Codes for Asymmetric Limited-Magnitude Errors with Application to Multi-Level Flash Memories

Yuval Cassuto, *Member, IEEE,* Moshe Schwartz, *Member, IEEE,* Vasken Bohossian, and Jehoshua Bruck, *Fellow, IEEE*

***Abstract*—Several physical effects that limit the reliability and performance of Multilevel Flash Memories induce errors that have low magnitudes and are dominantly asymmetric. This paper studies block codes for asymmetric limited-magnitude errors over q-ary channels. We propose code constructions and bounds for such channels when the number of errors is bounded by t and the error magnitudes are bounded by ℓ. The constructions utilize known codes for symmetric errors, over small alphabets, to protect large-alphabet symbols from asymmetric limited-magnitude errors. The encoding and decoding of these codes are performed over the small alphabet whose size depends only on the maximum error magnitude and is independent of the alphabet size of the outer code. Moreover, the size of the codes is shown to exceed the sizes of known codes (for related error models), and asymptotic rate-optimality results are proved. Extensions of the construction are proposed to accommodate variations on the error model and to include systematic codes as a benefit to practical implementation.**

***Index Terms*—asymmetric limited-magnitude errors, error-correcting codes, Flash memory codes, q-ary codes, systematic codes**

I. Introduction

THE most well-studied model for error-correcting codes is the model of symmetric errors. According to this model, a symbol, taken from the code alphabet, is changed to another symbol from the same alphabet, and all such transitions are equally likely. The popularity of this model stems from both its applicability to a broad set of applications, and from the powerful construction techniques that were found to address it. In addition to the symmetric model, many other models, variations and generalizations were studied, each motivated by a behavior of practical systems or applications.

In this paper we study block codes that correct *Asymmetric Limited-Magnitude* errors. This model is parameterized by two integer parameters: t is the maximum number of symbol errors within a codeword, and ℓ is the maximal magnitude of an error. This model is motivated by error mechanisms that affect Multi-Level Flash Memory reliability and access speed.

Yuval Cassuto is with Hitachi Global Storage Technologies, 3403 Yerba Buena Rd., San Jose, CA 95135, U.S.A. (e-mail: yuval.cassuto@hitachigst.com).

Moshe Schwartz is with the Department of Electrical and Computer Engineering, Ben-Gurion University, Beer Sheva 84105, Israel (e-mail: schwartz@ee.bgu.ac.il).

Vasken Bohossian was with the Department of Electrical Engineering, California Institute of Technology, 1200 E California Blvd., Mail Code 136-93, Pasadena, CA 91125, U.S.A. (e-mail: vincent@paradise.caltech.edu).

Jehoshua Bruck is with the Department of Electrical Engineering, California Institute of Technology, 1200 E California Blvd., Mail Code 136-93, Pasadena, CA 91125, U.S.A. (e-mail: bruck@paradise.caltech.edu).

This work was supported in part by the Caltech Lee Center for Advanced Networking and by GIF grant 2179-1785.10/2007.

Flash Memory is a Non-Volatile Memory (NVM) technology that is both electrically programmable and electrically erasable. To scale the storage density of Flash memories, the *Multi-Level Flash Cell* concept is used to increase the number of stored bits in a cell [5]. Thus each Multi-Level Flash cell stores one of q levels and can be regarded as a symbol over a discrete alphabet of size q. Flash devices exhibit a multitude of complex error types and behaviors, but common to all flavors of Flash storage is the inherent asymmetry between cell programming (charge placement) and cell erasing (charge removal). This asymmetry causes significant error sources to change cell levels in one dominant direction. Moreover, many reported common Flash error mechanisms induce errors whose magnitudes (the number of level changes) are small, and independent of the alphabet size, which may be significantly larger than the typical error magnitude. Altogether, Flash errors strongly motivate the model of asymmetric limited-magnitude errors studied in this paper. In addition to the (uncontrolled) errors that challenge Flash Memory design and operation, codes for asymmetric limited-magnitude errors can be used to speed-up memory access by allowing less-precise programming schemes that introduce errors in a controlled way. While not a panacea for all Flash issues, the potential error mitigation and performance boost by asymmetric limited-magnitude codes, justify their addition, alongside other coding innovations, to the menu of Flash coding solutions.

Asymmetric limited-magnitude error-correcting codes were proposed in [1] for the special case of correcting *all* such errors within a codeword. These codes turn out to be a special case of the general construction method provided here. Previous works that treated related error models include [2],[15] and [9].

The following example illustrates the coding problem and introduces the main idea of the code construction. Suppose we have a set of 5 cells, each in one of $q = 8$ possible levels, marked by the integers $\{0, 1, \ldots, 7\}$. The design goal is now chosen to be protecting this set of cells against $t = 2$ errors of magnitude $\ell = 1$ in the upward direction. As illustrated by the sample words in Figure 1 below, if the stored levels are restricted to have either all symbols with even parity or all symbols with odd parity, the required protection is achieved. For each of the two sample codewords in row (a) of Figure 1, the channel introduces two upward errors of magnitude 1 (b). By even/odd majority, the locations of the errors are detected (c), in bold, and the original symbols are recovered by decrementing the erroneous symbols (d).

The example above is one instantiation of a general construction method that provides codes for all possible code parameters. The main strength of this method is that for any

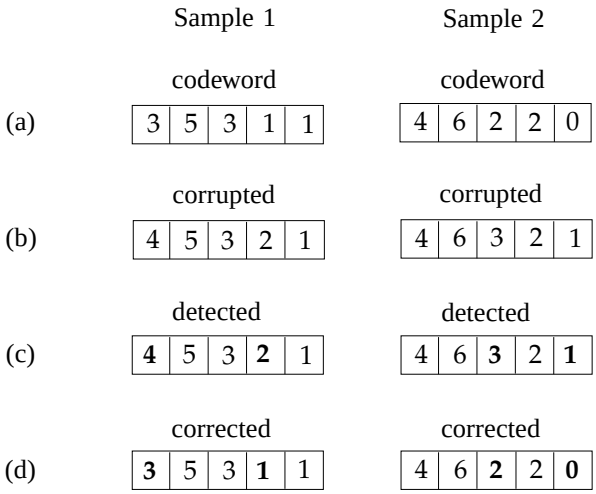


Figure 1. Example of correcting asymmetric limited-magnitude errors

target alphabet size (determined by the number of levels), asymmetric limited-magnitude error-correctability is inherited from *symmetric* error correctability of codes over alphabets of size $\ell+1$ (in the case of the example above, it is the binary repetition code). Thus a rich selection of known symmetric-error-correcting codes becomes handy to offer codes that are optimized for the asymmetric limited-magnitude channel. As a favorable by-product of the construction method, encoding and decoding of the resulting codes are performed on alphabets whose sizes depend only on ℓ, irrespective of the code alphabet (which may be much larger than ℓ). Working with both the $(\ell+1)$-ary and q-ary alphabets provides advantage in both redundancy and complexity, compared to earlier works on codes for Multilevel Flash memories [7] that employ pure q-ary constructions.

After discussing the asymmetric ℓ-limited-magnitude error model in Section II, the main code construction is presented in Section III, together with encoding and decoding procedures. Evaluation of the resulting codes is performed in Section IV, where asymptotic optimality is shown for $\ell=1$ and for a general ℓ when t grows "slowly" relative to the code length n. A more conclusive optimality is shown by constructing codes that are perfect in the asymmetric ℓ-limited-magnitude error model. In addition, Section IV compares the code sizes to sizes of codes for a related error model. Section V and Section VI discuss extensions of the code construction with motivations from practical applications. Those include the construction of systematic codes (V), and codes for simultaneous asymmetric and symmetric limited-magnitude errors (VI). Finally, section VII discusses the usage of asymmetric limited-magnitude codes in Flash devices, by showing their effectiveness in speeding up the memory write access.

II. t Asymmetric ℓ-Limited-Magnitude Error-Correcting Codes

An alphabet Q of size q is defined as the set of integers modulo q: $\{0,1,2,\ldots,q-1\}$. For a codeword $\boldsymbol{x}\in Q^n$ and a channel output $\boldsymbol{y}\in Q^n$, the definition of asymmetric limited-magnitude errors now follows.

Definition 1. *A vector of integers $\boldsymbol{e}=(e_1,\ldots,e_n)$ is called a t asymmetric ℓ-limited-magnitude error word if $|\{i:e_i\neq 0\}|\leqslant t$, and for all i, $0\leqslant e_i\leqslant \ell$. Given a codeword $\boldsymbol{x}\in Q^n$, a t asymmetric ℓ-limited-magnitude channel outputs a vector $\boldsymbol{y}\in Q^n$, such that $\boldsymbol{x}+\boldsymbol{e}=\boldsymbol{y}$, and $\boldsymbol{e}$ is a t asymmetric ℓ-limited-magnitude error word. The $+$ symbol denotes addition over the reals.*

Note that some t asymmetric ℓ-limited-magnitude error words $\boldsymbol{e}$ make $\boldsymbol{y}$ overshoot beyond the upper alphabet symbol, for which reason the restriction $\boldsymbol{y}\in Q^n$ was added. A generalization of the above definition is when we allow asymmetric errors to wrap around (from $q-1$ back to 0), whereby we interpret the $+$ symbol above as addition modulo q.

The q-ary asymmetric ℓ-limited-magnitude error model studied in this paper is a generalization of the binary asymmetric-error model studied by numerous authors (see [10] for a detailed treatment of this channel). Another generalization, proposed by Varshamov [15], studies q-ary asymmetric errors that have no magnitude limit for individual coordinates, but the sum of the error-vector elements is bounded by some integer T. When $T=t\ell$, codes for the Varshamov channel trivially correct t asymmetric ℓ-limited-magnitude errors. However, for many applications, such as Multi-Level Flash memories, the Varshamov channel may be too strong an error model. These applications can greatly benefit from the constructions presented here, which give better codes in terms of size, and also enjoy simple encoding and decoding algorithms.

The discussion of codes for the asymmetric ℓ-limited-magnitude channel-model is commenced with the definition of a distance that captures the correctability of t asymmetric ℓ-limited-magnitude errors.

Definition 2. *For $\boldsymbol{x}=(x_1,\ldots,x_n)\in Q^n$ and $\boldsymbol{z}=(z_1,\ldots,z_n)\in Q^n$, define $N(\boldsymbol{x},\boldsymbol{z})=|\{i:x_i>z_i\}|$. The distance d_ℓ between the words $\boldsymbol{x},\boldsymbol{z}$ is defined*

$$d_\ell(\boldsymbol{x},\boldsymbol{z}) = \begin{cases} n+1 & \text{if } \max_i\{|x_i-z_i|\}>\ell \\ \max(N(\boldsymbol{x},\boldsymbol{z}),N(\boldsymbol{z},\boldsymbol{x})) & \text{otherwise} \end{cases}$$

The d_ℓ distance defined above allows to determine the number of ℓ-limited-magnitude errors, correctable by a code $\mathcal{C}$.

Proposition 3. *A code $\mathcal{C}\subseteq Q^n$ can correct t asymmetric ℓ-limited-magnitude errors if and only if $d_\ell(\boldsymbol{x},\boldsymbol{z})\geqslant t+1$ for all distinct $\boldsymbol{x},\boldsymbol{z}\in\mathcal{C}$.*

Proof: A code fails to correct a t asymmetric ℓ-limited-magnitude error word if and only if there exist two distinct codewords $\boldsymbol{x}$,$\boldsymbol{z}$ and two t asymmetric ℓ-limited-magnitude error words $\boldsymbol{e}$,$\boldsymbol{f}$, such that $\boldsymbol{x}+\boldsymbol{e}=\boldsymbol{z}+\boldsymbol{f}$, or equivalently, $\boldsymbol{x}-\boldsymbol{z}=\boldsymbol{f}-\boldsymbol{e}$.

($\Leftarrow$) Assume that for a pair $\boldsymbol{x}$,$\boldsymbol{z}$, $d_\ell(\boldsymbol{x},\boldsymbol{z})\geqslant t+1$. Then at least one of the following holds:

1) $N(\boldsymbol{x},\boldsymbol{z})>t$ or $N(\boldsymbol{z},\boldsymbol{x})>t$
2) $|x_i-z_i|>\ell$ for at least one index $i\in\{1,\ldots,n\}$.

Case 1 implies that $\boldsymbol{f}-\boldsymbol{e}$ has either more than t positive elements or more than t negative elements, none of which is possible by the definition of the error vectors $\boldsymbol{e},\boldsymbol{f}$.

Case 2 implies that for some i, either $e_i > \ell$ or $f_i > \ell$, both impossible by the definition of $\boldsymbol{e},\boldsymbol{f}$.

Since the same arguments apply to any $\boldsymbol{x},\boldsymbol{z}$ in the code, it necessarily corrects all possible t asymmetric ℓ-limited-magnitude errors.

($\Rightarrow$) Assume there exist a pair of codewords $\boldsymbol{x},\boldsymbol{z}$, for which $d_\ell(\boldsymbol{x},\boldsymbol{z}) \leqslant t < n$. Then both $N(\boldsymbol{x},\boldsymbol{z}) \leqslant t$ and $N(\boldsymbol{z},\boldsymbol{x}) \leqslant t$ are true, and $|x_i - z_i| \leqslant \ell$ at all positions i. In that case we can set $f_i = x_i - z_i$ at all positions i such that $x_i > z_i$ and $e_i = z_i - x_i$ at all positions i such that $z_i > x_i$. With zeros at all other positions, such $\boldsymbol{e},\boldsymbol{f}$ satisfy $\boldsymbol{x}-\boldsymbol{z} = \boldsymbol{f}-\boldsymbol{e}$ without violating the conditions of t asymmetric ℓ-limited-magnitude errors. ∎

The following Corollary states that the same distance d_ℓ captures the error *detection* capability of the code, unless no codeword is greater than or equal to another codeword on every coordinate, in which case the code detects all asymmetric ℓ-limited-magnitude errors. Detailed treatment of joint correction/detection is beyond the scope of this paper, but these properties can be analyzed using similar methods to binary asymmetric codes [2].

Corollary 4. *Unless*

$$\forall(\boldsymbol{x}\in\mathcal{C},\boldsymbol{z}\in\mathcal{C}) : \min(N(\boldsymbol{x},\boldsymbol{z}),N(\boldsymbol{z},\boldsymbol{x})) > 0, \qquad (1)$$

a code $\mathcal{C} \subseteq Q^n$ can detect *t asymmetric ℓ-limited-magnitude errors if and only if $d_\ell(\boldsymbol{x},\boldsymbol{z}) \geqslant t+1$ for all distinct $\boldsymbol{x},\boldsymbol{z}\in\mathcal{C}$. If* (1) *is met, then the code detects* all *asymmetric ℓ-limited-magnitude errors.*

Proof: The proof is essentially the same as of Proposition 3, only with $\boldsymbol{e} = 0$. If (1) is not met, then for a codeword pair with $N(\boldsymbol{z},\boldsymbol{x}) = 0$ the equality $\boldsymbol{x}-\boldsymbol{z} = \boldsymbol{f}$ provides the same necessary and sufficient conditions as the former $\boldsymbol{x}-\boldsymbol{z} = \boldsymbol{f}-\boldsymbol{e}$. If, on the other hand, (1) is met, i.e. all codeword pairs have both $N(\boldsymbol{x},\boldsymbol{z}) > 0$ and $N(\boldsymbol{z},\boldsymbol{x}) > 0$, then no all-positive $\boldsymbol{f}$ can equal $\boldsymbol{x}-\boldsymbol{z}$, and the code detects all asymmetric ℓ-limited-magnitude errors. ∎

Although the asymmetric ℓ-limited-magnitude distance-measure d_ℓ is not a metric (since the triangle inequality does not hold), it still provides a necessary and sufficient condition for the correctability of asymmetric ℓ-limited-magnitude errors. In subsequent sections, it will be used both to prove the correction capability of code constructions, and to obtain upper bounds on the size of codes.

III. Construction of t Asymmetric ℓ-Limited-Magnitude Error-Correcting Codes

We now provide the main construction of the paper. We note that the general idea of the basic code construction below, restricted to binary codes ($q' = 2$), has appeared in Construction A of [11], for a different application (sphere packings in Euclidean spaces). The same work also considered the encoding and decoding of Construction A codes, which appear here in sub-sections III-B and III-C for the general case of $q' \geqslant 2$. For notational convenience, given $\boldsymbol{x} = (x_1,\ldots,x_n)$, the vector $(x_1 \bmod q', x_2 \bmod q', \ldots, x_n \bmod q')$ will be denoted by $\boldsymbol{x} \bmod q'$. To obtain a code over alphabet Q that corrects t or less asymmetric errors of ℓ-limited-magnitude, one can use codes for symmetric errors over small alphabets as follows.

Construction 1. *Let Σ be a code over the alphabet Q' of size $q' = \ell+1$. The code $\mathcal{C}$ over the alphabet Q of size q ($q > \ell+1$) is defined as*

$$\mathcal{C} = \left\{\boldsymbol{x} = (x_1,\ldots,x_n)\in Q^n : \boldsymbol{x} \bmod q' \in \Sigma\right\}. \qquad (2)$$

In other words, the codewords of $\mathcal{C}$ are the subset of the words of Q^n that are mapped to codewords of Σ, when their symbols are reduced modulo $q' = \ell+1$.

Codes obtained by Construction 1 have the following error-correction capability.

Theorem 5. *$\mathcal{C}$ corrects t asymmetric ℓ-limited-magnitude errors if Σ corrects t symmetric errors. If $q > 2\ell$,[1] the converse is true as well.*

Proof: The proof proceeds by showing that any pair of distinct codewords $\boldsymbol{x},\boldsymbol{z}\in\mathcal{C}$ is at d_ℓ distance of at least $t+1$ apart. By Proposition 3, this would conclude that $\mathcal{C}$ corrects all t asymmetric ℓ-limited-magnitude errors. We distinguish between two cases:

In the first case $\boldsymbol{x} \bmod q' = \boldsymbol{z} \bmod q'$. Since $\boldsymbol{x} \neq \boldsymbol{z}$, this implies that for at least one index $i\in\{1,\ldots,n\}$, $|x_i - z_i| > \ell$, settling their d_ℓ distance to be $n+1$.

In the other case, $\boldsymbol{x} \bmod q' \neq \boldsymbol{z} \bmod q'$. The fact that Σ has minimum Hamming distance of at least $2t+1$ implies that $\boldsymbol{x}$ and $\boldsymbol{z}$ differ in at least $2t+1$ locations and thus, in particular, $\max(N(\boldsymbol{x},\boldsymbol{z}),N(\boldsymbol{z},\boldsymbol{x})) \geqslant t+1$.

For the converse, if Σ does not correct all t symmetric errors, then there exists a quadruple $(\boldsymbol{\chi}\in\Sigma,\boldsymbol{\zeta}\in\Sigma,\boldsymbol{e},\boldsymbol{f})$, such that $\boldsymbol{\chi}+\boldsymbol{e} \equiv \boldsymbol{\zeta}+\boldsymbol{f} \pmod{q'}$, and $\boldsymbol{e},\boldsymbol{f}$ are t asymmetric ℓ-limited-magnitude error vectors. Therefore, the vectors $\boldsymbol{x} = \boldsymbol{\chi} + q'\cdot\Delta(\boldsymbol{\zeta}+\boldsymbol{f}-\boldsymbol{\chi}-\boldsymbol{e})$ and $\boldsymbol{z} = \boldsymbol{\zeta} + q'\cdot\Delta(\boldsymbol{\chi}+\boldsymbol{e}-\boldsymbol{\zeta}-\boldsymbol{f})$, (where $\Delta(\boldsymbol{v})$ is a vector with ones where $v_i > 0$ and zeros elsewhere), are codewords of $\mathcal{C}$ and they satisfy $\boldsymbol{x}+\boldsymbol{e} = \boldsymbol{z}+\boldsymbol{f}$. Since $q > 2\ell$, the last sum is a valid channel output. We conclude that there exists an uncorrectable error word for $\mathcal{C}$, and the converse follows. ∎

Construction 1 is clearly useful as it leverages the comprehensively studied theory of codes for symmetric errors, to obtain codes for asymmetric limited-magnitude errors. However, Construction 1 is a special case of the following construction.

Construction 1A. *Let Σ be a code over the alphabet Q' of size q'. The code $\mathcal{C}$ over the alphabet Q of size q ($q > q' > \ell$) is defined as*

$$\mathcal{C} = \left\{\boldsymbol{x} = (x_1,\ldots,x_n)\in Q^n : \boldsymbol{x} \bmod q' \in \Sigma\right\}. \qquad (3)$$

The relationship between $\mathcal{C}$ and Σ in the general case are summarized below. The proof is almost identical to that of Theorem 5.

[1] The biggest motivation to use asymmetric limited-magnitude codes is when $q \gg \ell$, so $q > 2\ell$ is a reasonable condition.

Theorem 6. *$\mathcal{C}$ corrects t asymmetric ℓ-limited-magnitude errors if Σ corrects t asymmetric ℓ-limited-magnitude errors with wrap-around. If $q \geqslant q' + \ell$, the converse is true as well.*

Remark: If $q' \mid q$ then $\mathcal{C}$ corrects t asymmetric ℓ-limited-magnitude errors *with wrap-around* for Σ with the same properties as above.

It is easy to see how Construction 1 is a special case of Construction 1A. When $q' = \ell + 1$, an asymmetric ℓ-limited-magnitude error with wrap-around is equivalent to a symmetric error (with no magnitude limit).

A. Discussion and Analysis of Code Constructions

The size of the code $\mathcal{C}$ is bounded from below and from above by the following theorem.

Theorem 7. *The number of codewords in the code $\mathcal{C}$ is bounded by the following inequalities,*

$$\left\lfloor \frac{q}{q'} \right\rfloor^n \cdot |\Sigma| \leqslant |\mathcal{C}| \leqslant \left\lceil \frac{q}{q'} \right\rceil^n \cdot |\Sigma| \,. \tag{4}$$

Proof: Let $\boldsymbol{\chi} = (\chi_1, \ldots, \chi_n)$ be a codeword of Σ. A valid codeword of $\mathcal{C}$ can be obtained by replacing each χ_i by any element of the set $\{x \in Q : x = \chi_i \pmod{q'}\}$. The size of this set is $\lceil q/q' \rceil$ if $\chi_i < q \bmod q'$ and $\lfloor q/q' \rfloor$ otherwise. Thus for any code Σ, the lower and upper bounds above follow. ∎

In the special case when $q' = 2$, the size of $\mathcal{C}$ can be obtained exactly from the weight enumerator of Σ.

Theorem 8. *Let $q' = 2$ and Σ be a code over $Q' = \{0, 1\}$. Then the size of the code $\mathcal{C}$, as defined in* (3)*, is given by*

$$|\mathcal{C}| = \sum_{w=0}^{n} A_w \left\lceil \frac{q}{2} \right\rceil^{n-w} \left\lfloor \frac{q}{2} \right\rfloor^{w}$$

where A_w is the number of codewords of Σ with Hamming weight w.

Proof: When $2 \mid q$, the right hand side equals $(q/2)^n \cdot |\Sigma|$, as the matching lower and upper bounds of (4) predict. When $2 \nmid q$, a 0 in $\boldsymbol{\chi}$ can be replaced by $\lceil q/2 \rceil$ different symbols of Q and a 1 in $\boldsymbol{\chi}$ can be replaced by $\lfloor q/2 \rfloor$ different symbols. Using the weight enumerator of Σ we obtain the exact value for the size of $\mathcal{C}$ above. ∎

This theorem can be extended to $q' > 2$, but in such cases knowing the weight distribution of Σ does not suffice, and more detailed enumeration of the code is needed for an exact count.

The ℓ-AEC codes suggested in [1], that correct all asymmetric ℓ-limited-magnitude errors, can also be regarded as a special case of this construction method. To show that, let $\mathbf{0}$ be the trivial length n code over the alphabet Q' of size $q' = \ell + 1$, that contains only the all-zero codeword. Define

$$\begin{aligned} \mathcal{C} &= \left\{ \boldsymbol{x} \in Q^n : \boldsymbol{x} \bmod q' \in \mathbf{0} \right\} \\ &= \left\{ \boldsymbol{x} \in Q^n : x_i \equiv 0 \bmod q' \text{ for } i = 1, 2, \ldots, n \right\} \quad [1]. \end{aligned}$$

Since $\mathbf{0}$ can correct $t = n$ symmetric errors, $\mathcal{C}$ can correct $t = n$ asymmetric ℓ-limited-magnitude errors.

B. Decoding

The main construction of this paper (Construction 1) reduces the problem of constructing asymmetric ℓ-limited-magnitude error-correcting codes, to the problem of constructing codes for symmetric errors. The correction capability of the code constructions was proved earlier in the section using arguments on their minimum d_ℓ distance, arguments that have a non-algorithmic character. We next show that a similar reduction applies also to the algorithmic problem of efficiently decoding asymmetric ℓ-limited-magnitude error-correcting codes.

In the following, we describe how, given a decoding algorithm for the code Σ, one can obtain a decoder for the code $\mathcal{C}$, that has essentially the same decoding complexity, with only a few additional simple arithmetic operations. The decoding procedure herein refers to the more general Construction 1A, and it clearly applies to the special case of Construction 1 ($q' = \ell + 1$).

Let $\boldsymbol{x} = (x_1, \ldots, x_n) \in \mathcal{C}$ be a codeword and $\boldsymbol{y} = (y_1, \ldots, y_n) \in Q^n$ be the channel output when up to t asymmetric ℓ-limited-magnitude errors have occurred. Denote the corresponding Σ codeword by $\boldsymbol{\chi} = \boldsymbol{x} \bmod q'$, and also define $\boldsymbol{\psi} = \boldsymbol{y} \bmod q'$ and $\boldsymbol{\epsilon} = (\boldsymbol{\psi} - \boldsymbol{\chi}) \pmod{q'}$. First we observe that since $q' > \ell$, if $0 \leqslant y_i - x_i \leqslant \ell$ then $y_i - x_i = (y_i - x_i) \bmod q'$. Using the simple modular identity

$$\begin{aligned} (y_i - x_i) \bmod q' &= (y_i \bmod q' - x_i \bmod q') \bmod q' \\ &= (\psi_i - \chi_i) \bmod q' = \epsilon_i, \end{aligned}$$

we get that $y_i - x_i = \epsilon_i$, and in particular, if $0 \leqslant y_i - x_i \leqslant \ell$, then $0 \leqslant \epsilon_i \leqslant \ell$. In other words, if the codeword $\boldsymbol{x}$ over Q suffered an asymmetric ℓ-limited-magnitude error at location i, then the codeword $\boldsymbol{\chi}$ over Q' suffered an asymmetric ℓ-limited-magnitude error with wrap-around at the same location i, and with the same magnitude. Given at most t asymmetric ℓ-limited-magnitude errors with wrap-around, a decoder for Σ can recover $\boldsymbol{\epsilon}$ from $\boldsymbol{\psi}$. Thus, the equality $y_i - x_i = \epsilon_i$ allows the same decoder to recover $\boldsymbol{x}$ from $\boldsymbol{y}$.

A schematic decoder of an asymmetric ℓ-limited-magnitude error-correcting code $\mathcal{C}$ that uses a decoder for a symmetric error-correcting code Σ is given in Figure 2. Given a channel output $\boldsymbol{y} \in Q^n$, the decoder takes the symbol-wise modulo q' of $\boldsymbol{y}$ to obtain $\boldsymbol{\psi} \in Q'^n$. Then a decoder for Σ is invoked with the input $\boldsymbol{\psi}$ and an error estimate $\hat{\boldsymbol{\epsilon}}$ is obtained such that $\hat{\boldsymbol{\chi}} + \hat{\boldsymbol{\epsilon}} \equiv \boldsymbol{\psi} \pmod{q'}$, and $\hat{\boldsymbol{\chi}}$ is a codeword of Σ within the correction radius of the decoder for Σ. Note that the codeword estimate $\hat{\boldsymbol{\chi}}$ is discarded and not used for the decoding of $\mathcal{C}$. Finally, $\hat{\boldsymbol{\epsilon}}$ is subtracted from $\boldsymbol{y}$ to obtain the codeword estimate $\hat{\boldsymbol{x}} \in \mathcal{C}$.

C. Encoding

Construction 1 (and 1A) define the code $\mathcal{C}$ as a subset of Q^n, without specifying how information symbols are mapped to codewords. There are many ways to map information to codewords of $\mathcal{C}$, and the simplest one, that applies to any q, q' such that $q \mid q'$, is detailed below. For an alphabet of size $q = A \cdot q'$, where A and q' are integers, information is

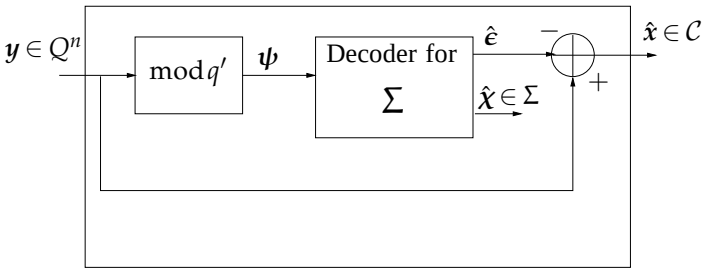


Figure 2. Decoding asymmetric ℓ-limited-magnitude error-correcting codes

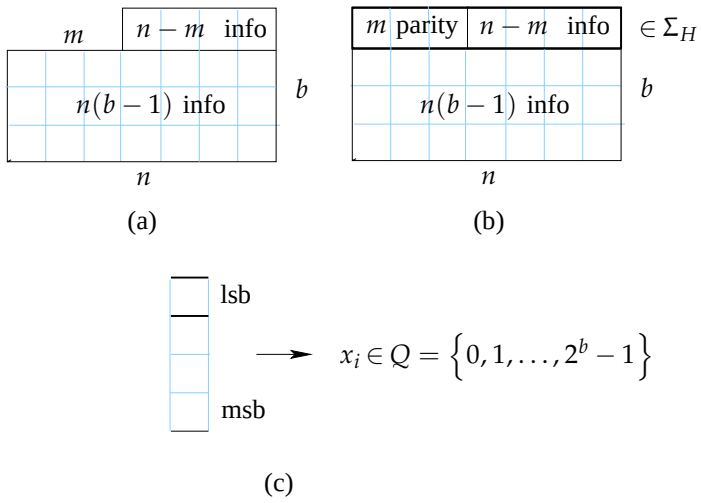


Figure 3. Encoding Procedure for $\mathcal{C}_H$

mapped to a length n codeword of $\mathcal{C}$ as follows: n symbols, $(a_1, \ldots, a_n)$, over the alphabet of size A are set as pure information symbols. Additionally, k information symbols over the alphabet of size q' are input to an encoder of Σ to obtain n symbols, $(\chi_1, \ldots, \chi_n)$, over the same alphabet. Finally, each code symbol x_i over Q is calculated by $a_i \cdot q' + \chi_i$.

Other encoding functions can map information symbols to codewords of $\mathcal{C}$ in a different way than the simple encoding function above. Different mappings with good properties are discussed in Section V and Section VI.

Example 1 now attempts to convey the main ideas of the encoding and decoding of asymmetric ℓ-limited-magnitude error-correcting codes.

Example 1. *Let Σ_H be the binary[2] Hamming code of length $n = 2^m - 1$, for some integer m. First we define the code $\mathcal{C}_H$ in the way of Construction* 1*.*

$$\mathcal{C}_H = \{\boldsymbol{x} = (x_1, \ldots, x_n) \in Q^n : \boldsymbol{x} \bmod 2 \in \Sigma_H\}.$$

By the properties of Σ_H, the code $\mathcal{C}_H$ corrects a single asymmetric $\ell = 1$ limited-magnitude error. When the code alphabet size is $q = 2^b$, for some integer b, the code $\mathcal{C}_H$, whose size equals $|\mathcal{C}_H| = A^n \cdot q'^{n-m} = 2^{(b-1)n} \cdot 2^{n-m} = 2^{nb-m}$ by equation (4)*, admits a simple function from $nb - m$ information bits to codewords of $\mathcal{C}_H$ over Q, as illustrated in Figure* 3 *below. In Figure* 3 *(a), $nb - m$ information bits are input to the encoder. The encoder then uses a binary Hamming encoder to encode $n - m$ of the information bits into a length n Hamming codeword (Figure* 3 *(b)). Finally, in Figure* 3 *(c), each q-ary symbol of the codeword $\boldsymbol{x} \in \mathcal{C}_H$ is constructed from b bits using the usual binary-to-integer conversion, the top row being the least-significant bits of $x_i \in Q$.*

Decoding is carried out by using a Hamming decoder on the top row to find the limited-magnitude error location and magnitude (for binary Hamming codes the magnitude is always 1*). The top row word is* not *corrected by the Hamming decoder, but rather the error magnitude is subtracted from the Q-ary word $\boldsymbol{y}$ to obtain a decoded codeword. To recover the information bits after decoding, the Q symbols are converted back to bits in the usual way, and the m parity bits are discarded.*

[2]Non-binary Hamming codes can be used as well when $\ell > 1$.

IV. Optimality of the Code Construction and Comparison to Related Codes

A. Perfect Codes

For some parameters, the codes constructed in the previous section are the best conceivable ones for the asymmetric ℓ-limited-magnitude error model. These codes are *perfect* codes in the sense that they attain the sphere-packing bound for asymmetric ℓ-limited-magnitude errors. The q-ary symmetric sphere-packing bound is first generalized to asymmetric ℓ-limited-magnitude errors (with wrap-around), and then it is shown that asymmetric ℓ-limited-magnitude error-correcting codes that meet this bound with equality can be obtained by using other known perfect codes, e.g., perfect codes in the Hamming metric.

Theorem 9. *If $\mathcal{C}$ is a t asymmetric ℓ-limited-magnitude (with wrap-around) error-correcting code, of length n over an alphabet of size q, then*

$$|\mathcal{C}| \cdot \sum_{i=0}^{t} \binom{n}{i} \ell^i \leqslant q^n. \tag{5}$$

Proof: The proof is essentially the same as the proof of the q-ary sphere-packing bound for symmetric errors [8, Ch.1], with ℓ replacing $q - 1$ in the sum. ∎

Perfect t asymmetric ℓ-limited-magnitude error-correcting codes are obtained through the following proposition.

Proposition 10. *If there exists a perfect asymmetric ℓ-limited-magnitude code over an alphabet of size q', then there exists a perfect asymmetric ℓ-limited-magnitude code with the same length, over an alphabet of any size q, such that $q' \mid q$, that corrects the same number of errors.*

Proof: Let $\mathcal{C}$ and Σ be as in Construction 1A. We first substitute the expression for the code size from (4) into the left side of the sphere packing bound

$$|\mathcal{C}| \cdot \sum_{i=0}^{t} \binom{n}{i} \ell^i = \left(\frac{q}{q'}\right)^n \cdot |\Sigma| \cdot \sum_{i=0}^{t} \binom{n}{i} \ell^i.$$

If the code Σ over the alphabet of size q' is perfect, then its size satisfies

$$|\Sigma| \cdot \sum_{i=0}^{t} \binom{n}{i} \ell^i = q'^n$$

Substituting the latter into the former we get

$$|\mathcal{C}| \cdot \sum_{i=0}^{t} \binom{n}{i} \ell^i = \left(\frac{q}{q'}\right)^n \cdot q'^n = q^n,$$

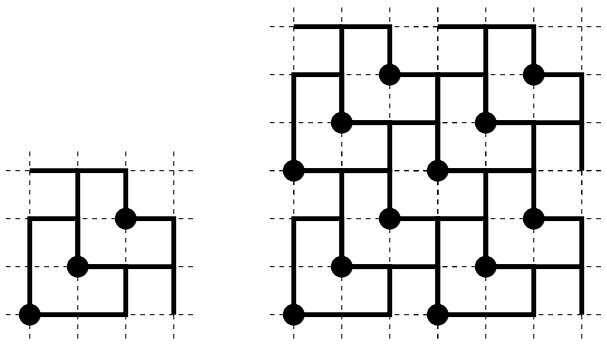
which completes the proof. ∎

Alternatively, perfect codes are codes which induce a partition of the space into error spheres. As was already noted, when $q' = \ell + 1$, the t asymmetric ℓ-limited-magnitude error sphere coincides with the Hamming metric t symmetric error sphere. Thus, taking Σ to be a perfect code in the Hamming metric (e.g., Hamming or Golay codes), produces perfect asymmetric ℓ-limited-magnitude error-correcting codes over an alphabet of size q, where $q' \mid q$.

Other perfect codes may exist even when $q' \neq \ell + 1$. For example, when $t = 1$, the asymmetric ℓ-limited-magnitude error sphere is the semi-cross examined by Stein in [14].

One may wonder if any *inherently* new perfect code is produced by Construction 1A. The answer, unfortunately, is no: Construction 1A simply takes translations of the tiling provided by the base code Σ to accommodate for the larger alphabet. This is depicted in the following example.

Example 2. *Let Σ be the perfect ternary length $n = 2$ code capable of correcting one asymmetric 1-limited-magnitude error, $\Sigma = \{00, 11, 22\}$. The code induces a tiling of $\mathbb{Z}_3^2$ with the error sphere, and is shown in Figure 4. Since this tiling is with wrap-around, it also induces a natural tiling with wrap-around of $\mathbb{Z}_{3k}^2$ for every $k \in \mathbb{N}$. Specifically, for $\mathcal{C}$, the code over an alphabet of size 6 produced from Σ by Construction 1A, the resulting tiling is also shown in Figure 4.*



(a) (b)

Figure 4. In Example 2, the tilings induced by (a) the code Σ, and (b) the code $\mathcal{C}$

B. Asymptotic Optimality of Construction 1

The implication of Construction 1 is that "large" codes for symmetric errors over an alphabet of size $\ell + 1$ imply "large" codes for asymmetric ℓ-limited-magnitude errors over any larger alphabet. Showing the reverse implication, namely, that "large" codes for asymmetric ℓ-limited-magnitude errors imply "large" codes for symmetric errors, would conclude that Construction 1 is optimal. Optimality is achieved in this case since given the "large" code for symmetric errors implied by the reverse direction, Construction 1 can be invoked to yield code of the same size as the original "large" code for asymmetric ℓ-limited-magnitude errors. The purpose of this subsection is to show that asymptotically, Construction 1 gives the largest possible codes for asymmetric ℓ-limited-magnitude errors.

Definition 11. *Define the rate R of a code $\mathcal{C}$ of length n over an alphabet of size q as*

$$R = \frac{1}{n} \log_q |\mathcal{C}|$$

where $|\mathcal{C}|$ is the number of codewords in $\mathcal{C}$.

Theorem 12. *If $\tilde{\mathcal{C}}$ is a t asymmetric ℓ-limited-magnitude error-correcting code with rate R and block-length n that tends to infinity, then*

1) *When $\ell = 1$ and for arbitrary t, there exists a code $\mathcal{C}$, constructed by Construction 1, with rate of at least R.*
2) *For general ℓ and for $t = o(n/\log n)$ (i.e., $\lim_{n\to\infty} t \log n/n = 0$, or in words, t has a slower asymptotic growth compared to $n/\log n$), there exists a code $\mathcal{C}$, constructed by Construction 1, with rate of at least R.*

Proof: We first introduce the following notation. Let $\mathsf{A}\ell\mathsf{M}_a(n,t)$ be the size of the largest length n code that corrects t asymmetric ℓ-limited-magnitude errors over an alphabet of size a. Let $\mathsf{Asym}_a(n,t)$ be the size of the largest length n code that corrects t asymmetric errors (symbols change only in the upward direction, with no magnitude limit), over an alphabet of size a. Finally, let $\mathsf{S}_a(n,t)$ be the size of the largest length n code that corrects t symmetric errors, over an alphabet of size a. $\mathsf{S}_a(n,t)$ used here is a replacement of the more commonly used $A_a(n,d)$ [8, Ch.2], whereby the parameter d stands for the minimum Hamming distance of the code instead of the number of correctable symmetric errors (therefore $\mathsf{S}_a(n,t) = A_a(n,2t+1)$).

To avoid the excessive use of the $\lceil\cdot\rceil$ operator, assume that $(\ell+1) \mid q$. The set of all q^n words over the alphabet of size q is partitioned by the quotient group $\mathbb{Z}_q^n/\mathbb{Z}_{\ell+1}^n$ into $q^n/(\ell+1)^n$ subsets, each of size $(\ell+1)^n$. In other words, each subset contains a single word whose symbol-wise modulo $\ell+1$ equals the all zero vector. In addition to this vector, the subset contains the sum of that vector with all $(\ell+1)^n - 1$ non-zero vectors over the alphabet of size $\ell+1$. Each subset has the property that no two words within it differ in any coordinate by more than ℓ. A sample such partition for $n = 2$, $q = 4$ and $\ell = 1$ is given below.

0	0	0	2	2	0	2	2
0	1	0	3	2	1	2	3
1	0	1	2	3	0	3	2
1	1	1	3	3	1	3	3

This property is equivalent to having $d_\ell(\boldsymbol{x}, \boldsymbol{z}) < n+1$ for every $\boldsymbol{x}, \boldsymbol{z}$ in the subset.

Suppose there is a code $\tilde{\mathcal{C}}$ that corrects t asymmetric ℓ-limited-magnitude errors. Then there exists at least one subset, with at least $\left|\tilde{\mathcal{C}}\right|(\ell+1)^n/q^n$ codewords of $\tilde{\mathcal{C}}$. Since any two codewords $\boldsymbol{x},\boldsymbol{z}$ in that subset satisfy $d_\ell(\boldsymbol{x},\boldsymbol{z}) < n+1$, each such pair has to satisfy $\max(N(\boldsymbol{x},\boldsymbol{z}),N(\boldsymbol{z},\boldsymbol{x})) > t$. In other words, the codewords of $\tilde{\mathcal{C}}$ that belong to the same subset, form a code that corrects t asymmetric errors with no magnitude limit of size at least $\left|\tilde{\mathcal{C}}\right|(\ell+1)^n/q^n$. Without loss of generality, assume the subset with the most codewords is the one that contains the all zero codeword. Generality is maintained since neither $N(\boldsymbol{x},\boldsymbol{z})$ nor $N(\boldsymbol{z},\boldsymbol{x})$ are changed when a constant vector is subtracted from both $\boldsymbol{x}$ and $\boldsymbol{z}$. Consequently, the codewords of this subset imply the existence of a code over an alphabet of size $\ell+1$ that corrects t asymmetric errors with no magnitude limit. Formally,

$$\mathsf{Asym}_{\ell+1}(n,t) \geqslant \left(\tfrac{\ell+1}{q}\right)^n \mathsf{A}\ell\mathsf{M}_q(n,t).$$

On the other hand, Construction 1 and Theorem 7 provide the following lower bound on $\mathsf{A}\ell\mathsf{M}_q(n,t)$:

$$\mathsf{A}\ell\mathsf{M}_q(n,t) \geqslant (\tfrac{q}{\ell+1})^n \mathsf{S}_{\ell+1}(n,t)$$

Combining the lower and upper bounds we obtain

$$\mathsf{S}_{\ell+1}(n,t) \leqslant \left(\tfrac{\ell+1}{q}\right)^n \mathsf{A}\ell\mathsf{M}_q(n,t) \leqslant \mathsf{Asym}_{\ell+1}(n,t) \quad (6)$$

which is consistent with the trivial inequality $\mathsf{S}_{\ell+1}(n,t) \leqslant \mathsf{Asym}_{\ell+1}(n,t)$ (any code for symmetric errors is also a code for asymmetric errors). The proof of the theorem is achieved by bounding the gap between $\mathsf{S}_{\ell+1}(n,t)$ and $\mathsf{Asym}_{\ell+1}(n,t)$ using the following lemmas.

Lemma 13. *[4]:* $\mathsf{S}_2(n,t) \geqslant \frac{1}{t+1}\mathsf{Asym}_2(n,t)$.

Proof: See [10]. ∎

Lemma 14. $\mathsf{S}_{\ell+1}(n,t) \geqslant \frac{1}{(n\ell)^{2t}}\mathsf{Asym}_{\ell+1}(n,t)$.

Proof: We will show that a code for symmetric errors can be obtained from a code for asymmetric errors by expurgating all but at least a $1/(n\ell)^{2t}$ fraction of codewords of the asymmetric-error-correcting code.

Any two codewords in a t asymmetric-error-correcting code have Hamming distance of at least $t+1$. Any two codewords in a t symmetric-error-correcting code have Hamming distance of at least $2t+1$. The number of words (and in particular, an upper bound on the number of codewords) that are at distance between $t+1$ and $2t$ from a codeword of a t asymmetric-error-correcting code is

$$\sum_{i=t+1}^{2t}\binom{n}{i}\ell^i = \ell^t\sum_{i=1}^{t}\binom{n}{t+i}\ell^i.$$

Since $\binom{n}{t+i} < n^{t+i}/t$,

$$\ell^t\sum_{i=1}^{t}\binom{n}{t+i}\ell^i < (n\ell)^{2t},$$

and thus expurgating all but at least $1/(n\ell)^{2t}$ of the codewords, yields a code for t symmetric errors:

$$\mathsf{S}_{\ell+1}(n,t) > \frac{1}{(n\ell)^{2t}}\mathsf{Asym}_{\ell+1}(n,t).$$

∎

Combining Lemma 13 with (6), for $\ell=1$ we obtain

$$\left(\frac{q}{2}\right)^n \mathsf{S}_2(n,t) \leqslant \mathsf{A}\ell\mathsf{M}_q(n,t) \leqslant n\left(\frac{q}{2}\right)^n \mathsf{S}_2(n,t).$$

While Lemma 14 end (6) imply, for general ℓ,

$$(\tfrac{q}{\ell+1})^n\mathsf{S}_{\ell+1}(n,t) \leqslant \mathsf{A}\ell\mathsf{M}_q(n,t) \leqslant (n\ell)^{2t}(\tfrac{q}{\ell+1})^n\mathsf{S}_{\ell+1}(n,t).$$

Taking the logarithm, dividing by n and taking the limit $n\to\infty$, the upper and lower bounds of $\mathsf{A}\ell\mathsf{M}_q(n,t)$ are identical for both $\ell=1$ and for general ℓ (under the restrictions on t of part 2 of the theorem). Hence asymmetric ℓ-limited-magnitude codes obtained from symmetric codes by Construction 1 are asymptotically optimal. ∎

C. Comparison to Varshamov Codes

Prior to this paper's introduction of the t asymmetric ℓ-limited-magnitude error model, the closest error model that achieves this correction capability is the q-ary asymmetric-error model proposed by Varshamov [15]. In particular, the known codes for the Varshamov channel are better than known codes for symmetric channels. According to the Varshamov model, parameterized by an integer parameter T, if a vector $\boldsymbol{x}=(x_1,\ldots,x_n)$ over Q^n is transmitted, the channel output is a vector $\boldsymbol{x}+\boldsymbol{e}$ over Q^n, such that $e_i\geqslant 0$ and $\sum_{i=1}^n e_i \leqslant T$ (the addition and summation are over the reals). When $T=t\ell$, a T error-correcting code for the Varshamov channel is also a t asymmetric ℓ-limited-magnitude error-correcting code. Since the $T=t\ell$ Varshamov channel allows errors that are not allowed by the t asymmetric ℓ-limited-magnitude channel (i.e., errors with high magnitudes, which are unlikely in the target application), we expect the code constructions of this paper to yield better codes compared to the best known Varshamov codes. This section thus compares between sizes of codes that are obtained using Construction 1, and lower bounds, provided in [13], on the sizes of various Varshamov codes. This comparison is incomplete since it only discusses the *sizes* of the codes. Evidently, our t asymmetric ℓ-limited-magnitude codes enjoy efficient encoding and decoding procedures, a property which Varshamov codes are not known to have in general. We also do not discuss the restrictions on the block sizes n of the code constructions, in order to avoid overloading the discussion with secondary details.

1) Comparison for $\ell=1$: When the asymmetric errors have a magnitude limit of $\ell=1$, we compare the codes of Construction 1 to Varshamov codes with $T=t$. When $t=1$, the two error models are identical and both constructions yield (different) codes that are perfect in that metric, whose sizes are $q^n/(n+1)$. When $t=2$ Varshamov codes are known to have $q^n/(n^2+n+1)$ codewords, while using the (punctured) Preparata codes [12, Ch.15] in Construction 1 gives $2q^n/(n+1)^2$, roughly twice as many codewords. For a general t, there exist Varshamov codes with sizes $q^n/(n+1)^t$. If we apply Construction 1 with BCH codes with designed distance $2t+1$, we get the same code size. However, using the Goppa codes [12, Ch.12] instead, is possibly superior to Varshamov codes with q^n/n^t codewords.

2) Comparison for a General ℓ: While for $\ell = 1$ the advantage of the codes for asymmetric ℓ-limited-magnitude errors, in terms of the code sizes is small, for larger ℓ values these codes are significantly larger than Varshamov codes. Even if we only use $(\ell+1)$-ary BCH codes in Construction 1, codes of sizes $q^n/(n+1)^{t'}$ are obtained, where $t' = 2t\ell/(\ell+1)$. Comparing to $q^n/(n+1)^{t\ell}$ of Varshamov codes shows a significant advantage to the favor of Construction 1.

V. Systematic Asymmetric Limited-Magnitude Error-Correcting Codes

All its advantages notwithstanding, Construction 1 suffers the shortcoming of not admitting a systematic representation over Q. A code $\mathcal{C}$ over an alphabet Q is said to be in systematic form if its coordinates $\{x_1, \ldots, x_n\}$ can be partitioned into an information set $I = \{x_1, \ldots, x_k\}$ and a parity set $P = \{x_{k+1}, \ldots, x_n\}$, such that each symbol in I is independent of other symbols in I, and each symbol in P is (non-trivially) a function of symbols in I only. As seen in Figure 3(b), m code symbols contain parity contribution. Each of these m symbols also has a pure-information component, so it can neither belong to the P set, nor to the I set of a systematic-code coordinate set. This non-systematic structure implies that "many" code symbols contain some parity contribution: a bad property in practice as it dictates accessing many Flash cells for each information update. In this section we propose *systematic* asymmetric limited-magnitude error-correcting codes that have few parity symbols.

A. Systematic Codes for $\ell = 1$ Limited-Magnitude Errors

When the error magnitude ℓ is bounded by 1, the code Σ in Construction 1 is a binary code. As we show next for this case, a modification of any code $\mathcal{C}$ can be carried out, that yields a systematic code with the same correction capability. The construction method of systematic codes for $\ell = 1$ is first presented in the following example.

Example 3. *In this example we propose a systematic variant to the code $\mathcal{C}_H$, given in Example 1. The encoding function given below generates a code that has the same correction capabilities as $\mathcal{C}_H$, namely any single $\ell = 1$ asymmetric error is correctable, though the resulting code is different. Specifically, the dimensions of the systematic code are different. For this example we assume that the alphabet size of the code is 2^m (m – the number of parity bits in the binary code), compared to 2^b for arbitrary b in $\mathcal{C}_H$. This assumption can be lifted with a small increase in redundancy that depends on the actual code parameters. For an $[n, k = n - m]$ binary Hamming code Σ_H, the length of the systematic code is $n - m + 1$, compared to n in the non-systematic case. The systematic code is encoded as follows. In Figure 5 (a), km information bits are input to the encoder. The encoder then uses a binary Hamming encoder to encode the k information bits of the top row into a length $n = k + m$ Hamming codeword (Figure 5 (b)). The parity bits of the Hamming codeword are now placed as a separate column. The mapping of bits to Q symbols, shown in Figure 5 (c), is the usual (positional) mapping for the k information symbols, and the Gray mapping for the parity symbol.*

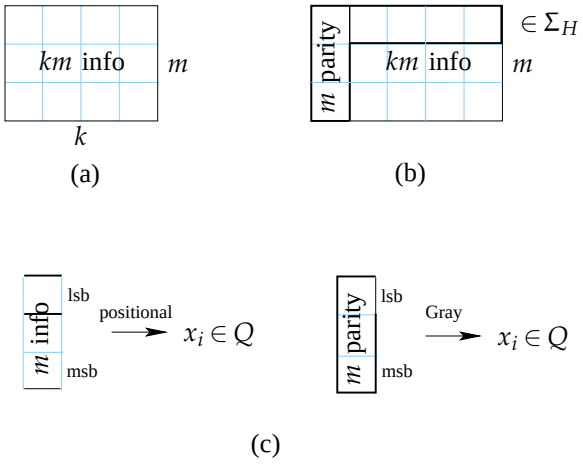


Figure 5. Encoding Procedure for a Systematic Code

To decode, a word from Q^{k+1} is converted back to bits using the same mappings, and a binary Hamming decoder is invoked for the n coded bits. By construction, a single $\ell = 1$ asymmetric error over Q translates to a single bit error in the Hamming codeword: in the k information symbols, an $\ell = 1$ error flips the least-significant bit that is part of the Hamming codeword, and in the parity symbol, an $\ell = 1$ error flips exactly one parity bit in the column, thanks to the Gray code used in the mapping.

The code proposed in Example 3, together with its encoding/decoding, can be generalized to any $\ell = 1$ limited-magnitude t asymmetric error-correcting code as stated by the following proposition.

Proposition 15. *Let Σ be a binary systematic code of length n and $m \leqslant b \cdot r$ parity bits, for any two integers r and $b > 1$. If Σ corrects t symmetric errors, then it can be used to construct a* systematic *t asymmetric $\ell = 1$ limited-magnitude error-correcting code over an alphabet of size $q = 2^b$. This code has length $n - m + r$, of which r symbols are parity symbols.*

Proof: The general construction follows closely the one in Example 3. The $n - m$ information bits are used to encode a codeword of Σ. The $m \leqslant br$ parity bits are grouped into r columns of b bits each. Then these r columns are mapped to Q symbols using the Gray mapping and information bits are mapped to symbols using the positional mapping. The property that each asymmetric limited-magnitude error results in one symmetric error in the codeword of Σ is preserved for this general case. ■

B. Systematic Codes for $\ell > 1$ Limited-Magnitude Errors

If we try to extend the construction of the previous subsection to codes for $\ell > 1$ limited-magnitude errors, we immediately face a stumbling block. Although generalized Gray codes exist for non-binary alphabets, their properties do not suffice to guarantee a similar general construction. The crucial property, that a single asymmetric limited-magnitude error translates to a single symmetric error in the $(\ell+1)$-ary

code, is lost for the general case. For example, if for $\ell = 2$ a symbol represents the ternary reflected Gray codeword 0001, then an error of magnitude 2 will result in the Gray codeword 0012, whose Hamming distance to 0001 is 2 and not 1 as required. Thus, a limited-magnitude error at this symbol may induce 2 errors for the ternary code Σ. Evidently, this effect is not unique to the $(\ell+1)$-ary reflected Gray code, and there is no mapping between q-ary symbols $\left\{0, 1, \ldots, (\ell+1)^b - 1\right\}$ and $(\ell+1)$-ary b-tuples with this property. This subsection proposes a construction for systematic asymmetric ℓ-limited-magnitude error-correcting codes, for arbitrary ℓ.

The construction builds on the non-systematic Construction 1. Two modifications of Construction 1 need to be instituted to yield a systematic code. The first is using a code Σ' that has different correction properties than Σ used before. The second is a special mapping between parity symbols of Σ' and code symbols of $\mathcal{C}'$ over Q.

Let q and $q' = \ell + 1$ be the alphabet sizes of the codes $\mathcal{C}'$ and Σ', respectively. Assume for simplicity that $q = 2(\ell+1)^s$, for some integer s. If this is not the case, the same construction can still be used, only the mappings between Q' and Q will be slightly more complicated.

1) ***The Code*** Σ'*:* Let Σ be a linear systematic code over an alphabet of size $q' = \ell + 1$. The number of information symbols of Σ is denoted κ, and the number of parity symbols is m. The parity-check matrix of Σ is denoted by H. Columns $\{0, \ldots, m-1\}$ of H correspond to the m parity symbols of the code Σ. Let H' be the parity-check matrix that is obtained from H by replicating all columns in $i \in \{0, \ldots, m-1\}$ such that $i \not\equiv 0 \pmod{s}$, and appending them to H. H' is the parity-check matrix of the linear code Σ' that has m parity symbols and $\kappa + \lfloor m(s-1)/s \rfloor$ information symbols.

2) ***The Mapping*** $Q' \leftrightarrow Q$ ***for Parity Symbols****:* From the m parity symbols of Σ', each set of s parity symbols, denoted $\phi_0^{(j)}, \ldots, \phi_{s-1}^{(j)}$, is mapped to a single parity symbol of $\mathcal{C}'$ using the following formula

$$x_j = \phi_0^{(j)} + 2\sum_{i=1}^{s-1} \phi_i^{(j)} (\ell+1)^i. \tag{7}$$

The systematic code $\mathcal{C}'$ is now specified using its encoding function.

Construction 2. *Let Σ be a $[\kappa + m, \kappa]$ linear code over the alphabet Q' of size $q' = \ell + 1$. The systematic code $\mathcal{C}'$ over the alphabet Q of size $q = 2(\ell+1)^s$ has $\kappa + \lfloor m(s-1)/s \rfloor$ information symbols and $\lceil m/s \rceil$ parity symbols. The parity symbols are computed by taking the modulo $\ell + 1$ of the information symbols, encoding them using a systematic encoder for Σ', and mapping the resulting m parity symbols over Q' to $\lceil m/s \rceil$ symbols over Q, as described in* (7)*.*

Note that the length of the code $\mathcal{C}'$ is $\kappa + \lfloor m(s-1)/s \rfloor + \lceil m/s \rceil = \kappa + m$, the length of the non-systematic code $\mathcal{C}$. Codes obtained by Construction 2 have the following error-correction capability.

Theorem 16. *$\mathcal{C}'$ corrects t asymmetric ℓ-limited-magnitude errors if Σ corrects t symmetric errors.*

Proof: The key point in the proof is that an asymmetric ℓ-limited-magnitude error in a parity symbol j of $\mathcal{C}'$ may only change $\phi_0^{(j)}$ out of the s parity symbols $\phi_0^{(j)}, \ldots, \phi_{s-1}^{(j)}$ of Σ', mapped to this symbol. The way Σ' was extended from Σ allows correcting errors in the added information symbols, as long as the parity symbols whose H columns were replicated are guaranteed to be error free. This fact can be verified by using a decoder for Σ' that first computes the syndrome using H' and then inputs this syndrome to a decoder for Σ. Thus t or less asymmetric ℓ-limited-magnitude errors in any combination of information and parity symbols will result in a correctable error for the code Σ'. ■

To clarify Construction 2 an example is now provided.

Example 4. *Suppose we want to protect* 20 *information bits with a systematic code that corrects $t = 1$ asymmetric $\ell = 3$ limited-magnitude error, over an alphabet of size $q = 32$. Since $t = 1$ and $\ell = 3$, we take Σ to be the quaternary Hamming code. More specifically, we choose Σ to be the $[5, 3]$ Hamming code over the alphabet of size $q' = 4$ whose parity-check matrix is given below.*

$$H = \left[\begin{array}{cc|ccc} 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 2 & 3 \end{array}\right]$$

The $m = 2$ left columns of H correspond to the parity symbols of Σ. Note that $q = 2(q')^s$ and $s = 2$.

Replicating the right parity column we obtain H', the parity-check matrix of Σ'.

$$H = \left[\begin{array}{cc|cccc} 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 2 & 3 \end{array}\right]$$

The encoding of 20 *bits of information into a codeword of a systematic code $\mathcal{C}'$ with the specified parameters is described in Figure* 6*. Shaded cells represent parity symbols and unshaded cells represent information symbols. In Figure* 6*(a), the top two bit rows are used to encode a word of Σ' over the Finite Field of size* 4*. In the right part of Figure* 6*(b), information bits are mapped to symbols of Q using the usual binary to integer conversion. In the left part, the parity symbols of Σ' are mapped to a symbol of Q using the mapping defined in equation* (7)*. Figure* 6*(c) shows the final codeword of $\mathcal{C}'$.*

As implied by the constant 2 in equation (7), only half of the alphabet Q is used in the parity symbols. That is equivalent to 1 extra redundant *bit* for each parity symbol of $\mathcal{C}'$. Note that the half factor is true for arbitrary ℓ. Whenever $\ell > 1$, that amount of additional redundancy compares favorably to restricting the parity symbols to be 0 modulo $\ell + 1$, (akin to the Ahlswede et al. "all error-correcting" scheme [1]), which allows using only a $1/(\ell+1)$ fraction of the alphabet Q in parity symbols. It is interesting to note however, that restricting the parity symbols to be 0 modulo $\ell + 1$ turns out to be optimal for the case $t = n$, as proved in [6].

To better understand Construction 2, it may be beneficial to view it as a concatenated coding scheme. The code $\mathcal{C}'$ is a concatenation of the outer code Σ' and an inner code for each symbol (the mapping $Q' \leftrightarrow Q$) that partially corrects an asymmetric ℓ-limited-magnitude error, to have the outer code Σ' observe at most one symmetric error. Figure 7 illustrates this view of the systematic code construction.

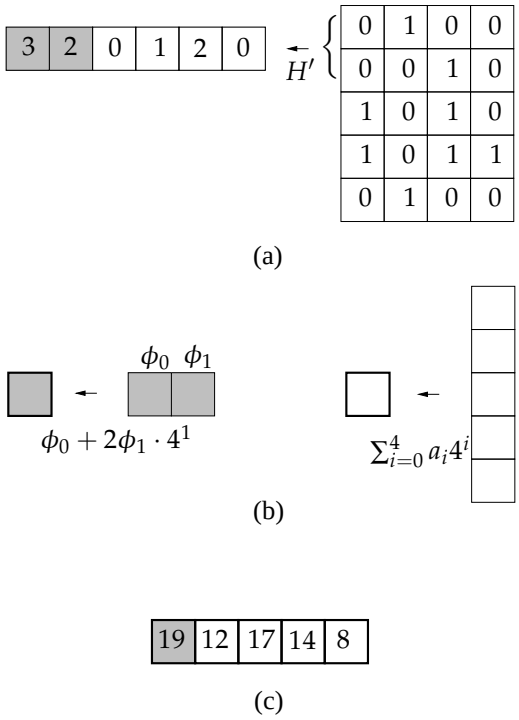


Figure 6. Encoding of a systematic code with $t = 1$ and $\ell = 3$

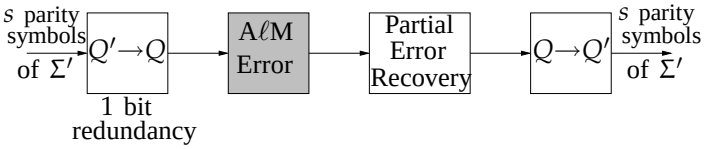


Figure 7. Concatenated Code view of Construction 2

VI. Codes for Asymmetric and Symmetric Limited-Magnitude Errors

In Flash memory applications, the dominant error sources may cause most of the errors to be in one known direction. However, other, more secondary error sources can inject errors that are more symmetrical in nature, but still have low magnitudes. To answer such plausible scenarios, we address a variation of the asymmetric ℓ-limited-magnitude error model to include a (small) number of symmetric ℓ-limited-magnitude errors.

Definition 17. *A $(t_\uparrow, t_\updownarrow)$ asymmetric/symmetric ℓ-limited-magnitude error is a vector $\boldsymbol{e}$ such that $|\{i : e_i \neq 0\}| \leqslant t_\uparrow + t_\updownarrow$. In addition, $t_\updownarrow$ of the indices of $\boldsymbol{e}$ satisfy $-\ell \leqslant e_i \leqslant \ell$, and the remaining $n - t_\updownarrow$ indices satisfy $0 \leqslant e_i \leqslant \ell$.*

In the following, we present a construction method for codes $\mathcal{C}_{\uparrow,\updownarrow}$ that correct $(t_\uparrow, t_\updownarrow)$ asymmetric/symmetric ℓ-limited-magnitude errors. This enhanced error correctability is achieved by modifying Construction 1 with the addition of an auxiliary binary code and a special mapping from information bits to q-ary symbols. We assume for simplicity that $q = 2^s(\ell+1)$, for some integer s.

Construction 3. *Let $\boldsymbol{\sigma} = (\sigma_1, \ldots, \sigma_n)$ be a codeword of a code Σ, over an alphabet of size $\ell + 1$, that corrects $t = t_\uparrow + t_\updownarrow$ symmetric errors. Let $V = (\vec{v}_1, \ldots, \vec{v}_n)$ be a two-dimensional binary array of size $s \times n$, taken from an array code $\mathbb{C}$ that corrects a single bit error in each of at most $t_\updownarrow$ columns[3]. Each symbol of $\boldsymbol{x} \in \mathcal{C}_{\uparrow,\updownarrow}$ is composed from a symbol of the codeword $\boldsymbol{\sigma}$ and a bit vector of the codeword V as follows. For any i,*

$$x_i = (\ell+1) \cdot \mathrm{Gray}(\vec{v}_i) + \sigma_i$$

where $\mathrm{Gray}(\vec{u})$ is the sequential number of the vector $\vec{u}$ in a binary Gray code on s bits. The code $\mathcal{C}_{\uparrow,\updownarrow}$ contains all $|\Sigma| \cdot |\mathbb{C}|$ compositions of the codewords of Σ and $\mathbb{C}$.

Proposition 18. *The code $\mathcal{C}_{\uparrow,\updownarrow}$ is a $(t_\uparrow, t_\updownarrow)$ asymmetric/symmetric ℓ-limited-magnitude error-correcting code.*

Proof: Decoding of $\mathcal{C}_{\uparrow,\updownarrow}$ is performed in two steps. Firstly, $\mathcal{C}_{\uparrow,\updownarrow}$ is decoded as if it were a plain t asymmetric ℓ-limited-magnitude error-correcting code (of Construction 1). For the $t_\updownarrow$ coordinates that possibly suffered errors in the downward direction, the first decoding step miscorrects these errors to *exactly* $\ell + 1$ levels below their correct levels. Thus, for each of these $t_\updownarrow$ miscorrections, the Gray mapping of the upper bits of the symbol guarantees that the resulting error observed by the code $\mathbb{C}$ is a single bit error. ■

Example 5 below illustrates the encoding and decoding of a code originating from Construction 3.

Example 5. *In this example we protect 7 symbols over an alphabet of size $q = 12$ against $t_\uparrow = 2$ asymmetric errors plus $t_\updownarrow = 1$ symmetric error. Both the asymmetric and symmetric errors have magnitude limit of $\ell = 2$. In Figure 8, $\boldsymbol{\sigma}$ is a codeword of the ternary repetition code that corrects $t_\uparrow + t_\updownarrow = 3$ symmetric errors. The bits of V, placed in two rows, are a codeword of the (shortened) binary Hamming code of length 14. Each column of V is mapped to an integer in $\{0,1,2,3\}$ using the Gray code, and the final codeword $\boldsymbol{x}$ combines V and $\boldsymbol{\sigma}$ through the formula*

$$\boldsymbol{x} = 3 \cdot \mathrm{Gray}(V) + \boldsymbol{\sigma}$$

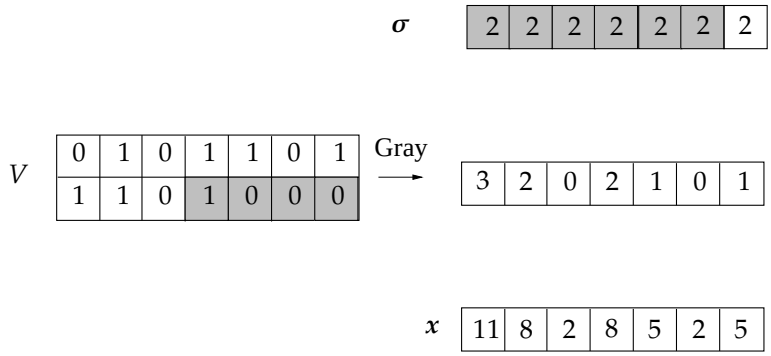


Figure 8. Example of a code for asymmetric and symmetric limited-magnitude errors. From top to bottom: a codeword $\boldsymbol{\sigma}$ of the ternary repetition code; a binary Hamming codeword arranged into a 2×7 array and its Gray mapping; the final codeword $\boldsymbol{x}$ obtained by combining $\boldsymbol{\sigma}$ and V.

[3]Such codes can be obtained by length sn, binary $t_\updownarrow$ error-correcting codes, or more cleverly, using J.K. Wolf's Tensor-Product code construction method [16].

Decoding of the sample code above is illustrated in Figure 9. *The codeword in (a) is corrupted by* 2 *asymmetric (upward) errors and* 1 *downward error; the resulting word is given in (b). In (c) the result of correcting* 3 *asymmetric limited-magnitude errors is given. The "corrected" array* $\tilde{V}$ *is shown in (d), and the top bit of the third column from right (marked with a boldface* **0***) is found to be in error. Finally, in (e) the third symbol from right (in bold face) is adjusted* 3 *levels upward after a miscorrection was detected at the previous step.*

(a) codeword

11	8	2	8	5	2	5

(b) corrupted

11	10	2	8	4	3	5

(c) AℓM decoded

11	8	2	8	2	2	5

(d) corrected $\tilde{V}$

0	1	0	1	**0**	0	1
1	1	0	1	0	0	0

(e) SℓM adjusted

				+3		
11	8	2	8	**5**	2	5

Figure 9. Example of decoding asymmetric and symmetric limited-magnitude errors. (a) Codeword. (b) Codeword corrupted by asymmetric and symmetric limited-magnitude errors. (c) First decoding step: correction of asymmetric limited-magnitude (AℓM) errors. (d) Resulting corrected codeword $\tilde{V}$ is decoded using a Hamming decoder. (e) Adjusting the miscorrection of the symmetric error found in the previous step.

Note that the amount of redundancy (of both $\boldsymbol{\sigma}$ *and* V*) required in the example to correct* $(2,1)$ *asymmetric/symmetric errors is smaller than if* V *is not restricted and the repetition code is taken over an alphabet of size* $2\ell+1=5$ *(that scheme would correct* 3 *symmetric* $\ell=2$ *limited magnitude errors).*

The counter-intuitive part of Construction 3 is that *binary* Gray mappings are used regardless of the error-magnitude ℓ. This fact implies that the codes Σ and $\mathbb{C}$ cooperate with each other to achieve the prescribed correction capability, otherwise $\mathbb{C}$ would need to operate over a larger alphabet for $\ell > 1$.

VII. Speeding up Flash Access with Asymmetric Limited-Magnitude Error-Correcting Codes

Error-correcting codes are usually used for Forward Error Correction (FEC), namely to protect the data integrity against uncontrolled errors. In this section we show that asymmetric limited-magnitude error-correcting codes, in addition to the standard FEC capabilities shown earlier in the paper, can be used to speed up the writing[4] process to Flash devices. This is done by relaxing the programming accuracy requirements, and using the codes to correct the resulting programming errors. Since the Flash programming mechanism is inherently probabilistic, the introduction of "intentional" programming errors in a controlled way can significantly reduce the average programming time and improve the write performance. Such an outcome would be highly desirable given the inferiority of Flash devices in write performance compared to their read performance, and to the sequential write performance of hard-disk drives. The programming speedup is next analyzed quantitatively by calculating the savings in programming time as a function of ℓ (the correctable error magnitude of the employed codes).

The behavior of a typical optimized Flash programming sequence is shown in the graphs of Figure 10, which is taken from [3]. The integers of the horizontal axis represent the program-pulse sequential numbers and the vertical axis represents electric-current levels to which Flash cells are programmed. A circle on the a graph represents a current level achieved by a pulse at some point along the programming sequence. The different graphs in Figure 10 represent program sequences with different target current values. As can be clearly seen, most of the progress toward the target value is achieved by the early pulses, and the numerous later pulses are used for a fine asymptotic convergence to a value very close to the target. Therefore, having even a small error resiliency against asymmetric limited-magnitude errors can allow the programming sequence to terminate long before hitting the target value (due to the asymptotic nature of the programming curves) thus significantly speeding up memory access. Increasing the error resiliency beyond the flat part of the curve does not add significant benefits, as at the steeper part of the curve the vertical concentration of programming points becomes sparser.

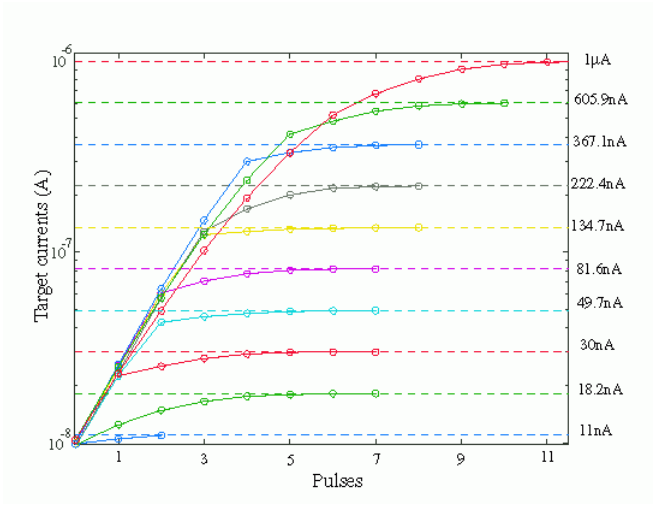


Figure 10. Performance of a Flash adaptive program sequence [3]. The circles on each curve describe the results of an iterative programming algorithm for a given target value.

To supplement the experimental evidence above, that tolerance to asymmetric limited-magnitude errors can speed-up the programming sequence, a quantitative analysis of the time

[4]Memory write is referred to as *programming* in the Flash literature

savings is now carried out. The inputs to a Flash programming algorithm are the *initial* and *target* current levels; its output is a programming pulse of some width and amplitude, that attempts to move closer to the target level, under some constraints. To have an analytic description of the programming sequence, we need to model the programming algorithm in a way that captures its main design constraints in practice. In Flash devices, preventing *over-programming*, whereby the programming result exceeds the target level, is a crucial consideration taken by the programming algorithm. The reason for that being that Flash devices do not support single-cell erases, and an over-programming instance requires erasing a full Flash *block*, an operation that is very costly in terms of time and device wear. The analysis that follows, strongly builds on that property of Flash devices.

Suppose a Flash cell is to be programmed from a lower level I_i to a higher target level I_F. Since the change δ in the current level is a random variable whose distribution depends on the chosen programming pulse, we model it as an *exponentially distributed* random variable with mean $1/\mu$. μ will be determined by the programming algorithm as a function of I_i, I_F, and subject to a constraint of fixing a low probability of over-programming. Specifically, μ will be taken such that

$$\Pr(I_i + \delta > I_F) = \epsilon$$

ϵ is a global parameter that specifies the allowable probability of over-programming. Substituting the exponential distribution of δ, we get the integral equation

$$\int_{I_F - I_i}^{\infty} \mu \exp(-\mu\delta) d\delta = \epsilon \tag{8}$$

(See Figure 11 for illustration.)

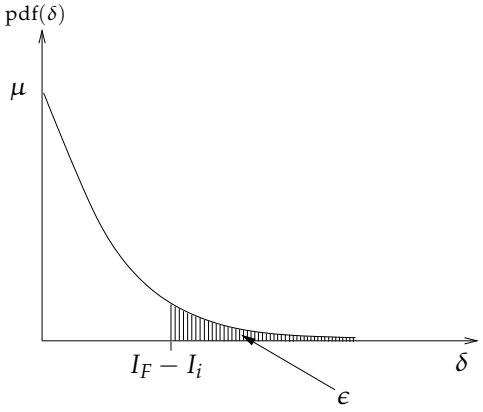


Figure 11. Choice of a programming distribution based on the specified probability of over-programming. For starting level I_i and target level I_F the parameter μ of the exponential distribution is chosen such that the marked area under the probability density function graph equals ϵ (the specified probability of over-programming)

Solving (8) and rearranging we get

$$\mu = -\frac{\ln(\epsilon)}{I_F - I_i}$$

Hence we have the following relationship between the lower level I_i and the final (higher) level I_{i+1}:

$$I_{i+1} = I_i + \delta_i \ , \qquad \delta_i \sim \text{Exponential}[-\ln(\epsilon)/(I_F - I_i)] \tag{9}$$

Note that the parameter of the exponential distribution of δ_i at each step i depends on the starting level I_i that is itself a random variable.

Starting from an initial level I_0, the programming algorithm recursively updates the cell level according to (9), and stops after the n^{th} step if $I_n \geqslant I_F - \Delta$, where Δ is the maximum allowed deviation from the target level I_F. Discussed in detail later, the parameter Δ specifies the device tolerance to programming errors in the downward direction. A pictorial illustration of the modeled programming sequence is given in Figure 12.

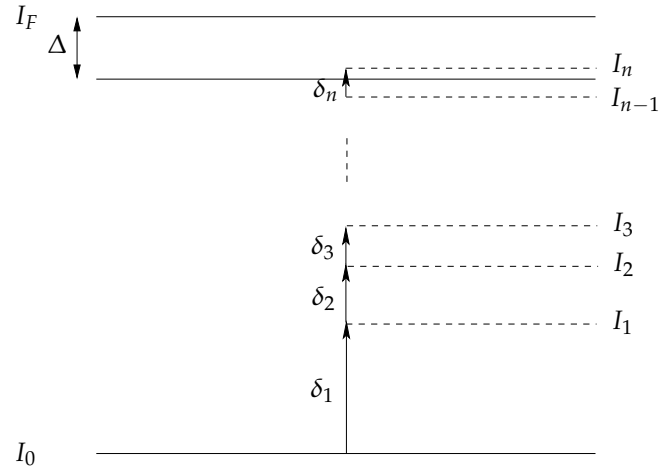


Figure 12. A pictorial illustration of the modeled programming sequence. On the left side are the initial level I_0, the target level I_F and the tolerance parameter Δ. In the middle is a sequence of exponentially distributed level increments $\delta_1, \delta_2, \ldots, \delta_n$ resulting from the programming algorithm. On the right side are the instantaneous levels I_i until the process terminates at I_n.

To analyze the performance of the programming algorithm, we need to find the expected number of steps n, such that

$$I_{n-1} < I_F - \Delta \leqslant I_n$$

However, given the complex[5] structure of the random process I_i, finding the mean of n is hard. Instead, we will approximate I_i's mean crossing time by the (deterministic) crossing time of the mean of I_i. This latter calculation is significantly easier since we can use the linearity of expectation to obtain a recursive formula for the mean of I_i. The accuracy of that approximation can be established using concentration bounds (e.g. Chebyshev inequality), however for the discussion here a first order approximation should suffice.

Now taking the mean of equation (9) we write

$$\overline{I_{i+1}} = \overline{I_i} + E\left[\frac{1}{\mu_i}\right] = \overline{I_i} + K_\epsilon(I_F - \overline{I_i}) \tag{10}$$

[5] I_i is a Markov process with an uncountable number of states

where $K_\epsilon \triangleq -1/\ln(\epsilon)$. Rewriting (10) provides a recurrence relation on the expected programmed levels

$$\overline{I_{i+1}} = \overline{I_i}(1-K_\epsilon) + K_\epsilon I_F$$

Solving the recurrence for initial level I_0 we get the expression

$$\overline{I_n} = I_0(1-K_\epsilon)^n + I_F K_\epsilon \sum_{i=1}^{n}(1-K_\epsilon)^{i-1}$$

which after simplification becomes

$$\overline{I_n} = I_F - (1-K_\epsilon)^n(I_F - I_0) \tag{11}$$

Now, by equating (11) to $I_F - \Delta$ we can calculate the time N when the sequence of means $\overline{I_n}$ crosses $I_F - \Delta$:

$$I_F - (1-K_\epsilon)^N(I_F - I_0) = I_F - \Delta$$

that gives

$$N = \frac{\log(I_F - I_0) - \log(\Delta)}{-\log(1-K_\epsilon)} \tag{12}$$

The importance of (12) is that it describes how the number of required pulses N depends on the error margin Δ. To compare the programming speed of Flash devices with and without an asymmetric limited-magnitude error-correcting code, we define two different error margins, Δ_c and Δ_{uc}, respectively (the subscript *c* stands for *coded* and the subscript *uc* stands for *uncoded*, and obviously $\Delta_c > \Delta_{uc}$). The difference between the corresponding numbers of pulses N_{uc} and N_c is then

$$N_{uc} - N_c = \frac{\log(\Delta_c/\Delta_{uc})}{-\log(1-K_\epsilon)}$$

A conservative assumption is to set $\Delta_c = (\ell+1)\Delta_{uc}$, where ℓ is the parameter of the asymmetric ℓ-limited-magnitude error-correcting code. This assumption corresponds to allowing the uncoded device a tolerance of one level (over the discrete alphabet Q), and the coded device a tolerance of ℓ additional levels for the total of $\ell+1$ levels. Under that assumption, the savings in the number of programming pulses equals

$$N_{uc} - N_c = \frac{\log(\ell+1)}{-\log(1-K_\epsilon)} \tag{13}$$

For an over-programming probability $\epsilon = 0.01$ the above equals

$$N_{uc} - N_c = 4.08\log(\ell+1)$$

Values of savings for different values of ℓ are given in Table I.

ℓ	$N_{uc} - N_c$
1	2.84
2	4.48
3	5.66
4	6.57
5	7.31
6	7.94

TABLE I
APPROXIMATE AVERAGE SAVINGS IN PROGRAMMING PULSES FOR SAMPLE VALUES OF ℓ

Another quantity of interest is the percentage of savings $(N_{uc} - N_c)/N_{uc} \times 100$, which depends on the particular difference $I_F - I_0$. For a programming window of $I_F - I_0 = a\Delta$, a is an integer specifying the target increase in discrete levels, the part of the programming duration saved by the code equals

$$\frac{\log(\ell+1)}{\log a},$$

as long as $a < q - \ell$. The median[6] percentage savings is obtained by taking $a = q/2$ and is equal to

$$\frac{\log(\ell+1)}{\log(q/2)}.$$

For a sample number of levels $q = 32$, the median savings in programming time suggested by the model is plotted in Figure 13.

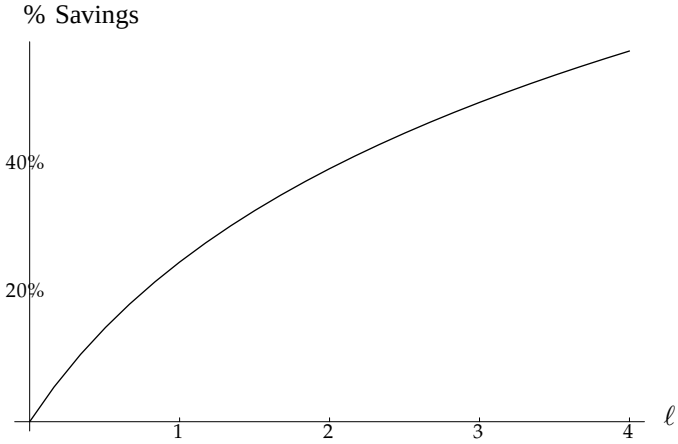


Figure 13. Percentage of program-time savings as a function of the code's magnitude limit parameter ℓ. Significant savings are suggested even for small ℓ and returns are diminishing for growing ℓ.

As seen in both Figure 13 and Table I, while even small ℓ values suggest significant savings, increasing ℓ beyond some point exhibits diminishing returns and does not significantly contribute to increased savings in programming time. Note that this last qualitative observation is one we have already made when discussing Figure 10 earlier in the sub-section. Thus both analytical and experimental evidence motivate the application of asymmetric limited-magnitude error-correcting codes (with small ℓ), as clearly codes for symmetric errors will not be an efficient solution for programming speed-up.

VIII. CONCLUSIONS AND FUTURE RESEARCH

This paper proposes a new coding technique that is motivated by multi-level Flash memories. Defining a natural new error model has opened the way to a simple but powerful construction method that enjoys good storage and implementation efficiencies. By an interplay between symbol mappings and constraints on the full code block, several useful extensions to the basic code construction are achieved. An attractive property of the codes herein, is that the coding parameters n, t, ℓ need not be fixed for a Flash memory device family. After implementing the simple circuitry to support this coding

[6]The median savings is a simple approximation to the average savings, which has an unwieldy expression. For small ℓ (compared to q) it is a relatively good approximation.

method in general (modulo and other arithmetic operations), different code parameters can be chosen, depending on the application, by using varying external coding modules for the symmetric error-correcting code. Many of the strengths of this construction method were not explored in the current paper. When the reading resolution is larger than the code alphabet size (e.g., readers that give a real number rather than an integer), improved decoding techniques can be readily applied using "limited-magnitude erasures" or other soft interpretations of the read symbols. Better systematic codes may be obtained by observing the relation between the limited-magnitude errors and the errors they impose on the low-alphabet code, and then replacing the symmetric error-correction properties we required (which are too strong) with various Unequal Error Protection properties. An interesting open problem is showing (if true) the asymptotic optimality of Construction 1 for all values of ℓ and t. This fact lies upon the existence of a proof to the following conjecture.

Conjecture 1. *For any a and t, $A_a(n,t)$ (size of largest a-ary code for symmetric errors) and $\mathsf{Asym}_a(n,t)$ (size of largest a-ary code for asymmetric errors) satisfy the following equality.*

$$\lim_{n\to\infty}\frac{1}{n}\log_a|A_a(n,t)| = \lim_{n\to\infty}\frac{1}{n}\log_a|\mathsf{Asym}_a(n,t)|$$

(This was proved here for $a=2$ and for restricted t if $a>2$).

ACKNOWLEDGMENTS

The authors wish to thank Anxiao Jiang for insightful comments and suggestions. In addition, the authors acknowledge the anonymous reviewers' contributions to the paper's quality of presentation.

REFERENCES

[1] R. Ahlswede, H. Aydinian, and L. Khachatrian, "Unidirectional error control codes and related combinatorial problems," in *Proc. of the Eighth International Workshop on Algebraic and Combinatorial Coding Theory (ACCT-8), St. Petersburg, Russia (Extended version available at: http://arxiv.org/abs/cs/0607132)*, 2002, pp. 6–9.

[2] S. Al-Bassam and B. Bose, "Asymmetric/unidirectional error correcting and detecting codes," *IEEE Transactions on Computers*, vol. 43, no. 5, pp. 590–597, 1994.

[3] A. Bandyopadhyay, G. Serrano, and P. Hasler, "Programming analog computational memory elements to 0.2% accuracy over 3.5 decades using a predictive method," in *Proc. of the IEEE International Symposium on Circuits and Systems*, 2005, pp. 2148–2151.

[4] J. Borden, "Bounds and constructions for error correcting/detecting codes on the Z-channel," in *Proc. IEEE International Symposium on Information Theory*, 1981, pp. 94–95.

[5] B. Eitan and A. Roy, "Binary and multilevel Flash cells," *Flash Memories, P. Cappelletti, C. Golla, P. Olivo, E. Zanoni Eds. Kluwer*, pp. 91–152, 1999.

[6] N. Elarief and B. Bose, "Optimal, systematic q-ary codes correcting all asymmetric errors of limited magnitude," in *Proc. of the IEEE International Symposium on Info. Theory*. Seoul, Korea: IEEE, Jun. 2009, pp. 2704–2707.

[7] S. Gregori, A. Cabrini, O. Khouri, and G. Torelli, "On-chip error correcting techniques for new-generation Flash memories," *Proceedings of the IEEE*, vol. 91, no. 4, pp. 602–616, 2003.

[8] W. Huffman and V. Pless, *Fundamentals of Error-Correcting Codes*. Cambridge, UK: Cambridge university press, 2003.

[9] H. Kaneko and E. Fujiwara, "A class of M-ary asymmetric symbol error correcting codes for data entry devices," *IEEE Transactions on Computers*, vol. 53, no. 2, pp. 159–167, 2004.

[10] T. Kløve, "Error correcting codes for the asymmetric channel," Dept. Mathematics, University of Bergen, Norway, Tech. Rep. 18-09-07-81, 1981.

[11] J. Leech and N. Sloane, "Sphere packing and error-correcting codes," *Canadian J. Math.*, vol. 23, no. 4, pp. 718–745, 1971.

[12] F. MacWilliams and N. Sloane, *The Theory of Error-Correcting Codes*. Amsterdam, The Netherlands: North Holland, 1977.

[13] R. McEliece, "Comments on 'A class of codes for asymmetric channels and a problem from the additive theory of numbers'," *IEEE Trans. on Inform. Theory*, vol. 19, no. 1, p. 137, Jan. 1973.

[14] S. Stein, "Packings of $\mathbb{R}^n$ by certain error spheres," *IEEE Trans. on Inform. Theory*, vol. 30, no. 2, pp. 356–363, Mar. 1984.

[15] R. Varshamov, "A class of codes for asymmetric channels and a problem from the additive theory of numbers," *IEEE Trans. on Inform. Theory*, vol. 19, no. 1, pp. 92–95, Jan. 1973.

[16] J. K. Wolf, "An introduction to tensor product codes and applications to digital storage systems," in *Proc. IEEE Information Theory Workshop*, Chengdu, China, 2006, pp. 6–10.

Yuval Cassuto Yuval Cassuto (S'02-M'08) is a Research Staff Member at Hitachi Global Storage Technologies, San Jose Research Laboratory. His research focuses on information theory, error-correcting codes, storage architecture, performance and security.

He received the B.Sc degree in Electrical Engineering, summa cum laude, from the Technion, Israel Institute of Technology, in 2001, and the MS and Ph.D degrees in Electrical Engineering from the California Institute of Technology, in 2004 and 2008, respectively.

From 2000 to 2002, he was with Qualcomm, Israel R&D Center, where he worked on modeling and analysis of physical layer communication principles.

Dr. Cassuto was awarded the 2001 Texas Instruments DSP and Analog Challenge $100,000 award, as well as the Powell and Atwood graduate research fellowship awards.

Moshe Schwartz Moshe Schwartz was born in Israel in 1975. He received the B.A., M.Sc., and Ph.D. degrees from the Technion – Israel Institute of Technology, Haifa, Israel, in 1997, 1998, and 2004 respectively, all from the Computer Science Department.

He was a Fulbright post-doctoral researcher in the Department of Electrical and Computer Engineering, University of California San Diego, USA, and a post-doctoral researcher in the Department of Electrical Engineering, California Institute of Technology. He now holds a position with the Department of Electrical and Computer Engineering, Ben-Gurion University, Israel. His research interests include algebraic coding, combinatorial structures, and digital sequences.

Vasken Bohossian Vasken Bohossian received his B.S.E. in electrical engineering (honors program) from McGill University in 1993, the M.S. degree in electrical engineering and the Ph.D. degree in computation and neural systems from the California Institute of Technology in 1994 and 1998 respectively. He is a co-founder of Rainfinity, acquired by EMC Corporation in 2005. His research interests include parallel and distributed computing, fault-tolerant computing, error-correcting codes, computation theory and threshold logic.

Jehoshua Bruck Jehoshua (Shuki) Bruck is the Gordon and Betty Moore Professor of Computation and Neural Systems and Electrical Engineering at the California Institute of Technology. His research focuses on information theory and systems and the theory biological networks.

He received the B.Sc. and M.Sc. degrees in electrical engineering from the Technion, Israel Institute of Technology, in 1982 and 1985, respectively and the Ph.D. degree in Electrical Engineering from Stanford University in 1989.

He has an extensive industrial experience, including working with IBM Research where he participated in the design and implementation of the first IBM parallel computer. He was a co-founder and chairman of Rainfinity, a spin-off company from Caltech that focused on software products for management of network information storage systems.

He is an IEEE fellow, and his awards include the National Science Foundation Young Investigator award and the Sloan fellowship. He published more than 200 journal and conference papers in his areas of interests and he holds more than 30 US patents. His publications were recognized by awards, including, a selection as an ISI highly cited researcher, winning the 2005 S. A. Schelkunoff Transactions prize paper award from the IEEE Antennas and Propagation society and the Best Paper Award in the 2003 Design Automation Conference.