Update-Efficiency and Local Repairability Limits for Capacity Approaching Codes

Arya Mazumdar *Member, IEEE*, Venkat Chandar, and Gregory W. Wornell *Fellow, IEEE*

***Abstract*—Motivated by distributed storage applications, we investigate the degree to which capacity achieving codes can be efficiently updated when a single information symbol changes, and the degree to which such codes can be efficiently repaired when a single encoded symbol is lost.**

Specifically, we first develop conditions under which optimum error-correction and update-efficiency are possible. We establish that the number of encoded bits that should change in response to a change in a single information bit must scale logarithmically in the block-length of the code, if we are to achieve any nontrivial rate with vanishing probability of error over the binary erasure or binary symmetric channels. Moreover, we show that there exist capacity-achieving codes with this scaling.

With respect to local repairability, we develop tight upper and lower bounds on the number of remaining encoded bits that are needed to recover a single lost encoded bit. In particular, we show that when the rate of an optimal code is ϵ below capacity, the maximum number of codeword symbols required to recover one lost symbol must scale as $\log 1/\epsilon$.

Several variations on—and extensions of—these results are also developed, including to the problem of rate-distortion coding.

***Index Terms*—error-correcting codes, linear codes, update-efficiency, locally repairable codes, low-density generator matrix codes, low-density parity check codes, channel capacity**

I. INTRODUCTION

THERE is a growing need to provide reliable distributed data storage infrastructure in highly dynamic and unreliable environments. Storage nodes (servers) can switch between on- and off-line frequently, due to equipment failures, breaks in connectivity, and maintenance activity. Corruptions of the data can also arise. Furthermore, the data itself is often changing frequently, requiring the constant updating of storage nodes.

These characteristics place additional demands on the error-control coding typically considered for such systems, and there are important questions about the degree to which those demands can be accommodated, which the community has begun to investigate in recent years.

In this paper, we focus on two specific such demands, for a scenario in which each symbol of an encoded message is stored at a different node in the network.[1] The first is that the coding be *update-efficient* or *locally updatable*, i.e., that small changes in the data require only small changes in its coded representation. The second is that the coding be *locally repairable* or *recovery-efficient*, i.e., that small portions of the coded representation that are lost can be recovered from correspondingly small portions of the rest of the encoded symbols. The degree to which these demands can be accommodated affects the bandwidth, energy, and/or other resource consumption characteristics of such infrastructure.

There is considerable flexibility in terms of how to model the notion of "portion" when investigating such questions, and the choice in practice is typically dictated by the application. For example, in the context of update-efficiency one can ask: if a single bit in the message changes, how many bits, in aggregate, of the symbols in the associated codeword must change in response? Analogously, in the context of recovery-efficiency, one can ask: if a single symbol in the encoding of a message is lost, how many bits of information, in aggregate, are required from the remaining symbols to specify how to repair the encoding?

Such formulations have received significant attention in the literature. They are appropriate in scenarios, e.g., where the resource cost in the system is proportional to the communication bandwidth between storage nodes. Examples of work in this vein include [1], which develops a class of locally "regenerative codes," and [2], which shows that it is possible to have "good" codes that are both update-efficient and minimize the repair bandwidth.

As an alternative model, in the context of update-efficiency we can ask how many symbols in the message must change in response to a single-symbol change in the message, in order to update the encoding. In the context of recovery-efficiency, as the counterpart, in response to the loss of a single encoded symbol we can ask how many of the remaining symbols are needed to be able to reconstruct it.[2] This formulation is more natural in a scenario where the dominant resource cost is that of simply establishing a link, regardless of its size, between storage nodes (such as when such nodes take the form of

Manuscript received May 15, 2013; revised October 1, 2013 and December 10, 2013. This work was supported in part by the US Air Force Office of Scientific Research under Grant No. FA9550-11-1-0183, by the National Science Foundation under Grant No. CCF-1017772, and by a grant from University of Minnesota. The results of this paper were presented in part at the 2012 IEEE International Symposium on Information Theory, Cambridge, MA, July 1-6, 2012, and at the Information Theory and Applications Workshop, San Diego, CA, Feb. 9-14, 2013.

A. Mazumdar was with the Research Laboratory of Electronics, Massachusetts Institute of Technology, Cambridge, MA 02139. He is now with the Department of Electrical and Computer Engineering, University of Minnesota, Minneapolis, MN 55455 (e-mail: arya@umn.edu).

V. Chandar is with MIT Lincoln Laboratory, Lexington, MA 02420 (e-mail: vchandar@mit.edu).

G. W. Wornell is with the Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139 (e-mail: gww@mit.edu).

Digital Object Identifier 10.1109/JSAC.2014.1405xx.

[1]Since the symbol alphabet is unconstrained, a symbol could represent a packet or block of bits of arbitrary size.

[2]It is worth emphasizing that in general neither model is necessarily more restrictive than the other, since in each case we have some flexibility in the choice of symbol alphabet.

servers).

This alternative model, which has attracted much attention in the community, is the focus of the present paper. Moreover, to simplify the exposition, our treatment will largely focus on a special case of this model in which the codeword symbols are, themselves, bits, i.e., *binary-valued*, in which it of course also corresponds to a special case of the first model.

A. *Update-Efficiency*

While the broader notion of update-efficiency is implicit in work on array codes—see, e.g., [3], [4]—the first substantial analysis of update-efficiency appears in [5].

When changes in the data are significant, updating a linear (in the blocklength of the code) number of the encoded symbols is unavoidable for any code. However, when the changes are incremental, updating a sublinear number of the symbols can be sufficient. For example, [5] considers codes over a binary alphabet and shows the existence of a linear code that achieves the capacity of the binary erasure channel (BEC) with the property that any single-bit change in the message requires only a logarithmic (in the block-length) number of bits to be updated in the codeword.

In this paper, we begin with a brief discussion of update-efficient codes that can correct arbitrary sets of (adversarial) errors/erasures, since such adversarial models are common in current storage applications. In this case, update-efficiency and error-correctability are directly conflicting objectives. In particular, it is not possible to correct more than (roughly) $t/2$ errors (or roughly t erasures) with a code that needs at most t bits of update for any single-bit change in the message. This is because the minimum pairwise distance between the codewords (i.e., the *minimum distance* of the code) is upper bounded by t. We mention properties of linear codes that are useful for constructing good update-efficient adversarial-error-correcting codes, i.e., codes that achieve the best possible tradeoff. Perhaps the most interesting observation for this scenario is that if there exists a linear code with a given rate and minimum distance, then there exists another linear code with same parameters that is as update-efficient as possible.

The remainder of our development focuses on the random failure model, where much better *average* performance is possible. We begin with a simple derivation of one of the main propositions of [5], i.e., that there exist linear codes that achieve the capacity of the BEC such that for any single-bit change in the message, only $O(\log n)$ bits have to be updated in a codeword of length n. However, our main result is the converse statement: we show that if a linear code of positive rate achieves arbitrarily small probability of error over a BEC, then a single-bit change in the message must incur an updating of $\Omega(\log n)$ bits in the codeword. In addition, we estimate $\gamma > 0$ such that there cannot exist a linear code with positive rate and arbitrarily small probability of error that requires fewer than $\gamma \log n$ bit updates in the codeword per single bit change in the message.

B. *Local Repairability*

The potential for a code to be locally repairable—which is, in some sense, a notion dual to that of update-efficiency—was first investigated in [6].

For significant losses of codeword symbols, the minimum distance properties of the code naturally characterize the worst-case requirements to repair the encoding. However, the ability to repair smaller losses depends on other properties of the code. Accordingly, [6] investigates code structure allowing any single symbol of any codeword to be recovered from at most a constant number of other symbols of the codeword, i.e., from a number of symbols that does not grow with the length of the code.

From this perspective, in practice one might hope to have codes with both a large minimum distance and structure that enables recovery from the single symbol losses using the fewest possible number of remaining symbols.

To this end, [6], [7] consider locally repairable codes that also correct a prescribed number of adversarial errors (or erasures), and develop a trade-off between the local repairability and error-correctability. In particular, it is shown that for a q-ary linear code ($q \geq 2$) of blocklength n, the minimum distance d satisfies

$$d \leq n - k - \left\lceil \frac{k}{r} \right\rceil + 2,$$

where k is the code dimension and r is the number of symbols required to reconstruct a single symbol of the code (referred to as the local repairability).

Such results can be generalized to nonlinear codes of any desired alphabet size. Indeed, [8] shows that for any q-ary code with size M, local repairability r, and minimum distance d, we have

$$\log M \leq \min_{1 \leq t \leq \lceil \frac{n}{r+1} \rceil} \Big[tr + \log A_q(n - t(r+1), d) \Big], \quad (1)$$

where $A_q(n, d)$ is the maximum size of a q-ary code of length n and distance d.

It should be noted that, in contrast to the case of update-efficiency, which requires a number of codewords to be close to each other, there is no immediate reason that a code cannot have both good local repairability and good error-correctability. Perhaps not surprisingly, there has, in turn, been a growing literature exploring locally repairable codes with other additional properties; see, e.g., [9]–[11].

In this paper, we focus on probabilistic channel models that take into account the statistics of node failure, and optimize average performance. To the best of our knowledge the associated capacity results for locally repairable codes have yet to be developed. Indeed, the analysis of local repairability in the existing literature is almost invariably restricted to an adversarial model for node failure. While combinatorially convenient, there is no guarantee the resulting codes are good in an average sense.

In our development, we first show that it is possible to construct codes operating at a rate within ϵ of the capacity of the BEC that have both local repairability $O(\log 1/\epsilon)$ and an update-efficiency scaling logarithmically with the block-length. However, our main result in this part of the paper is a converse result establishing that the scaling $O(\log 1/\epsilon)$ is optimal for a BEC—specifically, we establish that if the rate of a code that achieves arbitrarily small probability of error over a BEC is ϵ below capacity, then the local repairability is $\Omega(\log 1/\epsilon)$.

C. Channels with Errors

Most of our development focuses on the case of hard node failures that result in data loss. However, in some less common scenarios of distributed storage, node failures are undetected, resulting in data corruption. While the BEC is a natural model for the former, it is the binary symmetric channel (BSC) that is the corresponding model for the latter. Much, but not all, of our development carries over to the case of the BSC.

In particular, our results on the existence of capacity-achieving codes that are both update-efficient and locally repairable also hold for the BSC. Likewise, our converse result for update-efficient linear codes also holds the BSC. However, we have an additional converse result for general codes that applies only to the BSC, and our converse result for local repairability applies only to the BEC (indeed, local repairability perhaps makes little sense for BSC).

D. Source Coding

We note that the notions of update-efficiency and local repairability are also applicable to *source coding*, i.e., data compression. In the context of lossless source coding, update-efficient codes have been considered before in papers such as [12]; see also [13] and the references therein. However, the *lossy* source coding version of the update-efficient coding problem does not appear to have been formalized previously.

Some preliminary results regarding the update-efficiency and local repair properties are summarized towards the end of the paper, where we also summarize some open questions in this area.

E. Organization

The organization of the paper is as follows. After Section II establishes the concepts and notation that will be used throughout the paper, Section III describes the worst-case error-correction capability of an update-efficient code. In Section IV we show that there exist linear codes of length n and rate ϵ less than capacity, with update-efficiency logarithmic in blocklength and local repairability $O(\log 1/\epsilon)$. In Section V, we develop our main impossibility results for capacity-achieving update-efficient codes. In Section VI we turn to the local repairability of capacity-achieving codes, and develop our converse. In Section VII, we describe a generalization of update efficiency beyond single-bit updates, and discuss the existence of good codes for such scenarios. In Section VIII we introduce and comment on update-efficiency and local repairability in the context of the dual problem of lossy source coding. Finally, Section IX contains some concluding remarks on extensions of the results to larger alphabets.

II. Concepts and Notation

First, we use BEC(p) to denote a binary erasure channel with loss probability p, which has capacity $1-p$. Analogously, we use BSC(p) to denote a binary symmetric channel with crossover probability p, which has capacity $1 - h_{\mathrm{B}}(p)$ where $h_{\mathrm{B}}(p) = -p\log_2(p) - (1-p)\log_2(1-p)$ is the binary entropy function.[3]

Next, for our purposes a *code* $\mathcal{C} \in \mathbb{F}_2^n$ is a collection of binary n-vectors. The *support* of a vector $\boldsymbol{x}$ (written as $\mathrm{supp}(\boldsymbol{x})$) is the set of coordinates where $\boldsymbol{x}$ has nonzero values. By the *weight* of a vector $\boldsymbol{x}$ we mean the size of its support, which we denote using $\mathrm{wt}(\boldsymbol{x})$.

Let $\mathscr{M}$ be the set of all possible messages. Usually accompanied with the definition of the code, is an injective encoding map $\phi : \mathscr{M} \to \mathcal{C}$, which defines how the messages are mapped to codewords. In the following discussion, let us assume $\mathscr{M} = \mathbb{F}_2^k$. In an update-efficient code, for all $\boldsymbol{x} \in \mathbb{F}_2^k$, and for all $\boldsymbol{e} \in \mathbb{F}_2^k$ such that $\mathrm{wt}(\boldsymbol{e}) \le u$, we have $\phi(\boldsymbol{x}+\boldsymbol{e}) = \phi(\boldsymbol{x}) + \boldsymbol{e}'$, for some $\boldsymbol{e}' \in \mathbb{F}_2^n$ such that $\mathrm{wt}(\boldsymbol{e}') \le t$. A special case of this, captured in the following definition, is of primary interest in this paper.

Definition 1: The *update-efficiency* of a code $\mathcal{C}$ and the encoding ϕ, is the maximum number of bits that needs to be changed in a codeword when a single bit in the message is changed. A code $(\mathcal{C}, \phi)$ has update-efficiency t if for all $\boldsymbol{x} \in \mathbb{F}_2^k$, and for all $\boldsymbol{e} \in \mathbb{F}_2^k : \mathrm{wt}(\boldsymbol{e}) = 1$, we have $\phi(\boldsymbol{x}+\boldsymbol{e}) = \phi(\boldsymbol{x}) + \boldsymbol{e}'$, for some $\boldsymbol{e}' \in \mathbb{F}_2^n : \mathrm{wt}(\boldsymbol{e}') \le t$.

A *linear code* $\mathcal{C} \in \mathbb{F}_2^n$ of dimension k is a k-dimensional subspace of the vector space $\mathbb{F}_2^n$. By an $[n, k, d]$ code we mean a linear code with length n, dimension k and minimum distance between codeword pairs of d. Linear codes are of particular interest given their attractive representation, encoding, and decoding complexity.

For linear codes, the encoding is linear if the mapping $\phi : \mathbb{F}_2^k \to \mathcal{C}$ can be expressed in the form $\phi(\boldsymbol{x}) = \boldsymbol{x}^{\mathrm{T}} G$, for any $\boldsymbol{x} \in \mathbb{F}_2^k$. The $k \times n$ matrix G is called a *generator matrix* of the code. Also associated with such a code is its parity-check matrix H, a generator matrix of the nullspace of $\mathcal{C}$, whence $H\boldsymbol{x} = \mathbf{0}$ for all $\boldsymbol{x} \in \mathcal{C}$.

For a linear encoding, when changing a single bit in the message, the maximum number of bits that need to be changed in the codeword is the maximum over the weights of the rows of the generator matrix, as the following proposition establishes.

Proposition 1: A linear encoding given by a $k \times n$ generator matrix G will have update-efficiency t if and only if G has maximum row weight t.

Proof: It is easy to see that if the maximum number of ones in any row is bounded above by t, then at most t bits need to be changed to update one bit change in the message.

On the other hand, if the code has update-efficiency t then each of the vectors $(1, 0, \dots, 0), (0, 1, \dots, 0), \dots, (0, 0 \dots, 1) \in \mathbb{F}_2^k$ must produce codeword of weight at most t under ϕ. Therefore, the generator matrix given by ϕ will have row weight at most t. ■

Hence, to optimize update-efficiency in a linear code, we seek a representation in which the maximum weight of the rows of the generator matrix is low. A linear code with a sparse basis is informally called a *low density generator matrix (LDGM)* code. Evidently, to test for update-efficiency, we need to be able to identify the sparsest basis for the code.

[3]Throughout the paper, all logarithms are base-2 unless otherwise indicated.

Turning to local repairability, there are multiple possible formal definitions. We will use what is perhaps the simplest, which insists that for each codeword symbol, there is a set of at most r codeword positions that need to be queried to recover the given symbol with certainty. The formal definition is as follows.[4]

Definition 2: A code $\mathcal{C} \subset \mathbb{F}_2^n$ has *local repairability* r, if for any $1 \le i \le n$, there exists a function $f_i : \mathbb{F}_2^r \to \mathbb{F}_2$ and indices $1 \le i_1, \ldots, i_r \le n, i_j \ne i, 1 \le j \le r$, such that for any $\boldsymbol{x} = (x_1, \ldots, x_n) \in \mathcal{C}$, $x_i = f_i(x_{i_1}, \ldots, x_{i_r})$.

It follows immediately that the local repairability of a code is related to the maximum of the weights of the rows in the parity-check matrix. Specifically, we have the following.

Proposition 2: If the maximum row-weight of a parity-check matrix of a code is r, then the code has local repairability at most r.

From this proposition, it follows that *low density parity-check (LDPC)* codes—i.e., linear codes with a parity check matrix whose rows each have a small (constant) number of nonzero entries—are locally repairable. It turns out that this property is not quite a necessary condition for local repairability. To make this a necessary condition, one needs to consider not just rows of the parity-check matrix, but the entire dual code (of the linear code). In particular, when the recovery functions f_is are linear, a necessary and sufficient condition for local repairability of linear codes is that the dual code contains low-weight codewords whose supports cover any of the n coordinates.

III. Adversarial channels

The adversarial error model is ubiquitous in the storage analysis literature. In an adversarial error model, the channel is allowed to introduce up to s errors (or $2s$ erasures), and the location of these errors can be chosen by an adversary. It is known that to correct s errors ($2s$ erasures), the minimum distance of the code needs to be at least $2s + 1$. However, if a code has update-efficiency t, then there must exist two (in fact, many more) codewords that are within distance t of each other. Hence, small update-efficiency implies poor adversarial error correction capability, and we cannot hope to find good codes if we adopt an adversarial error model. Nevertheless, before moving on to the main results of the paper, as a reference point we make some observations of foundational nature regarding adversarial error models.

In a code with minimum pairwise distance between codewords d, the update-efficiency has to be at least d, because the nearest codeword is at least distance d away. That is, if the update-efficiency of the code $\mathcal{C}$ is denoted by $t(\mathcal{C})$, then

$$t(\mathcal{C}) \ge d(\mathcal{C}),$$

where $d(\mathcal{C})$ is the minimum distance of the code. Our main observation is that the above bound is in fact achievable with the best possible parameters of a linear code.

[4]A weaker definition could allow adaptive queries, i.e., the choice of which r positions to query could depend on the values of previously queried symbols. As another variation, one could consider relaxing our requirement that we be able to recover the codeword symbol with certainty, and require only that we recover it with some probability significantly greater than $1/2$, corresponding to probabilistic recovery. Our results can be extended to these adaptive and probabilistic recovery without much work.

To develop our result, in light of Proposition 1, the following theorem from [14] is useful.

Theorem 3: Any binary linear code of length n, dimension k and distance d has a generator matrix consisting of rows of weight $\le d + s$, where

$$s = \left(n - \sum_{j=0}^{k-1} \left\lceil \frac{d}{2^j} \right\rceil\right)$$

is a nonnegative integer.
The fact that s is a non-negative integer also follows from the well-known Griesmer bound [15], which states that for any linear code with dimension k, distance d, and length n must satisfy $n \ge \sum_{j=0}^{k-1} \lceil d/2^j \rceil$.

Corollary 4: For any linear $[n, k, d]$ code $\mathcal{C}$ with update-efficiency t,

$$d \le t \le d + \left(n - \sum_{j=0}^{k-1} \left\lceil \frac{d}{2^j} \right\rceil\right).$$

It is clear that for codes achieving the Griesmer bound with equality, the update-efficiency is precisely the minimum distance, i.e., the best possible. There are a number of families of codes that achieve the Griesmer bound. For examples of such families and their characterizations, see [16], [17].

Example: Suppose $\mathcal{C}$ is a $[n = 2^m - 1, k = 2^m - 1 - m, 3]$ Hamming code. For this code

$$t(\mathcal{C}) \le 3 + (n - 3 - 2 - (k - 2)) = n - k = m = \log(n + 1).$$

One can easily achieve update-complexity $1 + \log(n + 1)$ for Hamming codes: simply bring any $k \times n$ generator matrix of Hamming code into systematic form, resulting in the maximum weight of a row being bounded above by $1 + (n - k) = 1 + \log(n + 1)$. This special case was mentioned in [5]. Reasoned another way, since the generator polynomial of a Hamming code (cyclic code) has degree m, the maximum row-weight of a generator of a Hamming code is at most $m + 1 = \log(n + 1) + 1$.

However, we can do even better by explicitly constructing a generator matrix for the Hamming code in the following way. Let us index the columns of the generator matrix by $1, 2, \ldots, 2^m - 1$, and use the notation (i, j, k) to denote the vector with exactly three 1's, located at positions i, j, and k. Then, the Hamming code has a generator matrix given by the row vectors $(i, 2^j, i + 2^j)$ for $1 \le j \le m - 1, 1 \le i < 2^j$. This shows that for all n, Hamming codes have update-efficiency only 3.

To prove the above result without explicitly constructing a generator matrix, and to derive some other consequences, the following theorem from [18] is convenient.

Theorem 5: Any $[n, k, d]$ binary linear code can be transformed into a code with the same parameters that has a generator matrix consisting of only weight d rows.
The implication of this theorem is the following: if there exists an $[n, k, d]$ linear code, then there exists an $[n, k, d]$ linear code with update-efficiency d.

The proof of [18] can be expressed as an algorithm that transforms any linear code, given its parameters $[n, k, d]$ and a generator matrix, into an update-efficient linear code (a code with update-efficiency equal to the minimum distance). The algorithm, in time possibly exponential in n, produces a new generator matrix with all rows having weight d. It is of interest to find a polynomial time (approximation) algorithm for the procedure, that is, a generator matrix with all rows having weight within $d(1+\epsilon)$ for some small ϵ.

On the other hand, the above theorem says that there exists a linear $[n = 2^m - 1, k + 2^m - 1 - m, 3]$ code that has update-efficiency only 3. All codes with these parameters are equivalent to the Hamming code with the same parameters up to a permutation of coordinates [19], providing an alternate proof that Hamming codes have update-efficiency 3.

More generally, analysis of update-efficiency for BCH codes and other linear codes is of independent interest. In general, finding a sparse basis for a linear code given its generator matrix appears to be a hard problem, although the actual complexity class of the problem merits further investigation. Recently, a sparse basis is presented for 2-error-correcting BCH codes in [20].

We emphasize that the previous remarks, although of theoretical interest, are fairly strong negative results suggesting that update efficiency and error correction are fundamentally incompatible requirements. Although this is the case for adversarial models, if we allow randomization so that the code can be chosen in a manner unknown to the adversary, then it is possible to fool the adversary. In fact, with a randomized code it is possible to correct roughly pn adversarial errors with a code rate close to the capacity of BSC(p) using ideas put forward in [21]. The randomized code can be chosen to simultaneously have good update efficiency, as shown in the case of erasure channels by [5].

In the rest of the paper, instead of choosing a code at random to fool an adversary, we consider the classical information theoretic scenario of a random, rather than an adversarial, channel, although we note that our results easily extend to the case of using randomization to defeat an adversary.

IV. Existence of good codes

In this section, we show, in a rather simple way, that there exist linear codes of length n that

1) have rate ϵ less than capacity, $\epsilon > 0$,
2) achieve arbitrarily small probability of error,
3) have update-efficiency $O(\log n)$ and
4) have local repairability $O(\log 1/\epsilon)$.

It is relatively easy to construct a code with local repairability $O(\log 1/\epsilon)$ that achieves capacity over the BSC or BEC with an ϵ gap. One can in principle choose the rows of the parity-check matrix randomly from all low weight vectors, and argue that this random ensemble contains many codes that achieve the capacity of the binary symmetric channel (BSC) up to an additive term ϵ. Indeed, LDPC codes achieve the capacity of the binary symmetric channel [22].

Similarly, one may try to construct a low row-weight generator matrix randomly to show that the ensemble average performance achieves capacity. In this direction, some steps have been taken in [23]. However, these constructions fail to achieve local repairability and update-efficiency simultaneously. Also, in [24], parallel to a part of our work [25], a rateless code construction was proposed that achieves both $O(\log k)$ update-efficiency and local repairability, k being the dimension of the code. Below, we describe one simple and intuitive construction that simultaneously achieves $O(\log k)$ update efficiency and $O(\log 1/\epsilon)$ local repairability, where ϵ is the gap to capacity.

It is known that for for every $\epsilon > 0$ and any sufficiently large m, there exists a linear code of length m and rate $1 - h_{\rm B}(p) - \epsilon$ that has probability of incorrect decoding at most $2^{-E(p,\epsilon)m}$. There are numerous evaluations of this result and estimates of $E(p,\epsilon) > 0$. We refer the reader to [26] as an example. Below, $m, n, mR, nR, n/m$ are assumed to be integers. Floor and ceiling functions should be used where appropriate. However, we avoid using them to maintain clarity in the argument, unless the meaning is not obvious from the context.

Let $m = (1+\alpha)/E(p,\epsilon)\log n$, with $\epsilon, \alpha > 0$. We know that for sufficiently large n, there exists a linear code $\hat{\mathcal{C}}$ given by the $mR \times m$ generator matrix $\hat{G}$ with rate $R = 1 - h_{\rm B}(p) - \epsilon$ that has probability of incorrect decoding at most $2^{-E(p,\epsilon)m}$.

Let G be the $nR \times n$ matrix that is the Kronecker product of $\hat{G}$ and the $n/m \times n/m$ identity matrix $I_{n/m}$, i.e.,

$$G = I_{n/m} \otimes \hat{G}.$$

Clearly a codeword of the code $\mathcal{C}$ given by G is given by n/m codewords of the code $\hat{\mathcal{C}}$ concatenated side-by-side. The probability of error of $\mathcal{C}$ is therefore, by the union bound, at most

$$\frac{n}{m}2^{-E(p,\epsilon)m} = \frac{nE(p,\epsilon)}{(1+\alpha)n^{1+\alpha}\log n} = \frac{E(p,\epsilon)}{(1+\alpha)n^{\alpha}\log n}.$$

However, notice that the generator matrix has row weight bounded above by $m = (1+\alpha)/E(p,\epsilon)\log n$. Hence, we have constructed a code with update-efficiency $(1+\alpha)/E(p,\epsilon)\log n$, and rate $1 - h_{\rm B}(p) - \epsilon$ that achieves a probability of error less than $E(p,\epsilon)/[(1+\alpha)n^{\alpha}\log n]$ over a BSC(p).

We modify the above construction slightly to produce codes that also possess good local repairability. It is known that LDPC codes achieve a positive error-exponent. That is, for every $\epsilon > 0$ and any sufficiently large m, there exist an LDPC code of length m and rate $1 - h_{\rm B}(p) - \epsilon$ that has check degree (number of 1's in a row of the parity-check matrix) at most $O(\log 1/\epsilon)$, and probability of incorrect decoding at most $2^{-E_L(p,\epsilon)m}$, for some $E_L(p,\epsilon) > 0$.[5] This code will be chosen as $\hat{\mathcal{C}}$ in the above construction, and $\hat{G}$ can be any generator matrix for $\hat{\mathcal{C}}$.

The construction now follows without further modifications. We have, $m = (1+\alpha)/E_L(p,\epsilon)\log n$, an integer, $\epsilon, \alpha > 0$, and $G = I_{n/m} \otimes \hat{G}$. This generator matrix has row weight bounded above by $m = (1+\alpha)/E_L(p,\epsilon)\log n$, so the code has update-efficiency $(1+\alpha)/E_L(p,\epsilon)\log n$, rate $1 - h_{\rm B}(p) - \epsilon$, and achieves probability of error less than $E_L(p,\epsilon)/[(1+\alpha)n^{\alpha}\log n]$ over a BSC(p).

With respect to repairability, the parity-check matrix H of the resulting code will be block-diagonal, with each block

[5]There are several works, such as [22], [27], that discuss this result. For example, we refer the reader to [27, Thm. 7 and Eq. (17)] for a derivation of the fact.

being the parity-check matrix of the code $\hat{\mathcal{C}}$. The parity-check matrix of the overall code has row-weight $O(\log 1/\epsilon)$. Hence, any codeword symbol can be recovered from at most $O(\log 1/\epsilon)$ other symbols by solving one linear equation. Therefore, we have the following result.

Theorem 6: There exists a family of linear codes $\mathcal{C}_n$ of length n and rate $1 - h_{\mathrm{B}}(p) - \epsilon$, that have probability of error over BSC(p) going to 0 as $n \to \infty$. These codes simultaneously achieve update-efficiency $O(\log n/E_L(p,\epsilon))$ and local repairability $O(\log 1/\epsilon)$.

Hence, it is possible to simultaneously achieve local repairability and update-efficiency with a capacity-achieving code on BSC(p). Note, however, that this comes with a price. Namely, the decay of probability of error with the blocklength is only polynomial, not exponential. A similar result is immediate for BEC(p).

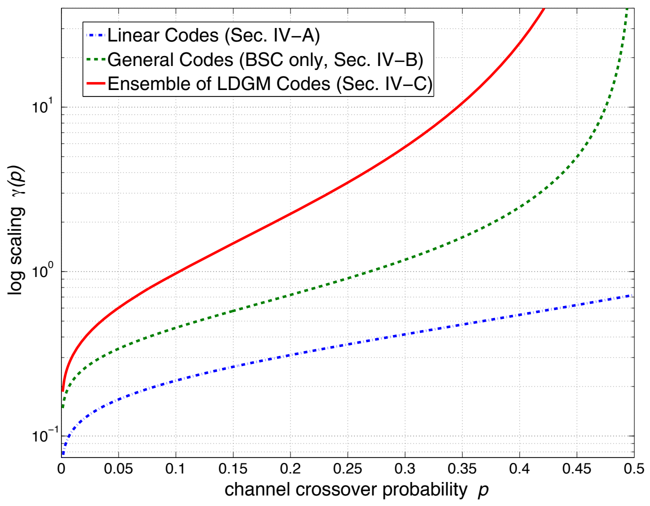

Fig. 1. The plot of constant factors of $\ln n$ from Theorems 7, 8 and 10. The fact that the bound for linear codes appears below the bound for general codes is a artifact of the bounding technique: the bound for general codes is not extendable to the BEC, but the bound for linear codes is.

V. Impossibility results for update-efficiency

In this section, we show that for suitably small γ, no code can simultaneously achieve capacity and have update-efficiency better than $\gamma \log n$, where n denotes the blocklength. More precisely, we give the following converse results.

1) *Linear codes.* Linear codes of positive rate cannot have arbitrarily small probability of error and update-efficiency better than $\gamma_1(p) \log n, \gamma_1(p) > 0$ when used over the BEC (Thm. 7). Since a BSC is degraded with respect to a BEC, this result implies same claim for BSC as well. To see that BSC(p) is a degraded version of a BEC with erasure probability $2p$, one can just concatenate BEC$(2p)$ with a channel with ternary input $\{0,1,?\}$ and binary output $\{0,1\}$, such that with probability 1 the inputs $\{0,1\}$ remain the same, and with uniform probability ? goes to $\{0,1\}$.
2) *General codes.* Any (possibly non-linear) code with positive rate cannot have update-efficiency better than $\gamma_2(p) \log n, \gamma_2(p) > 0$, and vanishing probability of error when transmitted over BSC. The value of $\gamma_2(p)$ that we obtain in this case is larger than $\gamma_1(p)$ of linear codes; moreover this result applies to more general codes than the previous (Thm. 8), but we have not been able to extend it to the BEC. It could be interesting to explore whether nonlinear codes of positive rate must have at least logarithmic update efficiency for the BEC.
3) *LDGM ensemble.* We also show that for the ensemble of LDGM codes with fixed row-weight $\gamma_3(p) \log n, \gamma_3(p) > 0$, almost all codes have probability of error approaching 1 when transmitted over a BSC (Thm. 10). The value of $\gamma_3(p)$ in this case is much larger than the previous two cases.

A plot providing the lower bound on update-efficiency of "good" codes is presented in Fig. 1. In this figure, the values of α, the constant multiplier of $\ln n$, as a function of BSC flip probability p is plotted. The plot contains results of Theorems 7, 8 and 10. Note that $\gamma_1(p), \gamma_3(p) \to \infty$ as $p \to 1/2$ for general codes (Theorem 8) and the LDGM ensemble (Theorem 10).

A. Impossibility result for linear codes

The converse result for linear codes used over a binary erasure channel is based on the observation that when the update-efficiency is low, the generator matrix G is very sparse, i.e., every row of G has very few non-zero entries. Let the random subset $I \in \{1,\dots,n\}$ denote the coordinates not erased by the binary erasure channel. Let G_I denote the submatrix of G induced by the unerased received symbols, i.e., the columns of G corresponding to I. Then, because G is sparse, it is quite likely that G_I has several all zero rows, and the presence of such rows implies a large error probability. We formalize the argument below.

Theorem 7: Consider using some linear code of length n, dimension k and update-efficiency t, specified by generator matrix G over BEC(p). Hence, all rows of G have weight at most t. Assume that for some $\epsilon > 0$,

$$t < \frac{\ln \frac{k^2}{2n\ln(1/\epsilon)}}{2\ln\frac{1}{p}}.$$

Then, the average probability of error is at least $1/2 - \epsilon$.

Proof: For linear codes over the binary erasure channel, analyzing the probability of error essentially reduces to analyzing the probability that the matrix G_I induced by the unerased columns of G has rank k (note that the rank is computed over $\mathbb{F}_2$). To show that the rank is likely to be less than k for sufficiently small t, let us first compute the expected number of all zero rows of G_I. Since every row of G has weight at most t, the expected number of all zero rows of G_I is at least kp^t. The rank of G_I, $\mathrm{rank}(G_I)$, is at most k minus the number of all zero rows, so the expected rank of G_I is at most $k - kp^t$.

Now, observe that the rank is a 1-Lipschitz functional of the independent random variables denoting the erasures introduced by the channel. Therefore, by Azuma's inequality

[28, Theorem 7.4.2], the rank of G_I satisfies

$$\Pr(\mathrm{rank}(G_I) \geq \mathbb{E}\,\mathrm{rank}(G_I) + \lambda) < e^{-\frac{\lambda^2}{2n}}.$$

Therefore,

$$\Pr(\mathrm{rank}(G_I) \geq k - kp^t + \lambda) < e^{-\frac{\lambda^2}{2n}}.$$

In particular, substituting $\lambda = kp^t$,

$$\Pr(\mathrm{rank}(G_I) = k) < e^{-\frac{k^2 p^{2t}}{2n}}.$$

Assuming the value given for t, we see that

$$\Pr(\mathrm{rank}(G_I) = k) < \epsilon.$$

Since even the maximum likelihood decoder makes an error with probability at least 0.5 when $\mathrm{rank}(G_I) < k$, this shows that when

$$t < \frac{\ln \frac{k^2}{2n\ln(1/\epsilon)}}{2\ln\frac{1}{p}},$$

the probability of error is at least $1/2 - \epsilon$. (In fact, the average error probability converges to 1. The above argument can easily be extended to show that the probability of decoding successfully is at most $e^{-\Omega(k^\delta/\log k)}$ for some $\delta > 0$, but we omit the details.) ■

B. Impossibility for general codes

Now, we prove that even nonlinear codes cannot have low update-efficiency for the binary symmetric channel. The argument is based on a simple observation. If a code has dimension k and update-efficiency t, then any given codeword has k neighboring codewords within distance t, corresponding to the k possible single-bit changes to the information bits. If t is sufficiently small, it is not possible to pack $k+1$ codewords into a Hamming ball of radius t and maintain a low probability of error.

Theorem 8: Consider using some (possibly non-linear) code of length n, size $2^k, k \in \mathbb{R}_+$, and update-efficiency t over BSC(p). Assume that $t \leq (1-\alpha)\log k/\log((1-p)/p)$, for some $\alpha > 0$. Then, the average probability of error is at least $1 - o(1)$, where $o(1)$ denotes a quantity that goes to zero as $k \to \infty$.

Proof: First, we show that a code consisting of $k+1$ codewords contained in a Hamming ball of radius t has large probability of error. Instead of analyzing BSC(p), consider the closely related channel where exactly w uniformly random errors are introduced, where $w + t \leq n/2$. For this channel, subject to the constraint that the $k+1$ codewords are contained in a Hamming ball of radius t, the average probability of error is at least

$$1 - \frac{(2t+1)\binom{n}{w+t}}{(k+1)\binom{n}{w}} \geq 1 - \frac{2t(n-w)^t}{kw^t}.$$

To see this, take $\boldsymbol{x}_1, \dots, \boldsymbol{x}_{k+1}$ to be the codewords, and $B_i, i = 1, \dots, k+1$, to be the corresponding decoding regions. Without loss of generality, we can assume that 1) the ML decoder is detetrministic, so the B_i's are all disjoint, and 2) the codewords are all contained in a Hamming ball centered at the zero vector. Now, let D_i be the set of possible outputs of the channel for input $\boldsymbol{x}_i, i = 1, \dots, k+1$. The average probability of correct decoding is

$$\frac{1}{k+1}\sum_{i=1}^{k+1}\frac{|B_i \cap D_i|}{\binom{n}{w}} = \frac{1}{k+1}\frac{|\cup_i (B_i \cap D_i)|}{\binom{n}{w}} \leq \frac{|\cup_i D_i|}{(k+1)\binom{n}{w}}.$$

But

$$|\cup_i D_i| \leq \sum_{j=-t}^{t}\binom{n}{w+j} \leq (2t+1)\binom{n}{w+t}.$$

The first inequality follows because an erroneous vector can have weight at least $w - t$ and at most $w + t$. The second inequality follows because $\binom{n}{i}$ increases with $i \leq n/2$.

Now, with probability at least $1 - o(1)$, the number of errors introduced by the the binary symmetric channel is at least $pn - n^{2/3}$ and at most $pn + n^{2/3}$, and conditioned on the exact number of errors, the error vector is uniformly distributed over all strings with this number of errors[6].

If $t \leq (1-\alpha)\log k/\log((1-p)/p)$, then for $p < \frac{1}{2}$, $pn + n^{2/3} + t < n/2$, for every sufficiently large n. Therefore, the probability of error on the binary symmetric channel is at least

$$1 - \frac{2t(1-p)^t}{kp^t} + o(1) = 1 - 2t/k^\alpha + o(1).$$

Now, for each message $\boldsymbol{x}$ of the given (n,k) code with update-efficiency t, consider the subcode $\mathcal{C}_{\boldsymbol{x}}$ consisting of the $k+1$ codewords $\phi(\boldsymbol{x}), \phi(\boldsymbol{x}+\boldsymbol{e}_1), \dots, \phi(\boldsymbol{x}+\boldsymbol{e}_k)$, corresponding to the encodings of $\boldsymbol{x}$ and the k messages obtained by changing a single bit of $\boldsymbol{x}$. These codewords lie within a Hamming ball of radius t centered around $\phi(\boldsymbol{x})$. The above argument shows that even a maximum likelihood decoder has a large average probability of error for decoding the subcode $\mathcal{C}_{\boldsymbol{x}}$. Let us call this probability $P_{\mathcal{C}_{\boldsymbol{x}}}$. We claim that the average probability of error of the code $\mathcal{C}$ with maximum likelihood decoding, $P_{\mathcal{C}}$, is at least the average, over all $\boldsymbol{x}$, of the probability of error for the code $\mathcal{C}_{\boldsymbol{x}}$, up to some factor. In particular,

$$P_{\mathcal{C}} \geq \frac{k}{n}\frac{1}{|\mathcal{C}|}\sum_{\boldsymbol{x}\in\mathcal{C}} P_{\mathcal{C}_{\boldsymbol{x}}}.$$

We will now prove this claim and thus the theorem. Note that $P_{\mathcal{C}} = 1/|\mathcal{C}|\sum_{\boldsymbol{x}\in\mathcal{C}} P_{\boldsymbol{x}}$, where $P_{\boldsymbol{x}}$ is the probability of error if codeword $\boldsymbol{x}$ is transmitted. Therefore,

$$P_{\mathcal{C}_{\boldsymbol{x}}} = \frac{1}{|\mathcal{C}_{\boldsymbol{x}}|}\sum_{\boldsymbol{y}\in\mathcal{C}_{\boldsymbol{x}}} P_{\boldsymbol{y}}.$$

We have,

$$\begin{aligned}\frac{1}{|\mathcal{C}|}\sum_{\boldsymbol{x}\in\mathcal{C}} P_{\mathcal{C}_{\boldsymbol{x}}} &\leq \frac{1}{|\mathcal{C}|}\sum_{\boldsymbol{x}\in\mathcal{C}}\frac{1}{|\mathcal{C}_{\boldsymbol{x}}|}\sum_{\boldsymbol{y}\in\mathcal{C}_{\boldsymbol{x}}} P_{\boldsymbol{y}} \\ &= \frac{1}{(k+1)|\mathcal{C}|}\sum_{\boldsymbol{x}\in\mathcal{C}}\sum_{\boldsymbol{y}\in\mathcal{C}_{\boldsymbol{x}}} P_{\boldsymbol{y}} \\ &= \frac{1}{|\mathcal{C}|}\sum_{\boldsymbol{x}\in\mathcal{C}}\frac{d_{\boldsymbol{x}}}{k+1}P_{\boldsymbol{x}},\end{aligned}$$

[6]This, of course, follows from standard large deviation inequalities such as Chernoff bound; see, e.g., [29, Thm. 3.1.2].

where $d_{\boldsymbol{x}} = |\{\boldsymbol{y} : \boldsymbol{x} \in \mathcal{C}_{\boldsymbol{y}}\}| \leq n$. Hence,

$$\frac{1}{|\mathcal{C}|}\sum_{\boldsymbol{x}\in\mathcal{C}} P_{\mathcal{C}_{\boldsymbol{x}}} \leq \frac{1}{|\mathcal{C}|}\sum_{\boldsymbol{x}\in\mathcal{C}} \frac{n}{k+1}P_{\boldsymbol{x}} = \frac{n}{k+1}P_{\mathcal{C}}.$$

We conclude that the original code $\mathcal{C}$ has probability of error at least $1-o(1)$ when

$$t \leq \frac{(1-\alpha)\log k}{\log(\frac{1-p}{p})}.$$

■

Remark 1: This argument does not work for the binary erasure channel. In fact, there exist zero rate codes for the binary erasure channel with vanishing error probability and sub-logarithmic update-efficiency. Specifically, consider an encoding from k bits to 2^k bits that maps a message $\boldsymbol{x}$ to the string consisting of all 0's except for a single 1 in the position with binary expansion $\boldsymbol{x}$. Repeat every symbol of this string c times to obtain the final encoding $\phi(\boldsymbol{x})$. The update-efficiency is $2c$, since every codeword has exactly c 1's, and different codewords never have a nonzero entry in the same position. Since the location of a nonzero symbol uniquely identifies the message, the error probability is at most the probability that all c 1's in the transmitted codeword are erased, i.e., at most p^c. Therefore, we achieve vanishing error probability as long as $c \to \infty$, and c can grow arbitrarily slowly.

We conjecture that for positive rates, even nonlinear codes must have logarithmic update complexity for the binary erasure channel.

C. Ensemble of LDGM codes

Let us motivate the study of one particular ensemble of LDGM codes here. Suppose we want to construct a code with update-efficiency t. From proposition 1, we know that a linear code with update-efficiency t always has a generator matrix with maximum row weight t. For simplicity we consider generator matrices with all rows having weight exactly t. We look at the ensemble of linear codes with such generator matrices, and show that almost all codes in this ensemble are bad for t less than certain value. Note that any $k \times n$ generator matrix with row weight at most t can be extended to a generator matrix with block-length $n+t-1$ and row weight exactly t (by simply padding necessary bits in the last $t-1$ columns).

Let $\Gamma_{n,k,t}$ be the set of all $k \times n$ matrices over $\mathbb{F}_2$ such that each row has exactly t ones. First of all, we claim that almost all the matrices in $\Gamma_{n,k,t}$ generate codes with dimension k (i.e., the rank of the matrix is k). Indeed, we quote the following lemma from [30].

Lemma 9: Randomly and uniformly choose a matrix G from $\Gamma_{n,k,t}$. If $k \leq \Big(1-e^{-t}/\ln 2 - o(e^{-t})\Big)n$, then with probability $1-o(1)$ the rank of G is k.

This lemma, along with the next theorem, which is the main result of this section, will show the fact claimed at the start of this section.

Theorem 10: Fix an $0 < \alpha < 1/2$. For at least a $1 - t^2 n^{2\alpha}/(n-t)$ proportion of the matrices in $\Gamma_{n,k,t}, k \geq n^{\alpha}$, the corresponding linear code has probability of error at least $1 - e^{-\frac{n^\alpha}{\sqrt{t}}2^{-\lambda_p t}}$ over a BSC(p), for $p < 1/2$ and $\lambda_p = -1 - 1/2\log p - 1/2\log(1-p) > 0$.

The proof of this theorem is deferred until later in this section. This theorem implies that for any $\alpha < 1/2$, most codes in the random ensemble of codes with fixed row-weight (and hence update-efficiency) $t < \alpha/\lambda_p \log n$ have probability of error bounded away from 0 for any positive rate. Indeed, we have the following corollary.

Corollary 11: For at least $1-o(1)$ proportion of all linear codes with fixed t-row-weight generator matrix, $t < (\alpha/\lambda_p)\log n$, $\alpha < \frac{1}{2}$, and dimension $k > n^{\alpha}$, the probability of error is $1-o(1)$ over a BSC(p), for $0 < p \leq 1/2$.

In particular, this shows that almost all linear codes with fixed row weight $t < 1/(2\lambda_p)\log n$ and rate greater than $1/\sqrt{n}$ are bad (result in high probability of error).

Proof of Corollary 11: From Lemma 9, it is clear that a $1-o(1)$ proportion of all codes in $\Gamma_{n,k,t}$ have rank k. Hence, if a $1-o(1)$ proportion of codes in $\Gamma_{n,k,t}$ have some property, a $1-o(1)$ proportion of codes with t-row-weight generator matrix and dimension k also have that property.

Now, plugging in the value of t in the expression for probability of error in Theorem 10, we obtain the corollary. ■

To prove Theorem 10, we will need the following series of lemmas.

Lemma 12: Let $\boldsymbol{x} \in \{0,1\}^n$ be a vector of weight t. Let the all-zero vector of length n be transmitted over a BSC with flip probability $p < 1/2$. If the received vector is $\boldsymbol{y}$, then

$$\Pr(\mathrm{wt}(\boldsymbol{y}) > d_{\mathrm{H}}(\boldsymbol{x},\boldsymbol{y})) \geq \frac{1}{\sqrt{t}}2^{-\lambda_p t},$$

where $\lambda_p = -1 - 1/2\log p - 1/2\log(1-p) > 0$.

Proof: Let $I \subset [n]$ be the support of $\boldsymbol{x}$. We have $|I| = t$. Now, $\mathrm{wt}(\boldsymbol{y}) > d_{\mathrm{H}}(\boldsymbol{x},\boldsymbol{y})$ whenever the number of errors introduced by the BSC in the coordinates I is $> t/2$. Hence,

$$\Pr(\mathrm{wt}(\boldsymbol{y}) > d_{\mathrm{H}}(\boldsymbol{x},\boldsymbol{y})) = \sum_{i > t/2}\binom{t}{i}p^i(1-p)^{t-i}$$
$$> \binom{t}{t/2}p^{t/2}(1-p)^{t-t/2} \geq \frac{1}{\sqrt{t}}2^{-\lambda_p t}.$$

■

Lemma 13: Suppose two random vectors $\boldsymbol{x},\boldsymbol{y} \in \{0,1\}^n$ are chosen independently and uniformly from the set of all length-n binary vectors of weight t. Then,

$$\Pr(\mathrm{supp}(\boldsymbol{x}) \cap \mathrm{supp}(\boldsymbol{y}) = \emptyset) > 1 - \frac{t^2}{n-t+1}.$$

Proof: The probability in question equals

$$\begin{aligned}\frac{\binom{n-t}{t}}{\binom{n}{t}} &= \frac{((n-t)!)^2}{(n-2t)!n!} \\ &= \frac{(n-t)(n-t-1)(n-t-2)\dots(n-2t+1)}{n(n-1)(n-2)\dots(n-t+1)} \\ &= \Big(1-\frac{t}{n}\Big)\Big(1-\frac{t}{n-1}\Big)\dots\Big(1-\frac{t}{n-t+1}\Big) \\ &> \Big(1-\frac{t}{n-t+1}\Big)^t \geq 1-\frac{t^2}{n-t+1}.\end{aligned}$$

In the last step we have truncated the series expansion of $\left(1-\frac{t}{n-t+1}\right)^t$ after the first two terms. The inequality will be justified if the terms of the series are decreasing in absolute value. Let us verify that to conclude the proof. In the following X_i denote the ith term in the series, $0 \le i \le t$.

$$\frac{X_{i+1}}{X_i} = \frac{\binom{t}{i+1}}{\binom{t}{i}} \cdot \frac{t}{n-t+1} = \frac{t-i}{i+1} \cdot \frac{t}{n-t+1} \le 1,$$

for all $i \le t-1$. ■

Lemma 14: Let us choose any $n^\alpha, 0 < \alpha < 1/2$, random vectors of weight t independently and uniformly from the set of weight-t vectors. Denote the vectors by $\boldsymbol{x}_i, 1 \le i \le n^\alpha$. Then,

$$\Pr(\forall i \neq j, \operatorname{supp}(\boldsymbol{x}_j) \cap \operatorname{supp}(\boldsymbol{x}_i) = \emptyset) \ge 1 - \frac{t^2 n^{2\alpha}}{n-t}.$$

This implies all of the vectors have disjoint supports with probability at least $1 - t^2 n^{2\alpha}/(n-t)$.

Proof: Form Lemma 13, for any pair of randomly and uniformly chosen vectors, the probability that they have overlapping support is at most $t^2/(n-t)$. The claim follows by taking a union bound over all $\binom{n^\alpha}{2}$ pairs of the randomly chosen vectors. ■

Now, we are ready to prove Theorem 10.

Proof of Theorem 10: We begin by choosing a matrix G uniformly at random from $\Gamma_{n,k,t}$. This is equivalent of choosing each row of G uniformly and independently from the set of all n-length t-weight binary vectors. Now, $k > n^\alpha$, hence there exists n^α vectors among the rows of G such that any two of them have disjoint support with probability at least $1 - t^2 n^{2\alpha}/(n-t)$ (from Lemma 14). Hence, for at least a proportion $1 - t^2 n^{2\alpha}/(n-t)$ of matrices of $\Gamma_{n,k,t}$, there are n^α rows with disjoint supports. Suppose G is one such matrix. It remains to show that the code $\mathcal{C}$ defined by G has probability of error at least $n^\alpha 2^{-\lambda_p t}/\sqrt{t}$ over BSC(p).

Suppose, without loss of generality, that the all zero vector is transmitted over a BSC(p), and $\boldsymbol{y}$ is the vector received. We know that there exists at least n^α codewords of weight t such that all of them have disjoint support. Let $\boldsymbol{x}_i, 1 \le i \le n^\alpha$, be those codewords. Then, the probability that the maximum likelihood decoder incorrectly decodes $\boldsymbol{y}$ to $\boldsymbol{x}_i$ is

$$\Pr(\operatorname{wt}(\boldsymbol{y}) > d_{\mathrm{H}}(\boldsymbol{x}_i, \boldsymbol{y})) \ge \frac{1}{\sqrt{t}} 2^{-\lambda_p t}$$

from Lemma 12. As the codewords $\boldsymbol{x}_1, \dots \boldsymbol{x}_{n^\alpha}$ have disjoint supports, the probability that the maximum likelihood decoder incorrectly decodes to any one of them is at least

$$1 - \left(1 - \frac{1}{\sqrt{t}} 2^{-\lambda_p t}\right)^{n^\alpha} \ge 1 - e^{-\frac{n^\alpha}{\sqrt{t}} 2^{-\lambda_p t}}.$$

■

Remark 2: Theorem 10 is also true for the random ensemble of matrices where the entries are independently chosen from $\mathbb{F}_2$ with $\Pr(1) = t/n$.

VI. Impossibility result for local repairability

In this section, we develop the converse for local repairability over the BEC. We show that any code with a given local repairability must have rate bounded away from capacity to provide arbitrarily small probability of error, when used over the BEC. In particular, for any code, including nonlinear codes, recovery complexity at a gap of ϵ to capacity on the BEC must be at least $\Omega(\log 1/\epsilon)$, proving that the above LDPC construction is simultaneously optimal to within constant factors for both update-efficiency and local repairability.

The intuition for the converse is that if a code has low local repairability complexity, then codeword positions can be predicted by looking at a few codeword symbols. As we will see, this implies that the code rate must be bounded away from capacity, or the probability of error approaches 1. In a little more detail, for an erasure channel, the average error probability is related to how the codewords behave under projection onto the unerased received symbols. Generally, different codewords may result in the same string under projection, and without loss of generality, the ML decoder can be assumed to choose a codeword from the set of codewords matching the received channel output in the projected coordinates uniformly at random. Thus, given a particular erasure pattern induced by the channel, the average probability of decoding success for the ML decoder is simply the number of different codeword projections, divided by 2^{Rn}, the size of the codebook. We now show that the number of different projections is likely to be far less than 2^{Rn}.

Theorem 15:

Let $\mathcal{C}$ be a code of length n and rate $1-p-\epsilon$, $\epsilon > 0$, that achieves probability of error bounded away from 1, when used over BEC(p). Then, the local repairability of $\mathcal{C}$ is at least $c \log 1/\epsilon$, for some constant $c > 0$ and n sufficiently large.

Proof: Let $\mathcal{C}$ be a code of length n and size 2^{nR} that has local recoverability r. Let T be the set of coordinates with the property that the query positions required to recover these coordinates appear before them. To show that such an ordering exists with $|T| \ge n/(r+1)$, we can randomly and uniformly permute the coordinates of $\mathcal{C}$. The expected number of such coordinates is then $n/(r+1)$, hence some ordering exists with $|T| \ge n/(r+1)$.

Assume $I \subseteq \{1, \dots, n\}$ is the set of coordinates erased by the BEC, and let $\bar{I} = \{1, \dots, n\} \setminus I$. Let $\boldsymbol{x} \in \mathcal{C}$ be a randomly and uniformly chosen codeword. $\boldsymbol{x}_I$ and $\boldsymbol{x}_{\bar{I}}$ denote the projection of $\boldsymbol{x}$ on the respective coordinates. We are interested in the logarithm of the number of different codeword projections onto $\bar{I}$, which we denote by $\log S(\boldsymbol{x}_{\bar{I}})$. Note that this is a random-variable with respect to the random choice of I by the BEC.

Suppose that the number of elements of T that have all r of their recovery positions un-erased is u. Then, the number of different codeword projections is unchanged if we remove these u elements from T. Hence,

$$\log S(\boldsymbol{x}_{\bar{I}}) \le |\bar{I}| - u.$$

But $\mathbb{E}u \ge (1-p)^r |T|$. Therefore,

$$\mathbb{E} \log S(\boldsymbol{x}_{\bar{I}}) \le n(1-p) - (1-p)^r \frac{n}{r+1}.$$

Observe that $\log S(\boldsymbol{x}_{\bar{I}})$ is a 1-Lipschitz functional of independent random variables (erasures introduced by the channel). This is because projecting onto one more position cannot decrease the number of different codeword projections, and at most doubles the number of projections. Therefore, we can use Azuma's inequality to conclude that

$$\Pr\Big(\log S(\boldsymbol{x}_{\bar{I}}) > n(1-p) - (1-p)^r \frac{n}{r+1} + \epsilon n\Big) \le e^{-\frac{\epsilon^2 n}{2}}.$$

If we have,

$$r \le \frac{\log \frac{1}{3\epsilon}}{\log \frac{2}{1-p}},$$

then,

$$\frac{(1-p)^r}{1+r} \ge \Big(\frac{1-p}{2}\Big)^r \ge 3\epsilon.$$

But this implies,

$$\Pr\Big(\log S(\boldsymbol{x}_{\bar{I}}) > n(1-p-2\epsilon)\Big) \le e^{-\frac{\epsilon^2 n}{2}}.$$

This means that for a suitable constant c, if $r \le c \log 1/\epsilon$, then with very high probability $\log S(\boldsymbol{x}_{\bar{I}}) \le n(1-p-2\epsilon)$. However, there are $2^{Rn} = 2^{n(1-p-\epsilon)}$ codewords, so we conclude that the probability of successful decoding is at most

$$2^{-\epsilon n} + e^{-\frac{\epsilon^2 n}{2}}.$$

Thus, we have proved that if $r \le c\log 1/\epsilon$, the probability of error converges to 1, and in particular, is larger than any $\alpha < 1$, for sufficiently large n. ■

Remark 3: Rather than considering the number of different codeword projections, we could have considered the entropy of the distribution of codeword projections onto I, which is also a 1-Lipschitz functional. This is a more general approach that can be extended to the case where local repairability can be adaptive and randomized, and only has to succeed with a certain probability (larger than .5), as opposed to providing guaranteed recovery. However, one obtains a bound of $n(1-p-2\epsilon)$ on the the entropy, so Fano's inequality only shows that the probability of error must be $\Omega(\epsilon)$, while the above analysis shows that the probability of error must be close to 1.

VII. General update-efficient codes

In this section, to develop further insights into the update-efficiency of codes, we return to the more general definition of update-efficiency that we mentioned in the introduction. Specifically, we have the following.

Definition 3: A code is called (u,t)-update-efficient if, for any u bit changes in the message, the codeword changes by at most t bits. In other words, the code $(\mathcal{C}, \phi)$ is (u,t)-*update-efficient* if for all $\boldsymbol{x} \in \mathbb{F}_2^k$, and for all $\boldsymbol{e} \in \mathbb{F}_2^k : \mathrm{wt}(\boldsymbol{e}) \le u$, we have $\phi(\boldsymbol{x}+\boldsymbol{e}) = \phi(\boldsymbol{x}) + \boldsymbol{e}'$, for some $\boldsymbol{e}' \in \mathbb{F}_2^n : \mathrm{wt}(\boldsymbol{e}') \le t$.

Obviously, an $(1,t)$-update-efficient code is a code with update-efficiency t. As discussed earlier, any (u,t)-update-efficient code must satisfy $t > d$, the minimum distance of the code. In fact, we can make a stronger statement.

Proposition 16: Suppose a (u,t)-update-efficient code of length n, dimension k, and minimum distance d exists. Then,

$$\sum_{i=0}^{u} \binom{k}{i} \le B(n,d,t),$$

where $B(n,d,w)$ is the size of the largest code with distance d such that each codeword has weight at most w.

Proof: Suppose $\mathcal{C}$ is an update-efficient code, where $\boldsymbol{x} \in \mathbb{F}_2^k$ is mapped to $\boldsymbol{y} \in \mathbb{F}_2^n$. The $\sum_{i=0}^{u}\binom{k}{i}$ different message vectors within distance u from $\boldsymbol{x}$ should map to codewords within distance t from $\boldsymbol{y}$. Suppose these codewords are $\boldsymbol{y}_1, \boldsymbol{y}_2, \dots$. Consider the vectors $\boldsymbol{y}-\boldsymbol{y}, \boldsymbol{y}_1-\boldsymbol{y}, \boldsymbol{y}_2-\boldsymbol{y}, \dots$. These must be at least distance d apart from one another and all of their weights are at most t. This proves the claim. ■

There are a number of useful upper bounds on the maximum size of constant weight codes (i.e., when the codewords have a constant weight t) that can be used to upper bound $B(n,d,t)$. Perhaps the most well-known bound is the Johnson bound [31]. An easy extension of this bound says $B(n,d,t) \le dn/(dn - 2tn + 2t^2)$, as long as the denominator is positive. However, this bound is not very interesting in our case, where we have $n \gg t \ge d$. The implications of some other bounds on $B(n,d,t)$ on the parameters of update-efficiency is a topic of independent interest.

Note that any code with update-efficiency t is a (u,ut)-update-efficient code. Hence, from Section IV, we can construct an $(u, O(u\log n))$ update-efficient code that achieves the capacity of a BSC(p). On the other hand one expects a converse result of the form

$$\sum_{i=0}^{u}\binom{k}{i} \le K(n,t,p),$$

where $K(n,t,p)$ is the maximum size of a code with codewords having weight bounded by t that achieves arbitrarily small probability of error. Indeed, just by emulating the proof of Theorem 8, we obtain the following result.

Theorem 17: Consider using some (possibly non-linear) (u,t)-update-efficient code of length n, and dimension (possibly fractional) k over BSC(p). Assume that

$$t \le \frac{(1-\alpha)\log \sum_{i=0}^{u}\binom{k}{i}}{\log(1-p)/p},$$

for any $\alpha > 0$. Then, the average probability of error is at least $1 - o(1)$, where $o(1)$ denotes a quantity that goes to zero as $k \to \infty$.

This shows that the $(u, O(u\log n))$ update-efficient code constructed by the method of Section IV, is almost optimal for $u \ll n$.

Remark 4 (Bit error rate and error reduction codes): Suppose we change our model of update-efficient code in the following way (limited to only this remark). The encoding map $\phi : \mathbb{F}_2^k \to \mathbb{F}_2^n$ and the decoding map $\theta : \mathbb{F}_2^n \to \mathbb{F}_2^k$, must satisfy the property that for a random error vector $\boldsymbol{e}$ induced by the BSC(p) and any $\boldsymbol{x} \in \mathbb{F}_2^n$, $d_{\mathrm{H}}(\theta(\phi(\boldsymbol{x}) + \boldsymbol{e}), \boldsymbol{x}) \sim o(k)$ with high probability. This can be thought of as an error-reducing code, or a code with low message bit error rate [32]. Under this notion, error-reducing codes are update-efficient. When the message changes $\le u$ bits from the previous state $\boldsymbol{x} \in \mathbb{F}_2^k$, we do not change the codeword. Then, the decoder output will be within $o(k) + u$ bits from the original message.

VIII. Rate-Distortion Counterparts

In this paper, we have focused on error correcting codes possessing good update efficiency and local repairability properties. In principle, these properties are also applicable to the problem of lossy source coding. Informally, lossy source coding is concerned with optimally compressing a source so that the source can be reconstructed up to a specified distortion from its compressed representation. We formulate the problem more precisely below.

Let $X_1, X_2, \dots, X_n$ be a sequence of i.i.d. random variables distributed according to P_X, a probability distribution over the alphabet $\mathcal{X}$. Let $\mathcal{Y}$ denote the reconstruction alphabet, and let

$$d : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}_+ \cup \{0\}$$

denote the distortion function, i.e., $d(x, y)$ is the penalty incurred by reconstructing a given source symbol x to the value y. We define a distortion function for sequences as the average per-symbol distortion, i.e.,

$$d(X^n, Y^n) = \frac{1}{n} \sum_{i=1}^{n} d(X_i, Y_i).$$

The lossy source coding problem considers the following situation. Given the random sequence X^n, an encoder computes a string of Rn bits. The decoder, based on this string, computes a reconstruction Y^n. An encoder/decoder pair is admissible if $E[d(X^n; Y^n)] \le D$, where D is a parameter characterizing the tolerable distortion, and the expectation is with respect to the distribution of X^n and any randomness used by the encoder and decoder. The goal of lossy source coding is to minimize the rate R of the encoder over all admissible encoder/decoder pairs. As usual in information theory, we consider asymptotic performance as the source sequence length $n \to \infty$. In this limit, optimal performance is characterized by the rate-distortion function $R(D)$, which specifies the minimum possible rate as a function of the tolerable distortion D. For our purposes, the above brief introduction to lossy source coding will be sufficient; see, e.g., [29] for a more thorough development.

We now turn to defining update-efficiency and local repairability for lossy source codes. Both of these properties are defined analogously to the earlier definitions given for error correcting codes. In more detail, update-efficiency is measured by the maximum number of bits of the Rn bit encoding of the source that change when the source is changed slightly. For example, when $\mathcal{X}$ is binary, we define update-efficiency as the maximum number of bits of the encoding that change when exactly one bit of X^n is changed. Depending on the application, more general update models might be appropriate. For example, one could define a broader class of allowed source distortions than just changing a single element of X^n, and measure the number of bits of the encoding that change when the source is distorted according to any of the allowed source distortions. We leave a thorough investigation of different formulations of update-efficiency for future work.

We define the local repairability of a lossy source code as the maximum number of bits of the Rn bit source encoding that must be queried in order to recover any particular symbol Y_i of the source reconstruction. That is, a lossy source code has local repairability t if, for all indices i, at most t bits of the source encoding must be read in order to compute Y_i.

The main questions to be asked, in the spirit of this paper, are 1) if we allow a compression rate slightly above the optimal rate specified by rate-distortion function, i.e., rate $R(D)+\epsilon$, what is the best possible local recoverability, and 2) what is the best possible update-efficiency (again, as a function of ϵ)? As a simple example, we consider these questions in the context of compressing a uniform binary source under Hamming distortion. In more detail, this is a special case of the lossy source coding problem where the X_i are uniformly distributed random bits, $\mathcal{Y} = \{0, 1\}$, and the distortion function is given by $d(x, y) = 1$ if $x \ne y$, and $d(x, y) = 0$ if $x = y$. In other words, the distortion $d(X^n; Y^n)$ is simply the (fractional) Hamming distance between X^n and Y^n.

For this case, we briefly describe known results that allow us to make some progress on questions 1 and 2. First, it can be shown that the local repairability must grow as $\Omega(\log(1/\epsilon))$, that is, for some i, at least $\Omega(\log(1/\epsilon))$ bits of the source encoding must be queried in order to recover Y_i. This is a corollary of known results for LDGM codes (e.g., [13, Theorem 5.4.1]; although the theorem is stated for linear codes, the proof given there applies to arbitrary (even non-linear) codes). Regarding achievability, it is known that LDGM codes can achieve $O(\log(1/\epsilon))$ local repairability (e.g., [13, Sec. 5.4] or [33, Theorem 1]), so known results characterize the local repairability up to a constant factor.

Update-efficiency, on the other hand, remains an open question, even for this simple model. As a crude upper bound, we observe that update-efficiency of $O(1/\epsilon \log(1/\epsilon))$ can be achieved via random codes. Specifically, the following lemma is a straightforward consequence of, e.g., [34, Theorem 1].

Lemma 18: For any $0 \le D \le .5$, there exists a lossy source code with length $n = O(1/\epsilon \log(1/\epsilon))$ achieving rate at most $R(D) + \epsilon$ (in particular, a random code has this property with high probability).

Armed with this lemma, we copy the strategy from Section IV. Given a large n, we construct a length n lossy source code by splitting the n-bit source into blocks of length $O(1/\epsilon \log(1/\epsilon))$, and applying the code from Lemma 18 to each block. Changing one bit of the source only affects the corresponding block, so changing one bit of the source requires updating at most $O(1/\epsilon \log(1/\epsilon))$ bits of the source encoding. The length n code clearly has the same compression rate R as the base code of Lemma 18. Also, by linearity of expectation, the distortion achieved by the length n code is identical to the distortion of the base code. Therefore, $O(1/\epsilon \log(1/\epsilon))$ update-efficiency is achievable. However, it is unclear that this is optimal. In particular, we are unaware of any lower bound showing that the update-efficiency has to scale with ϵ at all. A thorough study of local recoverability and update-efficiency for lossy source coding is left for future work.

IX. Concluding Remarks: Extensions to Larger Alphabets

Although our results are derived for binary-input channels, as opposed to the larger alphabet channel models usually considered for distributed storage, our proofs generalize in a straightforward manner. The q-ary generalizations for BSC and BEC are respectively the *q-ary symmetric channel* and *q-ary erasure channel*; formal definitions appear in, e.g., [35, Sec. 1.2 and Sec. 1.5.3].

The existential result of Section IV extends to the case of q-ary channels. See, [26, Sec. IIIB, Remark 6] for more detail on how the error-exponent result for BSC extends to q-ary case. There are also results concerning the error-exponent of q-ary *low density* codes that can be used to extend Theorem 6. The result one can most directly use is perhaps [36].

The converse results for Section V and VI, in particular Theorem 7, Theorem 8 and Theorem 15 can easily be stated for the case of q-ary channel. The observations regarding adversarial error case of Section III are also extendable to q-ary case in a straight-forward manner.

Acknowledgment

A. M. thanks Alexander Barg, Yury Polyanskiy and Barna Saha for useful discussions.

References

[1] A. G. Dimakis, P. B. Godfrey, Y. Wu, M. J. Wainwright, and K. Ramchandran, "Network coding for distributed storage systems," *IEEE Trans. Inform. Theory*, vol. 56, no. 9, pp. 4539–4551, Sep. 2010.

[2] A. S. Rawat, S. Vishwanath, A. Bhowmick, and E. Soljanin, "Update efficient codes for distributed storage," in *Proc. Int. Symp. Inform. Theory*, St. Petersburg, Russia, July 2011, pp. 1457–1461.

[3] L. Xu and J. Bruck, "X-code: MDS array codes with optimal encoding," *IEEE Trans. Inform. Theory*, vol. 45, no. 1, pp. 272–276, Jan. 1999.

[4] C. Jin, H. Jiang, D. Feng, and L. Tian, "P-Code: A new RAID-6 code with optimal properties," in *Proc. ACM Int. Conf. Supercomputing*, 2009, pp. 360–369.

[5] N. P. Anthapadmanabhan, E. Soljanin, and S. Vishwanath, "Update-efficient codes for erasure correction," in *Proc. Allerton Conf. Commun., Contr., Computing*, Monticello, IL, Oct. 2010, pp. 376–382.

[6] P. Gopalan, C. Huang, H. Simitci, and S. Yekhanin, "On the locality of codeword symbols," *IEEE Trans. Inform. Theory*, vol. 58, no. 11, pp. 6925–6934, Nov. 2012.

[7] D. S. Papailiopoulos and A. G. Dimakis, "Locally repairable codes," in *Proc. Int. Symp. Inform. Theory*, Cambridge, MA, July 2012, pp. 2771–2775.

[8] V. Cadambe and A. Mazumdar, "An upper bound on the size of locally recoverable codes," in *Proc. IEEE Int. Symp. Network Coding*, 2013.

[9] G. M. Kamath, N. Prakash, V. Lalitha, P. V. Kumar, N. Silberstein, A. S. Rawat, O. O. Koyluoglu, and S. Vishwanath, "Explicit MBR all-symbol locality codes," 2013, preprint, arXiv:1302.0744.

[10] A. S. Rawat, O. O. Koyluoglu, N. Silberstein, and S. Vishwanath, "Optimal locally repairable and secure codes for distributed storage systems," 2012, preprint, arXiv:1210.6954.

[11] I. Tamo, D. S. Papailiopoulos, and A. G. Dimakis, "Optimal locally repairable codes and connections to matroid theory," 2013, preprint, arXiv:1301.7693.

[12] A. Montanari and E. Mossel, "Smooth compression, Gallager bound and nonlinear sparse-graph codes," in *Proc. Int. Symp. Inform. Theory*, Toronto, Canada, July 2008, pp. 2474–2478.

[13] V. B. Chandar, "Sparse graph codes for compression, sensing, and secrecy," Ph.D. dissertation, Massachusetts Institute of Technology, Cambridge, MA, 2010.

[14] S. M. Dodunekov and N. L. Manev, "An improvement of the Griesmer bound for some small minimum distances," *Discrete Applied Mathematics*, vol. 12, no. 2, pp. 103–114, Dec. 1985.

[15] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes*. North-Holland, 1997.

[16] B. I. Belov, V. N. Logachev, and V. P. Sandimirov, "Construction of a class of linear binary codes achieving the Varshamov-Griesmer bound," *Problemy Peredachi Informatsii*, vol. 10, no. 3, pp. 36–44, 1974.

[17] T. Helleseth, "New constructions of codes meeting the Griesmer bound," *IEEE Trans. Inform. Theory*, vol. 29, no. 3, pp. 434–439, Mar. 1983.

[18] J. Simonis, "On generator matrices of codes," *IEEE Trans. Inform. Theory*, vol. 38, no. 2, p. 516, Feb. 1992.

[19] W. C. Huffman and V. Pless, *Fundamentals of Error-Correcting Codes*. Cambridge University Press, 2003.

[20] E. Grigorescu and T. Kaufman, "Explicit low-weight bases for BCH codes," *IEEE Trans. Inform. Theory*, vol. 58, no. 1, pp. 78–81, Jan. 2012.

[21] M. Langberg, "Private codes or succinct random codes that are (almost) perfect," in *Proc. Conf. Found. Comp. Sci. (FOCS)*, vol. 4, 2004, pp. 325–334.

[22] R. Gallager, "Low-density parity-check codes," *IRE Trans. Inform. Theory*, vol. 8, no. 1, pp. 21–28, Jan. 1962.

[23] A. M. Kakhaki, H. K. Abadi, P. Pad, H. Saeedi, F. Marvasti, and K. Alishahi, "Capacity achieving linear codes with binary sparse generating matrices," 2011, preprint, arXiv:1102.4099.

[24] M. Asteris and A. G. Dimakis, "Repairable fountain codes," in *Proc. Int. Symp. Inform. Theory*, Cambridge, MA, July 2012, pp. 1752–1756.

[25] A. Mazumdar, G. W. Wornell, and V. Chandar, "Update efficient codes for error correction," in *Proc. Int. Symp. Inform. Theory*, Cambridge, MA, July 2012, pp. 1558–1562.

[26] A. Barg and G. D. Forney Jr., "Random codes: Minimum distances and error exponents," *IEEE Trans. Inform. Theory*, vol. 48, no. 9, pp. 2568–2573, Sep. 2002.

[27] G. Miller and D. Burshtein, "Bounds on the maximum-likelihood decoding error probability of low-density parity-check codes," *IEEE Trans. Inform. Theory*, vol. 47, no. 7, pp. 2696–2710, July 2001.

[28] N. Alon and J. H. Spencer, *The Probabilistic Method*. New York, NY: Wiley, 2004.

[29] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. New York, NY: Wiley-interscience, 2012.

[30] N. J. Calkin, "Dependent sets of constant weight binary vectors," *Combinatorics, Probability, Computing*, vol. 6, no. 3, pp. 263–271, 1997.

[31] S. M. Johnson, "A new upper bound for error-correcting codes," *IRE Trans. Inform. Theory*, vol. 8, no. 4, pp. 203–207, Apr. 1962.

[32] Y. Kochman, A. Mazumdar, and Y. Polyanskiy, "The adversarial joint source-channel problem," in *Proc. Int. Symp. Inform. Theory*, Cambridge, MA, July 2012, pp. 2112–2116.

[33] M. J. Wainwright, E. Maneva, and E. Martinian, "Lossy source compression using low-density generator matrix codes: Analysis and algorithms," *IEEE Trans. Inform. Theory*, vol. 56, no. 3, pp. 1351–1368, Mar. 2010.

[34] Z. Zhang, E.-H. Yang, and V. K. Wei, "The redundancy of source coding with a fidelity criterion I: Known statistics," *IEEE Trans. Inform. Theory*, vol. 43, no. 1, pp. 71–91, Jan. 1997.

[35] R. M. Roth, *Introduction to Coding Theory*. Cambridge University Press, 2006.

[36] E. Hof, I. Sason, and S. Shamai, "Performance bounds for nonbinary linear block codes over memoryless symmetric channels," *IEEE Trans. Inform. Theory*, vol. 55, no. 3, pp. 977–996, Mar. 2009.

Arya Mazumdar (S'05-M'13) is an assistant professor in University of Minnesota-Twin Cities (UMN). Before coming to UMN, he was a postdoctoral scholar at the Massachusetts Institute of Technology (MIT). He received the Ph.D. degree in Electrical and Computer Engineering from the University of Maryland, College Park, in 2011.

Arya is a winner of the 2010 IEEE ISIT Student Paper Award. He is also the recipient of the Distinguished Dissertation Fellowship Award, 2011, at the University of Maryland. He spent the summers of 2008 and 2010 at the Hewlett-Packard Laboratories, Palo Alto, CA, and IBM Almaden Research Center, San Jose, CA, respectively. Arya's research interests include error-correcting codes, reliable memory and storage, and information theory.

Venkat Chandar received S.B. degrees in EECS and mathematics in 2006, an M. Eng. in EECS in 2006, and a Ph.D. in EECS in 2010, all from MIT. His current research interests include coding theory and algorithms, with an emphasis on the construction and analysis of sparse graph codes for various problems related to communication, compression, sensing, and information theoretic secrecy.

Gregory W. Wornell (S'83-M'91-SM'00-F'04) received the B.A.Sc. degree in electrical engineering from the University of British Columbia, Vancouver, BC, Canada, and the S.M. and Ph.D. degrees in electrical engineering and computer science from the Massachusetts Institute of Technology, Cambridge, MA, in 1985, 1987, and 1991, respectively.

Since 1991, he has been on the faculty at MIT, where he is the Sumitomo Professor of Engineering in the department of Electrical Engineering and Computer Science (EECS). He leads the Signals, Information, and Algorithms Laboratory in the Research Laboratory of Electronics, and co-chairs the EECS department graduate program. He has held visiting appointments at the former AT&T Bell Laboratories, Murray Hill, NJ, the University of California, Berkeley, CA, and Hewlett-Packard Laboratories, Palo Alto, CA.

His research interests and publications span the areas of information theory, digital communication, and signal processing, and include algorithms and architectures for wireless networks, sensing and imaging systems, multimedia applications, and aspects of computational biology and neuroscience.

Dr. Wornell has been involved in the Information Theory and Signal Processing Societies of the IEEE in a variety of capacities, and maintains a number of close industrial relationships and activities. He has won a number of awards for both his research and teaching.