates that can be obtained from Theorem 1 using the Griesmer bound [9], and the Ball–Blokhuis bound [1] on the size of codes containing the all-one word.

9.2 Comparison among Theorems 7, 8 and $(q-1)$-ary Gilbert-Varshamov bound

We plot the achievable rates ($R = log_q(M)/n$) as a function of t for different fixed values of u. Figure 4 is the resulting plot for $n = 200$, $\mu = 3$ and $q = 2^\mu$. It can be seen that the GV-like bound in different ranges of u and t based on Construction 2 improves upon the $(q-1)$-ary GV bound for $u \leq 5$ as depicted in the solid red curve, and improves further (up to $u \leq 20$) based on Construction 3 as shown in the dashed gray line (3rd one from above). The dashed dotted blue curve is used to see what if we map our 2^3 levels such that we avoid the subscript 0 to compare with 7 levels. It is obvious that for $\mu = 3$, the rate loss[1] resulting

[1] For $t = 0$, the loss is $1 - \log_8(7) = 0.0642$.

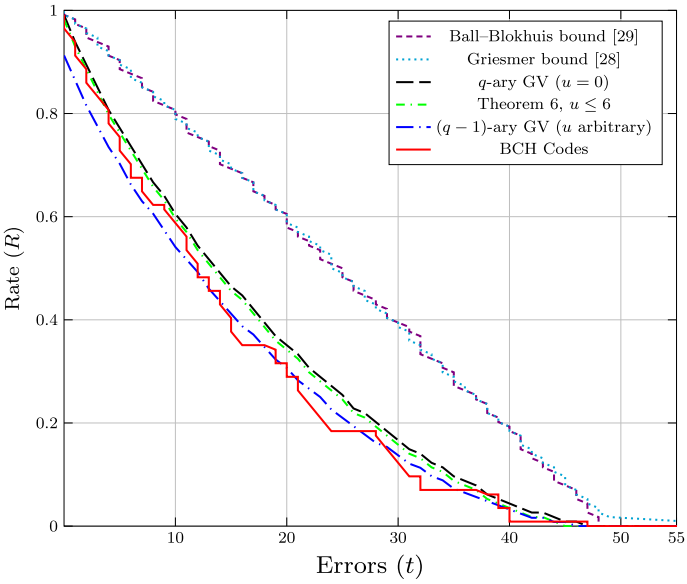

Fig. 3 Comparison of other upper and lower limits to our derived GV-like bound in Theorem 6 taking $n = 114$, $q = 7, 0 \leq t \leq 56$ and $u \leq q-1$. The dashed-dotted green curve shows the rates for Theorem 1 by Theorem 6 for $u \leq q-1$ in which codes that have the all-one words are considered. This curve for several code parameters matches the red line that shows the rates of BCH codes that contain all-one word with regard to the designed distances $d \geq 2t+1$

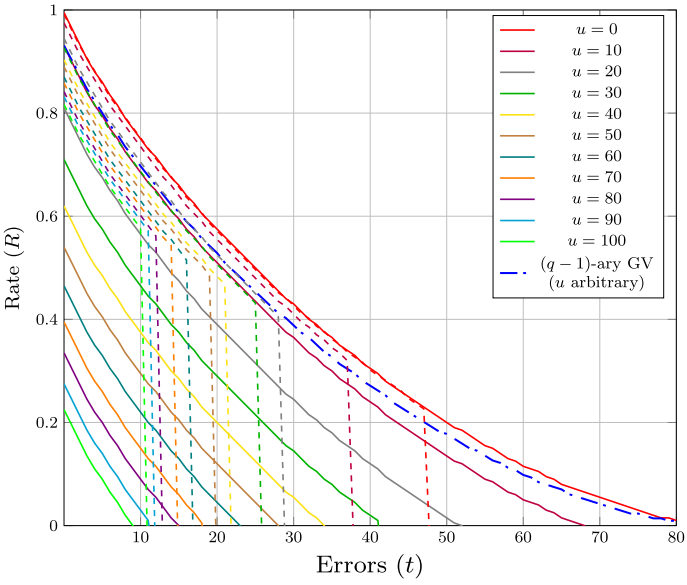

Fig. 4 The achievable rates $R = \frac{1}{n}\log_{2^3}\mathcal{M}$ of GV bounds for different u, t for $n = 200$ and $q = 2^3$ in Theorems 7 and 8, where $\mathcal{M}$ is the code cardinality. They are also compared to the rates from an ordinary 7-ary GV bound for different t as illustrated in the dashed-dotted blue plot. The solid and the dashed lines represent the derived GV like bounds from Theorems 7 and 8, respectively

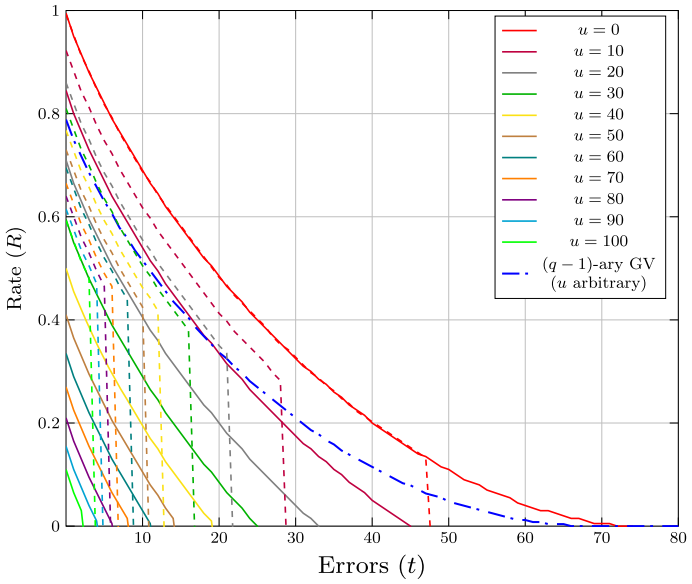

Fig. 5 The achievable rates $R = \frac{1}{n}\log_{22}\mathcal{M}$ of GV bounds for different u, t for $n = 200$ and $q = 2^2$ in Theorems 7 and 8. They are also compared to the rates from an ordinary 3-ary GV bound for different t as illustrated in the dashed-dotted blue plot. The solid and the dashed lines correspond to the derived GV like bounds by Theorem 7 and by Theorem 8, respectively

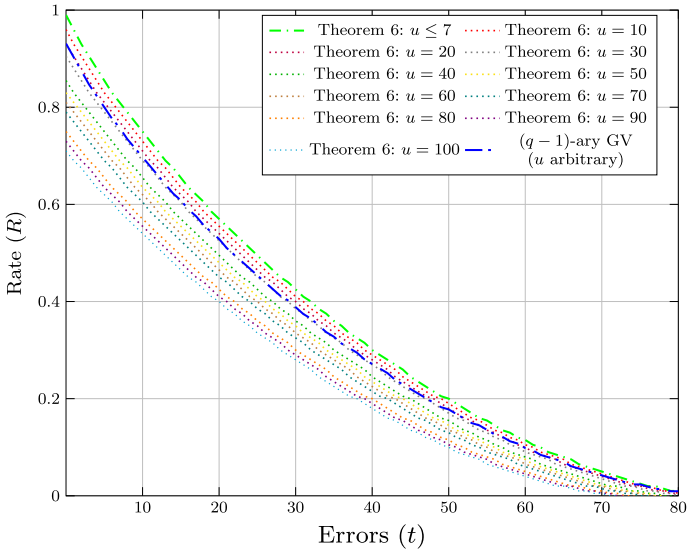

Fig. 6 The achievable rates $R = \frac{1}{n}\log_{23}\mathcal{M}$ of GV bounds for different u, t for $n = 200$ and $q = 2^3$ in Theorem 6 that are compared to the reduced alphabet conventional $(q-1)$-ary GV bound for different t. The dashed-dotted green curve represents the rates from Theorem 6 when $u \leq q - 1$

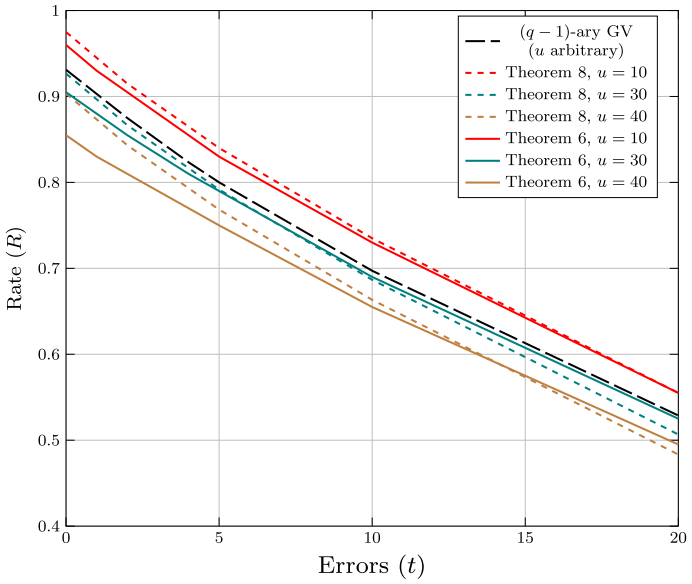

Fig. 7 The achievable rates $R = \frac{1}{n}\log_q \mathcal{M}$ of GV bounds for different $u = \{10, 30, 40\}$, and $t = \{0, 1, 2, 4, 5, 10, 20\}$ for $n = 200$ and $q = 2^3$ in Theorems 8 and 6. They are also compared to the rates for $(q-1)$-ary GV bound as shown in dashed black graph

from using $q-1$ instead of q symbols is already quite small. Note that for $u = 0$ the solid red curve mostly achieves the exact rates obtained from the standard 2^3-ary GV bound for $0 \le t \le 80$, and so as for the dashed red curve but for $0 \le t \le 47$.

For $\mu = 2$, the improvements from Construction 2 ($u \le 10$) and Construction 3 ($u \le 30$) upon a usual $(q-1)$-ary GV bound are more significant as shown in Fig. 5.

9.3 Comparisons between Theorem 6 and $(q-1)$-ary Gilbert-Varshamov bound

In Fig. 6, we compare the GV like bound from Theorem 6 for $q = 2^3$ with the conventional GV bound for $q - 1 = 7$ shown in dashed blue curve. For $q = 8$, we observe that the conventional $q-1$-ary GV bound is superior to the derived GV-like bound from Theorem 6 for $u \ge 40$. However, applying Theorem 6 where $u \le 20$, the traditional $q-1$-ary GV bound is a bad choice. We observe that the dashed-dotted green curve by Theorem 6 for ($u \le 7$ as stated in Remark 9) shows the highest rates.

9.4 Comparisons between Theorems 8 and 6

In Fig. 7, we compare Theorems 8 and 6. Theorem 8 is showing higher rates for larger u values, for example taking $u = 40$ and $t = 1$, the rate is $R = 0.87$ from Theorem 8 while $R = 0.83$ from Theorem 6. It is interesting to note that for $u = 30$ and $t > 10$ Theorem 6 is better, and for $u = 10$ and $t > 18$ Theorem 6 is as good as Theorem 8.

Table 4 Table of selected points from Fig. 8 with slightly lower and higher rates due to trading. All points are from Theorem 7

u \ t	13	14	15	...	40	41	42
16	0.560	0.545	**0.525**	...	0.170	**0.160**	0.150
17	0.545	**0.530**	0.510	...	**0.155**	0.145	0.135
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
21	0.505	0.490	**0.470**	...	0.115	**0.105**	0.095
22	0.490	**0.475**	0.455	...	**0.100**	0.090	0.080
23	0.480	0.465	0.445	...	0.090	0.080	0.070

9.5 Comparisons of application of Theorem 5 vs direct application of Theorem 7

For given (u, t), we illustrate the trading $(u+1, t-1)$ in Fig. 8. For some of t and a few of u values, it is advantageous if the encoder introduces an error in a partially-stuck-at position in order to mask this position. The orange solid curve, for instance, represents the rates that have been determined by Theorem 7 for $u = 17$ and $0 \leq t \leq 50$, while the orange dotted sketch highlights the rates for $u = 16$ while $1 \leq t \leq 51$. Due to the exchange such that $u + 1 = 17$ and $0 \leq t - 1 \leq 50$, the orange dotted line slightly fluctuates up and down the rates shown in the orange solid curve for most t values.

Let us describe some points of Fig. 8 in Table 4. Let $\mathcal{C}_{u,t}$ be a code by Theorem 7 whose rate is R given in Table 4 at u row and t column. Take $\mathcal{C}_{21,15}$ so that its rate $R = \mathbf{0.470}$. By applying Theorem 5 on $\mathcal{C}_{21,15}$, we obtain a code $\mathcal{C}_{22,14}$ of $R = 0.470$. Direct application of Theorem 7 yields a $\mathcal{C}_{22,14}$ of rate $R = \mathbf{0.475}$ as highlighted in Table 4. We conclude that in this case, the trade by Theorem 5 gives lower rates than taking the same code directly by Theorem 7 for given $(u = 22, t = 14)$.

On contrary, for larger t values, Table 4 shows that the exchange is beneficial giving higher rates. For example, we start with $\mathcal{C}_{21,41}$ whose $R = \mathbf{0.105}$, then applying Theorem 5 gives $\mathcal{C}_{22,40}$ of $R = 0.105$ which is greater than $R = \mathbf{0.100}$ that has been obtained directly by Theorem 7 as stated in Table 4.

9.6 Comparisons of applications of Theorem 5, Lemma 1, Lemma 2 vs direct application of Theorem 8

For the derived GV bound based on Construction 3 obtained by Theorem 8, we demonstrate the exchange of a one error correction ability with a single masking capability of a partially stuck cell following Theorem 5 in Fig. 9. The solid and dotted lines represent the rates before and after trading, respectively. We also show the exchange by Lemmas 1 and 2 in which the reduction of the correctable errors by one increases u by $2^{\mu-1}$ and 2^{μ}, respectively. As it is seen in Fig. 9, every single solid curve by Theorem 8 corresponds to multiple values of u due to the floor operation where $\mathcal{C}_0^{\perp}$ has a minimum distance at least $\lfloor \frac{u}{2^{\mu-1}} \rfloor + 1$. For instance, for all $u = 8, 9, 10, 11$ and $\mu = 3$, we obtain: $\lfloor \frac{u}{2^{3-1}} \rfloor + 1 = 3$, and the transition starts for $u = 12$ as $\lfloor \frac{u}{2^{3-1}} \rfloor + 1 = 4$. Similarly by the floor operation there is no change for the larger values of $u = 13, 14, 15$, and so on. Let us discuss the following curves. For $u = 19$, the orange solid curve shows the rates by Theorem 8. Exchanging $u + 1$ and $t - 1$ throughout

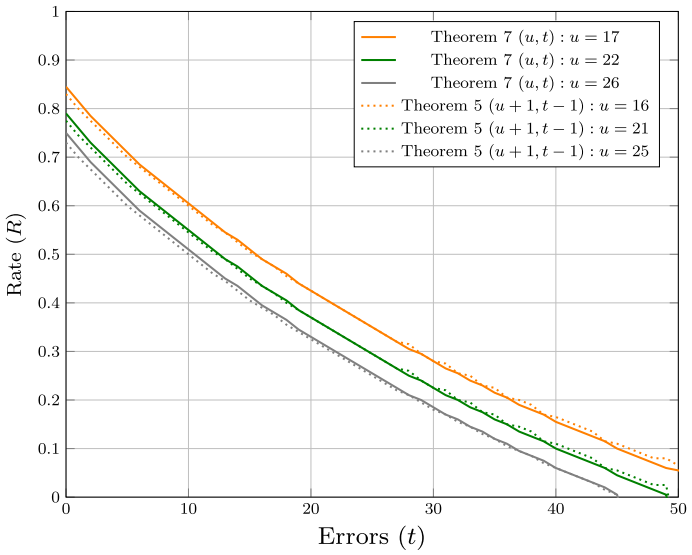

Fig. 8 The achievable rates $R = \frac{1}{n}\log_q \mathcal{M}$ of GV bounds for different u, and t for $n = 200$ and $q = 2^3$ in Theorem 7. The solid plots are the rates from the derived GV-like bound and the dotted graphs are the rates after trading $u + 1, t - 1$ by Theorem 5

Theorem 5 obtains the orange dotted line for $u+1 = 20$ which lies slightly bellow the orange solid plot. Hence, the exchange gives lower rates. However, it provides rate $R = 0.380$ for $t = 30$ while direct application of Theorem 8 (compared to its corresponding graph which is the solid green curve at $u = 20, 21, 22, 23$) does not.

Now, we apply Lemma 1 rather than Theorem 5. For the same achieved rates, we observe that the dashed red graph for $u + 2^{3-1} = 23$ shows the exact rates from the orange dotted curve for $u + 1 = 20$. Therefore, it is clear that Lemma 1 provides a gain of masking exactly 3 more cells with regard to Theorem 5. On the other hand, for the same masked u, applying Lemma 1 for $u = 19$ gives $(u + 4, t - 1)$ PSMC represented by the dashed red graph which presents slightly higher rates if compared to the dotted green for $u = 23$ by Theorem 5. Further, Lemma 1 provides rate at $t = 30$ while Theorem 5 stops giving rates at $t \geq 29$. For this graph, we conclude that Lemma 1 surpasses Theorem 5.

However, if we take $u = 23$ directly by Theorem 8, we achieve slightly higher rates. We conclude that Theorem 8 can directly estimate the maximum possible masked u cells that can also be achieved applying Lemma 1, and can achieve slightly higher rates. On contrary, Theorem 8 does not give rates for larger t values while Lemma 1 and Theorem 5 do that.

On the other hand, as Theorem 8 is based on Construction 3 that contains a word of weight n, Lemma 2 is applicable under the condition that $2(t + \lfloor \frac{u}{q} \rfloor) < d$ (cf. Remark 6). Hence, we can achieve higher rates as shown in the dashed-dotted curve while masking up to the same number of u cells, rather employing Lemma 1 or Theorem 5.

For that we describe some points of Fig. 9 by Table 5. Let $\mathcal{C}_{u,t}$ be a code by Theorem 8 whose rate is R given in Table 5 at u row and t column. Taking $\mathcal{C}_{19,27}$ gives $\mathcal{C}_{20,26}$ and $\mathcal{C}_{23,26}$ with $R = 0.435$ applying Theorem 5 and Lemma 1, respectively. In contrary, taking $\mathcal{C}_{19,31}$ is advantageous as there are codes ($\mathcal{C}_{20,30}$ by Theorem 5 and $\mathcal{C}_{23,30}$ by Lemma 1) with $R = 0.380$ while direct application of Theorem 8 cannot provide these codes as highlighted in green with "**None**". Now, we apply Lemma 2 on a code obtained by Theorem 6 for $(u = 7, t = 27)$

Table 5 Table of selected points from Fig. 9. All points are from Theorem 8

u \ t	26	27	28	29	30	31	32
11	0.471	0.456	0.441	0.431	0.416	0.401	0.391
12	0.460	0.445	0.430	0.420	0.405	0.390	0.380
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
14	0.460	0.445	0.430	0.420	0.405	0.390	0.380
15	0.460	0.445	0.430	0.420	0.405	0.390	0.380
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
19	0.450	**0.435**	0.420	0.410	0.395	**0.380**	None
20	**0.441**	0.426	0.411	0.401	**None**	None	None
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
23	**0.441**	0.426	0.411	0.401	**None**	None	None

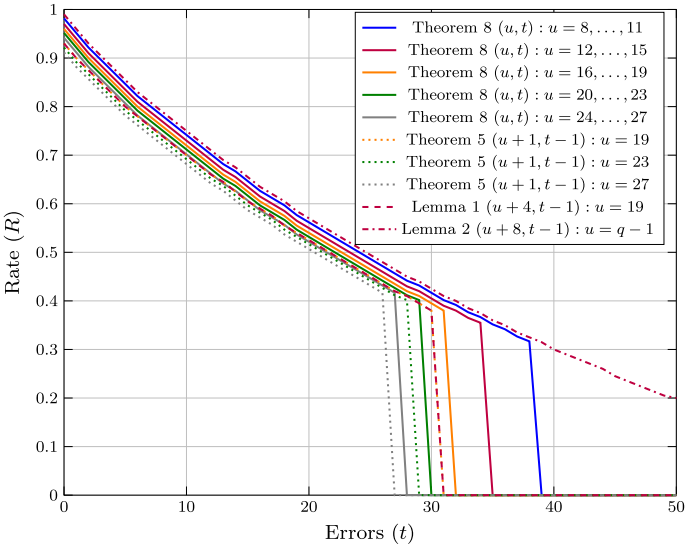

Fig. 9 The achievable rates $R = \frac{1}{n}\log_q \mathcal{M}$ of GV bounds for different u, and t for $n = 200$ and $q = 2^3$ in Theorem 8. The solid plots are the rates from the derived GV like bound and the dotted graphs are the rates after trading $u+1, t-1$ by Theorem 5. We also show the exchange by Lemma 1 and Lemma 2 in which the reduction of the correctable errors by one increases u by $2^{\mu-1}$ and 2^{μ}, respectively

to obtain the code $\mathcal{C}_{15,26}$ of rate $R = 0.465$ that satisfies $2(26 + \lfloor \frac{15}{8} \rfloor) < 55$. The achieved rate is higher compared to $\mathcal{C}_{15,26}$ of $R = 0.460$ that is directly obtained by Theorem 8, or applying Theorem 5 on $\mathcal{C}_{14,27}$ to obtain $\mathcal{C}_{15,26}$ of $R = 0.445$, or using Lemma 1 on $\mathcal{C}_{11,27}$ to obtain $\mathcal{C}_{15,26}$ of $R = 0.456$. This result does not mean that application Lemma 2 on a code obtained by Theorem 6 always provides higher code rates for the same parameters u, t (see Fig. 7).

9.7 Analytical comparison of asymptotic GV-like bounds

In this section, we state the results of the analytical comparisons of the asymptotic GV bounds from Theorems 9, 10 and 11, ignoring the terms that tend to zero for increasing n. We will use the following lemma.

Lemma 11 *Let $q \geq 2$ be an integer. If $0 \leq x, y$ are such that $x + y \leq 1$, then*

$$h_q(x + y) \leq h_q(x) + h_q(y).$$

Proof Let $0 \leq y < 1$, and consider the function $f_y(x) = h_q(x + y) - h_q(x) - h_q(y)$ on the interval $[0, 1 - y]$. Clearly, $f'_y(x) = h'_q(x + y) - h'_q(x) \leq 0$, where the inequality follows from the fact that the second derivative of h_q is non-negative. Hence, $f_y(x) \leq f_y(0) = 0$ for each $x \in [0, 1 - y]$. □

Proposition 5 *If υ, τ and q are such that the conditions of Theorems 9 and 10 are met, then the rate guaranteed by Theorem 9 is at least equal to the code rate guaranteed by Theorem 10.*

Proof Assume τ and υ are such that the conditions of Theorems 9 and 10 are satisfied, that is, such that $2\tau + 2\frac{\upsilon}{q} < 1 - \frac{1}{q}$ and $h_q(\upsilon) + h_q(2\tau) < 1$. By invoking Lemma 11, we see that

$$h_q(2\tau + 2\frac{\upsilon}{q}) \leq h_q(2\tau) + h_q(2 \cdot \frac{\upsilon}{q}) \leq h_q(2\tau) + h_q(\upsilon),$$

where the final inequality holds as $q \geq 2$ and h_q is monotonically increasing on $[0, 1 - \frac{1}{q}]$. As a consequence, the code rate guaranteed by Theorem 9 is at least equal to the code rate guaranteed by Theorem 10. □

Proposition 6 *If υ, τ and $q = 2^\mu$ are such that the conditions of Theorems 10 and 11 are met, then the code rate guaranteed by Theorem 11 is at least equal to the code rate guaranteed by Theorem 10.*

Proof Assume that the conditions of Theorems 10 and 11 are satisfied. The difference between the rate of Theorem 11 and of Theorem 10 equals

$$h_{2^\mu}(\upsilon) - \frac{1}{\mu} h_2(\frac{\upsilon}{2^{\mu-1}}). \tag{16}$$

According to the conditions of Theorem 10, $\upsilon \leq 1 - \frac{1}{2^\mu}$, and so $h_{2^\mu}(\upsilon) \geq h_2(\frac{\upsilon}{2^{\mu-1}})$. As $h_{2^\mu}(x) = \frac{1}{\mu} h_2(x) + x \log_{2^\mu}(2^\mu - 1)$, the difference in (16) is non-negative. That is, Theorem 11 is better than Theorem 10. □

We note that the requirement $2\tau < \frac{1}{2}$ from Theorem 11 is stricter than the requirement $2\tau < 1 - \frac{1}{2^\mu}$ from Theorem 10. That is, there are pairs (τ, υ) for which Theorem 10 is applicable, but Theorem 11 is not.

Comparison of Theorems 9 and 11 is more complicated. We have the following partial result.

Proposition 7 *Let $\upsilon, \tau > 0$ and $q = 2^\mu$ be such that the conditions of Theorems 9 and 11 are met. If υ is sufficiently small, then the rate guaranteed by Theorem 9 is larger than the rate guaranteed by Theorem 11.*

Proof Assume that τ and υ are such that the conditions of Theorem 9 and of Theorem 11 are satisfied, that is, $2\tau + 2\frac{\upsilon}{2^\mu} < 1 - \frac{1}{2^\mu}$,

$$0 \leq \upsilon \leq 2^{\mu-2},\ 0 \leq 2\tau \leq \frac{1}{2} \text{ and } h_2(\frac{\upsilon}{2^{\mu-1}}) + h_2(2\tau) < 1.$$

Let $f_\mu(\tau, \upsilon_0)$, where $\upsilon_0 = \frac{\upsilon}{2^{\mu-1}}$, be the bound from Theorem 9 minus the bound from Theorem 11, that is

$$f_\mu(\tau, \upsilon_0) = h_{2^\mu}(2\tau) + \frac{1}{\mu} h_2(\upsilon_0) - h_{2^\mu}(2\tau + \upsilon_0).$$

The definition of the entropy function implies that for any $x \in [0, 1]$

$$h_{2^\mu}(x) = \frac{1}{\mu} \left(h_2(x) + x \log_2(2^\mu - 1) \right). \tag{17}$$

Applying (17), we infer that

$$\mu f_\mu(\tau, \upsilon_0) = h_2(2\tau) + h_2(\upsilon_0) - h_2(2\tau + \upsilon_0) - \upsilon_0 \log_2(2^\mu - 1). \tag{18}$$

In particular, $\mu f_\mu(0, \upsilon_0) = -\upsilon_0 \log_2(2^\mu - 1) \leq 0$.

So for $\tau = 0$, Theorem 11 is better than Theorem 9. It follows from Lemma 11 that the three leftmost terms in (18) form a non-negative number. The subtraction of the fourth term, however, can result in a negative function value, especially for large μ.

Example 6 (*Numerical example*) $\mu f_\mu(0.055, 0.11) = 2 \cdot h_2(0.11) - h_2(0.22) - 0.11 \log_2(2^\mu - 1) \approx 0.23397 - 0.11 \log_2(2^\mu - 1)$ is positive for $\mu \leq 2$ and negative otherwise. □

We now prove Proposition 7. That is, we show that for $\tau > 0$ and υ_0 sufficiently small, $f_\mu(\tau, \upsilon_0) > 0$. This follows from the Taylor expansion of $\mu f_\mu(\tau, \upsilon_0)$ around $\upsilon_0 = 0$. Indeed, $f_\mu(\tau, 0) = 0$, and $h_2'(x) \to \infty$ if $x \downarrow 0$. □

10 Conclusion

In this paper, code constructions and bounds for non-volatile memories with partial defects have been proposed. Our constructions can handle both: partial defects (also called partially stuck cells) and random substitution errors, and require less redundancy symbols for $u > 1$ and $q > 2$ than the known constructions for stuck cells. Compared to error-free masking of partially stuck cells, our achieved code sizes coincide with those in [29], or are even larger as shown in Proposition 1. We summarize our constructions and the previous works on partially/fully stuck cells in Table 3.

Further, we have shown that it can be advantageous to introduce errors in some partially stuck cells in order to satisfy the stuck-at constraints. For the general case that is applicable for all of our constructions, we have shown in Theorem 5 how to replace any $0 \leq j \leq t$ errors by j masked partially stuck cells. This theorem has been improved for Construction 3 by Lemma 1, and further enhanced by another method for introducing errors in the partially stuck locations through Lemma 2 (cf. Example 5). We gain (e.g., for $j = 1$) exactly $2^{\mu-1}$ and 2^μ (under the condition that $2(t + \lfloor \frac{u}{q} \rfloor) < d$) additional masked partially stuck cells applying Lemmas 1 and 2, respectively. So far, determining if introducing errors in partially stuck cells is advantageous or not can only be done numerically.

We also derived upper and lower limits on the size of our constructions. Our sphere-packing-like bound for the size of (Σ, t) PSMCs has been compared to the usual sphere-packing upper bound, and for the case of no errors ($t = 0$) to [29, Theorem 2].

We have numerically compared our Gilbert–Varshamov-type bounds, for given (u, t), to each other and to $(q-1)$-ary codes. For $u \leq q-1$, Theorem 6 states the existence of $(u, 1, t)$ PSMCs with rates that almost match the ones from the usual q-ary GV bound (shown in Fig. 3). Moreover, up to $u = 20$ for $q = 8$, Fig. 6 shows that application of Theorem 6 is better than using $(q-1)$-ary code as mentioned in [29, Section III]. On the other hand, for $q = 4$ and $u = 10$, Theorems 7 and 8 obviously require less redundancy than $(q-1)$-ary code as shown in Fig. 5.

Figures 8 and 9 demonstrate the application of Theorem 5, Lemmas 1 and 2 on $(u, 1, t)$ PSMCs of rates that have been obtained based on Theorems 7 and 8. For some parameters (i.e. $u = 16, t = 41$ as shown in Table 4 and $u = 19, t = 31$ as shown in Table 5), application of Theorem 5 and Lemma 1 achieve higher code rates and more masked cells. Application Lemma 2 on a code obtained by Theorem 6 (i.e. $u = 7$, $t = 27$) provides higher code rate compared to the direct employment of Theorems 8, 5 and Lemma 1.

In the asymptotic regime of our GV-like bounds, Theorems 9 and 11 are remarkable competitors to Theorem 10. However, the analytical comparison between Theorem 9 and Theorem 11 is more complicated to decide which one is the better choice. This was also confirmed numerically via Fig. 7.

Funding Open Access funding enabled and organized by Projekt DEAL.

Data Deposition Information The authors declare that all other data supporting the findings of this study are available within the article and its supplementary information files.

Declarations

Conflict of interest The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

A An alternative Proof of Proposition 2

We start with Lemma 12.

Lemma 12 *Let $\boldsymbol{M} \in \mathbb{F}_q^{m \times n}$ be such that each column of $\boldsymbol{M}$ has at least one non-zero entry. Let $\boldsymbol{s} \in \mathbb{F}_q^n$. For each $\boldsymbol{w} \in \mathbb{F}_q^n$, there is a $\boldsymbol{v} \in \mathbb{F}_q^m$ such that*

$$\left|\left\{i \in [n] \mid w_i + (\boldsymbol{v}\boldsymbol{M})_i \geq s_i\right\}\right| \geq n - \left\lfloor \frac{1}{q} \sum_{i=0}^{n-1} s_i \right\rfloor.$$

Proof We define the set S as

$$S = \left\{(i, \boldsymbol{v}) \in [n] \times \mathbb{F}_q^m \mid w_i + (\boldsymbol{vM})_i \geq s_i\right\}.$$

Clearly, there is $\boldsymbol{v} \in \mathbb{F}_q^m$ such that

$$\left|\left\{i \in [n] \;\middle|\; w_i + (\boldsymbol{vM})_i \geq s_i\right\}\right| \;\geq \left\lceil \frac{|S|}{q^m} \right\rceil. \tag{19}$$

Let $i \in [n]$. As the i-th column of $\boldsymbol{M}$ has a non-zero entry, for each $y \in \mathbb{F}_q$ there are exactly q^{m-1} vectors $\boldsymbol{x} \in \mathbb{F}_q^m$ such that $(\boldsymbol{xM})_i = y$. As a consequence,

$$\left|\left\{\boldsymbol{v} \in \mathbb{F}_q^m \mid w_i + (\boldsymbol{vM})_i \geq s_i\right\}\right| \;= (q - s_i)q^{m-1},$$

and so

$$|S| = \sum_{i=0}^{n-1} (q - s_i)q^{m-1} = nq^m - q^{m-1} \sum_{i=0}^{n-1} s_i. \tag{20}$$

The lemma follows from combining (19) and (20). □

We are now in a position to introduce an alternative *non-constructive* proof for Proposition 2.

Let $\boldsymbol{s} \in \Sigma$. In order to simplify notation, we assume without loss of generality that $\sum_{i=d_0-2}^{n-1} s_i \leq q - 1$. Let $\boldsymbol{w} \in \mathbb{F}_q^n$. We wish to find $\boldsymbol{z} \in \mathbb{F}_q^l$ such that $w_i + (\boldsymbol{zH}_0)_i \geq s_i$ for many indices i.

As the $d_0 - 2$ leftmost columns of $\boldsymbol{H}_0$ are independent, there exists an invertible matrix $\boldsymbol{T} \in \mathbb{F}_q^{l \times l}$ such that

$$\boldsymbol{TH}_0 = \begin{bmatrix} I_{d_0-2} & \boldsymbol{A} \\ \boldsymbol{0} & \boldsymbol{B} \end{bmatrix},$$

where I_{d_0-2} denotes the identity matrix of size $d_0 - 2$.

For $i \in [d_0 - 2]$, we choose $z_i = s_i - w_i$ and write

$$\boldsymbol{v} = \boldsymbol{w} + \boldsymbol{z} \cdot (I_{d_0-2} \mid \boldsymbol{A}).$$

By definition, $v_i = s_i$ for all $i \in [d_0 - 2]$.

As any $d_0 - 1$ columns of $\boldsymbol{TH}_0$ are independent, no column of $\boldsymbol{B}$ consists of only zeroes. Lemma 12 implies that there is an $\boldsymbol{\eta} \in \mathbb{F}_q^{l-d_0+2}$ such that

$$\begin{aligned} &\left|\left\{i \in [d_0 - 2, n - 1] \;\middle|\; w_i + (\boldsymbol{\eta B})_{i+d_0-2} \geq s_i\right\}\right| \\ &\geq n - d_0 + 2 - \left\lfloor \frac{1}{q} \sum_{i=d_0-2}^{n-1} s_i \right\rfloor. \end{aligned}$$

Combining this with the fact that $v_i = s_i$ for all $i \in [d_0 - 2]$, we infer that for all indices $i \in [n]$, $w_i + ((\boldsymbol{z}, \boldsymbol{\eta})\boldsymbol{TH}_0)_i \geq s_i$. □

References

1. Ball S., Blokhuis A.: A bound for the maximum weight of a linear code. SIAM J. Discrete Math. **27**(1), 575–583 (2013).

2. Belov I., Shashin A.M.: Codes that correct triple defects in memory. Problems Inf. Transmiss. **13**(4), 62–65 (1977) **(in Russian)**.
3. Borden J., Vinck A.J.: On coding for 'stuck-at' defects (Corresp.). IEEE Trans. Inf. Theory **33**(5), 729–735 (1987).
4. Chen C.L.: Linear codes for masking memory defects (Corresp.). IEEE Trans. Inf. Theory **31**(1), 105–106 (1985).
5. Dumer, I.I.: On linear defect-correcting codes. In: Proc. 1987 Int. Workshop on Convolutional Codes and Multiuser Communication, Sochi, pp. 222-225 (1987)
6. Dumer I.I.: Asymptotically optimal codes correcting memory defects of fixed multiplicity. Problemy Peredachi Informatsii **25**(4), 3–10 (1989).
7. Dumer I.I.: Asymptotically optimal linear codes correcting defects of linearly increasing multiplicity. Problemy Peredachi Informatsii **26**(2), 3–17 (1990).
8. Gleixner B., Pellizzer F., Bez R.: "Reliability Characterization of Phase Change Memory," In: 10th Annual NVMTS. IEEE **2009**, 7–11 (2009).
9. Griesmer J.H.: A bound for error-correcting codes. IBM J Res Dev **4**(5), 532–542 (1960).
10. Guruswami, V., Rurda, A., Sudan, M.: "Essential Coding Theory", March 15, (2019). Available at https://cse.buffalo.edu/faculty/atri/courses/coding-theory/book/web-coding-book.pdf
11. Hartmann C.R.P., Tzeng K.K.: Generalizations of the BCH-bound. Inf. Control **20**, 489–498 (1972).
12. Heegard C.: Partitioned linear block codes for computer memory with 'Stuck-at' defects. IEEE Trans. Inf. Theory **29**(6), 831–842 (1983).
13. Kim K., Ahn S.J.: Reliability Investigations for Manufacturable High Density PRAM. In: IEEE International Reliability Physics Symposium, 2005. Proceedings. 43rd Annual. IEEE 2005, 157–162 (2005).
14. Kuznetsov A.: Coding in a channel with generalized defects and random errors. Problems Inf. Transmiss. **21**(1), 28–34 (1985) **(in Russian)**.
15. Kuznetsov A., Tsybakov B.: for memories with defective cells. Probl. Inf. Transmiss. **10**(2), 52–60 (1974) **(in Russian)**.
16. Kuznetsov A.V., Kasami T., Yamamura S.: An error correcting scheme for defective memory. IEEE Trans. Inf. Theory **24**(6), 712–718 (1978).
17. Lee S., Jeong J.-H., Lee T.S., Kim W.M., Cheong B.-K.: A study on the failure mechanism of a phase-change memory in write/erase cycling. IEEE Electron Device Lett. **30**(5), 448–450 (2009).
18. Loeliger H.A.: Averaging Arguments for Lattices and Linear Codes. IEEE Trans. Inf. Theory **43**(6), 1767–1772 (1997).
19. Losev V.V., Konopel'ko V.K., Daryakin Y.D.: Double-and-triple-defect-correcting codes. Problems Inf. Transmiss. **14**(4), 98–101 (1978) **(in Russian)**.
20. Pirovano A., Redaelli A., Pellizzer F., Ottogalli F., Tosi M., Ielmini D., Lacaita A.L., Bez R.: Reliability study of phase-change nonvolatile memories. IEEE Trans. Device Matter Reliab. **4**(3), 422–427 (2004).
21. Roos C.: On the structure of convolutional and cyclic convolutional codes. IEEE Trans. Inf. Theory **IT–25**(6), 676–683 (1979).
22. Roth R.M.: Introduction to Coding Theory. Cambridge University Press, Cambridge (2006).
23. Sidorenko, V., Schmidt, G., Gabidulin, E., Bossert, M., Afanassiev, V.: On polyalphabetic block codes. In: IEEE Information Theory Workshop, 2005., Rotorua, New Zealand, pp. 4 pp.-, (2005) https://doi.org/10.1109/ITW.2005.1531889.
24. Solomon G.: A note on alphabet codes and fields of computation. Inform. Control **25**, 395–398 (1974).
25. Tsybakov, B.S., Gelfand, S.I., Kuznetsov, A.V., Ortyukov, S.I.: Reliable computation and reliable storage of information. In: Proc. IEEE-USSR Workshop (1975)
26. Tsybakov B.S.: "Group additive defect-correcting codes," (in Russian). Problems Inf. Transmiss. **11**(1), 111–113 (1975).
27. Tsybakov B.S.: Defects and error correction. Problems Inf. Transmiss. **11**(1), 21–30 (1975) **(in Russian)**.
28. van Lint J.H.: Introduction the Coding Theory, 2nd edn Springer, Berlin (1992) https://doi.org/10.1007/978-3-662-00174-5.
29. Wachter-Zeh A., Yaakobi E.: Codes for partially stuck-at memory cells. IEEE Trans. Inf. Theory **62**(2), 639–654 (2016).

Authors and Affiliations

Haider Al Kim[1,2] · Sven Puchinger[3] · Ludo Tolhuizen[4] · Antonia Wachter-Zeh[1]

Sven Puchinger
mail@svenpuchinger.de

Ludo Tolhuizen
ludo.tolhuizen@philips.com

Antonia Wachter-Zeh
antonia.wachter-zeh@tum.de

1 School for Computation, Information and Technology, Technical University of Munich (TUM), Munich, Germany

2 Department Electronics and Communications Engineering, University of Kufa (UoK), Kufa, Iraq

3 Hensoldt Sensors GmbH, Ulm, Germany

4 Philips Research, Eindhoven, The Netherlands