Explicit Capacity-achieving Codes for Worst-Case Additive Errors

VENKATESAN GURUSWAMI*
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA 15213.
guruswami@cmu.edu

ADAM SMITH†
Computer Science & Engineering Department
Pennsylvania State University
University Park, PA 16802.
asmith@cse.psu.edu

October 2009

Abstract

For every $p \in (0, 1/2)$, we give an *explicit* construction of binary codes of rate approaching "capacity" $1 - H(p)$ that enable reliable communication in the presence of *worst-case additive errors*, caused by a channel oblivious to the codeword (but not necessarily the message). Formally, we give an efficient "stochastic" encoding $E(\cdot, \cdot)$ of messages combined with a small number of auxiliary random bits, such that for *every* message m and *every* error vector e (that could depend on m) that contains at most a fraction p of ones, w.h.p over the random bits r chosen by the encoder, m can be efficiently recovered from the corrupted codeword $E(m, r) + e$ by a decoder ***without knowledge of the encoder's randomness*** r.

Our construction for additive errors also yields explicit *deterministic* codes of rate approaching $1 - H(p)$ for the "average error" criterion: for *every* error vector e of at most p fraction 1's, *most* messages m can be efficiently (uniquely) decoded from the corrupted codeword $C(m) + e$. Note that such codes cannot be linear, as the bad error patterns for all messages are the same in a linear code. We also give a new proof of the *existence* of such codes based on list decoding and certain algebraic manipulation detection codes. Our proof is simpler than the previous proofs from the literature on arbitrarily varying channels.

*Supported in part by a David and Lucile Packard fellowship, NSF CCF-0953155, and US-Israel BSF 2008293.
†Supported in part by NSF awards 0729171 and 0747294.

1 Introduction

By Shannon's seminal work [23], we know that for the binary symmetric channel BSC_p which flips each transmitted bit independently with probability p, there exist binary codes of rate $1 - H(p) - \varepsilon$ that enable reliable information transmission with exponentially small probability of miscommunication. Here $\varepsilon > 0$ is arbitrary and $H(\cdot)$ is the binary entropy function. The quantity $1 - H(p)$ is called the (Shannon) capacity of the BSC_p channel. But what if the errors are *adversarial* and not randomly distributed? For the adversarial channel ADV_p where the channel can corrupt up to a fraction p of symbols in an *arbitrary* manner, it is known that for error-free communication to be possible, the rate has to be much smaller than the Shannon capacity $1 - H(p)$. The best rate known to date, even by non-constructive methods, equals $1 - H(2p)$ (the Gilbert-Varshamov bound). Determining the best asymptotic rate for error fraction p (equivalently, minimum relative distance $2p$) remains an important open question in combinatorial coding theory.

Shared randomness. There are some relaxations of the model, however, which enable achieving capacity even against worst-case errors. One of these is to allow randomized coding strategies where the sender and receiver share "secret" randomness (hidden from the channel) which is used to pick a coding scheme at random from a family of codes (such codes were called *private codes* in [17]). Using such strategies, it is easy to see that one can achieve the capacity $1 - H(p)$ against ADV_p (for example, by randomly permuting the symbols and adding a random offset [20, 17]). Using explicit codes achieving capacity on the BSC_p [9], we can even get such randomized codes of rate approaching $1 - H(p)$ explicitly.

The amount of shared randomness in the above setting can be reduced if we make computational assumptions on the channel [20, 6] — the encoder and decoder only need to share a private seed for a pseudorandom generator. For channels which distribute errors "evenly" (that is, without significant bursts), this seed can be communicated over the channel in a separate "setup" round by encoding it in a highly redundant code of very small rate [10]. One can also reduce the shared randomness without computational assumptions by using "list-decodable" codes together with standard message authentication techniques [17, 24]. It was also shown that one can eliminate the shared randomness and instead require that the sender know a public key (which the channel may also know) for which the receiver has the secret key [21]. To achieve capacity, however, the last two approaches require *list-decodable* codes of optimal rate, for which no explicit constructions are known.

List decoding. List decoding is another relaxation of the basic coding model that enables one to communicate on the adversarial channel ADV_p at rates close to capacity $1 - H(p)$. List decoding was introduced in the late 1950s [7, 26] and has witnessed a lot of recent algorithmic work (cf. the survey [12]). With list decoding, the decoder outputs a small list of messages that must include the correct message. Random coding arguments assert the existence of binary codes of rate $1 - H(p) - \varepsilon$ for which error-correction against ADV_p is possible if the decoder is allowed to output a list of size $O(1/\varepsilon)$ [8, 27, 13]. However, there are two significant drawbacks to list decoding as it currently stands. First, in the worst case, one has to be content with pinning down the message to a small list of possibilities. (If a small amount of auxiliary information can be communicated on a noiseless side channel or shared ahead of time, then it becomes possible to pick the correct element from the list with high probability [11, 17], but again this requires extra setup) Second, explicit constructions of binary list-decodable codes with rate close to capacity are not known; whether an explicit construction is possible remains a major open question.[1]

Assuming shared randomness, a separate setup round, or side information all go beyond the basic model

[1]For list decoding over large alphabets, capacity achieving codes were constructed in [14], but the binary case remains wide open. The constructions with the best currently known rate are given in [15], but these are still quite far from capacity.

of one-shot communication of a message from a sender to a receiver (over a noisy channel). Our motivation is to avoid shared randomness, ambiguous list output, a setup stage, etc. and construct a block code of rate close to capacity that can be used to recover the correct message from a broad class of errors (in particular, a class that siginificantly generalizes BSC_p but remains weaker than the fully adversarial channel ADV_p).

Arbitrarily varying channels (AVCs) are used to model channel uncertainty and they bridge between worst-case and random error models. AVCs are extremely well studied in the information theory literature (see [19] for a nice survey), but have not received much algorithmic attention. An AVC is specified by a finite state space $\mathcal{S}$ and the channel's behavior varies according to its state. For the case of binary-input binary-output channels, the channel noise when in state $s \in \mathcal{S}$ is given by a 2×2 stochastic matrix W_s s.t. $W(y \mid x)$ is the probability that y is output on input x. The goal is to achieve reliable comcommunication when the AVC state changes arbitrarily (hence the name) for each codeword bit — for an arbitrary state sequence $(s_1, s_2, \ldots, s_n) \in \mathcal{S}^n$, the i'th bit is affected by W_{s_i}. In a *state-constrained* AVC, the state sequence $(s_1, \ldots, s_n)$ is limited to a subset of possible vectors in $\mathcal{S}^n$. A simple AVC is the "additive channel" where $\mathcal{S} = \{0, 1\}$ and, when in state s, the channel adds s mod 2 to the input bit. If we constrain the state sequence $(s_1, s_2, \ldots, s_n)$ so that $\sum_{i=1}^{n} s_i \leqslant pn$, we can model the setting where an *arbitrary* error vector e with at most p fraction 1's is added (xor'd) to the codeword by the channel; the important restriction here is that the error pattern is chosen independently of the codeword.

Oblivious Additive Errors. Csiszár and Narayan determine the capacity of an AVC with state constraints for the "average error criterion" [4]. A special case of their result is the following surprising claim for the binary additive AVC channel: There *exist* binary codes $E : \{0, 1\}^k \to \{0, 1\}^n$ of rate approaching $1 - H(p)$ with a deterministic decoding rule D such that for *every* error vector e of Hamming weight at most pn, for *most* (all but a $\exp(-\Omega(k))$ fraction) of messages m, D correctly recovers m from $E(m) + e$. Note that codes providing this guarantee *cannot* be linear, since the bad error vectors for all codewords are the same in a linear code. The decoding rule used in [4] to prove this claim was quite complex, and it was simplified to the more natural closest codeword rule in [5]. Langberg [18] revisited this special case and gave another proof of the above claim. He also coined the term *oblivious channel* for this noise model, to capture the fact that the error vector is chosen without knowledge of the message. We will use the term ***oblivious additive errors*** to refer to this model. The proofs of the existence of the above capacity-achieving codes for the average error criterion [4, 5, 18] are quite involved; one of the contributions of this paper is an alternate, simpler proof using list decoding.

Worst-case Additive Errors. The proof of Csiszár and Narayan [4] mentioned above in fact implies a stronger claim (see the remark at the end of [4]): For *every* message and *every* error pattern of at most a fraction p of errors (in particular the error can depend arbitrarily on the message), there is a *"stochastic"* code of rate approaching $1 - H(p)$ that encodes the message m combined with a small number extra random bits r, such that, with high probability over r, the message m as well as r can be correctly recovered by a decoder *without a priori knowledge of the encoder's random bits* r. This is stronger than the average error criterion since we can define a deterministic code whose message consists of both m and r, and its average error probability will be at most the maximum error probability of the stochastic code.

The ability to flip random coins at the encoder is a modest assumption that is easily met in most settings. Also, the above stochastic codes allow the channel to pick the error vector e with full knowledge of the message m to be transmitted, and correct communication still occurs w.h.p. (The error vector is "codeword oblivious" but *not* "message oblivious.") We call this channel model WEC_p (***worst-case additive errors***): an adversarially picked error vector e is added to the encoding of m regardless of which of its many possible encodings (corresponding to different random choices at the encoder) is transmitted. Note that the WEC_p

model captures both random error models such as BSC_p as well as some channels with memory, such as bursty channels.

Unfortunately, the techniques of Csiszár and Narayan [4] are non-constructive, based on (non-trivial) random coding arguments. The challenge therefore is to construct *explicit* stochastic codes achieving capacity on the WEC_p channel. Our main result is exactly such a construction; we also give a new, simpler existential results.

2 Our Results

Existential result via list decoding. We give a novel construction of stochastic codes for the WEC_p channel by combining *linear* list-decodable codes with a certain kind of authentication code called algebraic manipulation detection (AMD) codes (Section 5). Such AMD codes can detect *additive corruption* with high probability, and were defined and constructed for a cryptographic motivation in [3]. The decoder does not have access to the randomness r to "sign" the message m. The linearity of the list-decodable code is therefore crucial to make the combination with AMD codes work. The linearity ensures that the spurious messages output by the list-decoder are all additive offsets of the true message that depend only on the error vector (and not on m, r).

By using the existence of binary linear codes that achieve list decoding capacity, one can conclude the existence of stochastic codes of rate approaching $1 - H(p)$ for communication on WEC_p. This is a simpler proof than the earlier ones [4, 5]. An additional positive feature of our construction is that even when the fraction of errors exceeds p, the decoder outputs a decoding failure with high probability (rather than decoding incorrectly)! This feature is important when using these codes as a component in our explicit construction mentioned next.

An explicit construction achieving capacity. For list decoding, explicit binary codes of optimal rate are not known, so one cannot use the above connection to construct explicit stochastic codes of rate $\approx 1 - H(p)$ on the WEC_p channel. Nevertheless, using several tools and constructions from coding theory and pseudorandomness (see Section 3 for an overview), we give an explicit construction of capacity-achieving stochastic codes against worst-case additive errors.[2]

Theorem 1. *For every $p \in (0, 1/2)$, every $\varepsilon > 0$, and infinitely many n, there exists an explicit, efficiently encodable and decodable stochastic code of block length n and rate $R \geqslant 1 - H(p) - \varepsilon$ which corrects a p fraction of worst-case additive errors with probability $1 - o(1)$, that is, for every message m and error vector e of Hamming weight at most pn, we have*

$$\Pr_r(\textsc{Decode}(\textsc{Encode}(m, r) + e) = m) \geqslant 1 - \exp(-\Omega_{p,\varepsilon}(n/\log^2 n)) .$$

Deterministic codes against oblivious errors. We also get a similar explicit construction of capacity-achieving deterministic codes for average error criterion against oblivious additive errors.

Theorem 2. *For every $p \in (0, 1/2)$, and every $\varepsilon > 0$, and infinitely many n, there is an explicit binary code of rate R at least $1 - H(p) - \varepsilon$ together with deterministic polynomial time algorithms* $\textsc{Enc}$ *and* $\textsc{Dec}$ *such that for every error vector e of Hamming weight at most pn, we have*

$$\Pr_m(\textsc{Dec}(\textsc{Enc}(m) + e) = m) \geqslant 1 - \exp(-\Omega_{p,\varepsilon}(n/\log^2 n)) ,$$

where the probability is over a uniform choice of message $m \in \{0, 1\}^{Rn}$.

[2]Although we focus on the binary alphabet case in this paper, we believe, though have not verified, that our techniques will extend to any fixed finite field.

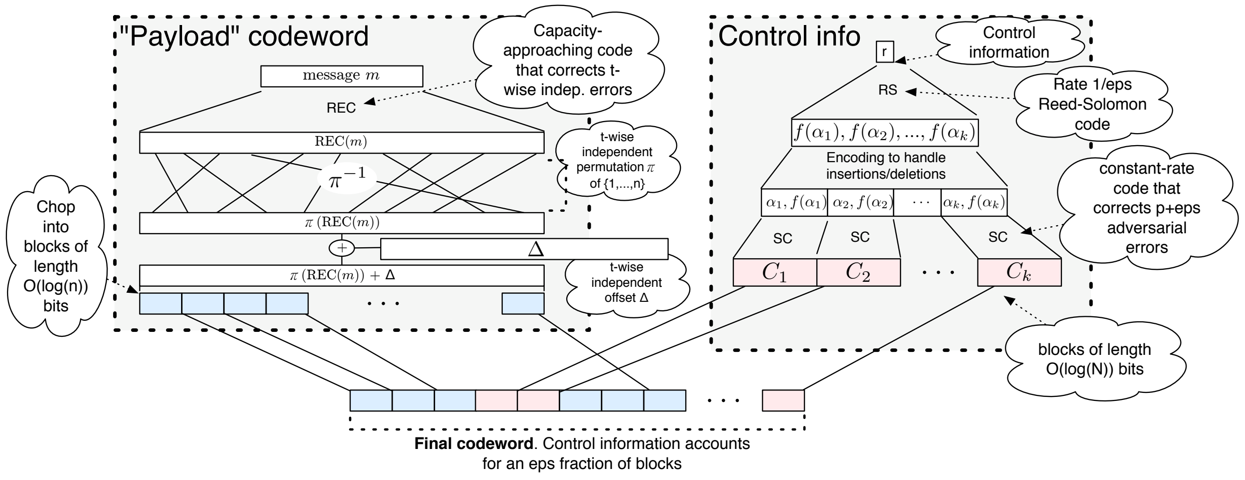
"Payload" codeword
Capacity-approaching code that corrects t-wise indep. errors
message m
REC
REC(m)
π^{-1}
t-wise independent permutation π of {1,...,n}
π (REC(m))
Δ
π (REC(m)) + Δ
t-wise independent offset Δ
Chop into blocks of length O(log(n)) bits
Control info
Control information
r
RS
Rate 1/eps Reed-Solomon code
$f(\alpha_1), f(\alpha_2), ..., f(\alpha_k)$
Encoding to handle insertions/deletions
$\alpha_1, f(\alpha_1)$
$\alpha_2, f(\alpha_2)$
$\alpha_k, f(\alpha_k)$
constant-rate code that corrects p+eps adversarial errors
SC
C_1
C_2
C_k
blocks of length O(log(N)) bits
Final codeword. Control information accounts for an eps fraction of blocks

Figure 1: Schematic description of encoder from Algorithm 1.

3 Construction overview

Our result is obtained by combining several ingredients in pseudorandomness and coding theory. At a high level the idea (introduced by Lipton [20] in a different context) is that if we permute the symbols of the codewords randomly (*after* the error pattern is fixed), then the adversarial error pattern looks random to the decoder. Therefore, an explicit code C_{BSC} that can achieve capacity for the binary symmetric channel (such as Forney's concatenated code construction [9]) can be used to communicate on WEC_p (in fact even on ADV_p) after the codewords symbols are randomly permuted. This allows one to achieve capacity against adversarial errors when the encoder and decoder share randomness that is unknown to the adversary causing the errors. But, crucially, this requires the decoder to know the random permutation that was used at the encoding.

Our encoder communicates the random permutation (in encoded form) also as part of the overall codeword, without relying on any shared randomness, public key, or other "extra" information. The decoder must be able to figure out the permutation correctly solely based on a noisy version of the overall codeword (that encodes the permutation plus the actual data). The seed used to pick this random permutation (plus some extra random seeds needed for the construction) is encoded by a low rate code that can correct several errors (say a Reed-Solomon code) and this information is dispersed into randomly located blocks of the overall codeword (see Figure 1). The random locations to place the control blocks are picked by a "sampler" — the seed for this sampler is also part of the *control information* along with the seed for the random permutation.

The key challenge is to ensure that the decoder can figure out which blocks encode the control information, and which blocks consist of "data" bits from the codeword of C_{BSC} (the "payload" codeword) that encodes the actual message. The control blocks (which comprise a tiny portion of the overall codeword) are further encoded by a stochastic code (call it the *control code*) that can correct somewhat more than a fraction p, say a fraction $p + \varepsilon$, of errors. These codes can have any constant rate — since they encode a small portion of the message their rate is not so important, so we can use explicit sub-optimal codes for this purpose.

Together with the random placement of the encoded control blocks, the control code ensures that a reasonable ($\Omega(\varepsilon)$) fraction of the control blocks (whose encodings by the control code incur fewer than $p + \varepsilon$ errors) will be correctly decoded. Moreover any blocks with too many errors will be flagged as an erasure with high probability. The fraction of correctly recovered control blocks will be large enough that all the control information can be recovered by decoding the Reed-Solomon code used to encode the control information into these blocks. This recovers the permutation used to scramble the symbols of the concatenated codeword. The decoder can then unscramble the symbols in the message blocks and run the standard algorithm for the concatenated code to recover the message.

One pitfall in the above approach is that message blocks could potentially get mistaken for corrupted control blocks and get decoded as erroneous control information that leads the whole algorithm astray. To prevent this, in addition to scrambling the symbols of the message blocks by a (pseudo)random permutation, we also add a pseudorandom offset (which is nearly t-wise independent for some t much larger than the length of the blocks). This will ensure that with high probability each message block will be very far from every codeword and therefore will not be mistaken for a control block.

One important issue we have glossed over is that a completely random permutation of the n bits of the payload codeword will take $\Omega(n \log n)$ bits to specify. This would make the control information be too big compared to the message length (whereas we need it to be a tiny fraction of the message length). Therefore, we use almost t-wise independent permutations for $t \approx \varepsilon n / \log n$. Such permutations can be sampled with $\approx \varepsilon n$ random bits. We then make use of the fact that C_{BSC} enables reliable decoding even when the error locations have such limited independence instead of being a uniformly random subset of all possible locations [24].

4 Some Coding Terminology

We define the relevant coding-theoretic notions.

Definition 1 (List decodable codes)**.** For a real p, $0 < p < 1$, and an integer $L \geqslant 1$, a code $C \subseteq \Sigma^n$ is said to be (p, L)-list decodable if for every $y \in \Sigma^n$ there are at most L codewords of C within Hamming distance pn from y. If for every y the list of $\leqslant L$ codewords within Hamming distance pn from y can be found in time polynomial in n, then we say C is *efficiently* (p, L)-list decodable. Note that $(p, 1)$-list decodability is equivalent to the distance of C being greater than $2pn$. □

An efficiently (p, L)-list decodable code can be used for communication on the ADV_p channel with the guarantee that the decoder can always find a list of at most L messages that includes the correct message.

Definition 2 (Codes for average error)**.** A code C with encoding function $\mathcal{E} : \mathcal{M} \to \Sigma^n$ is said to be (efficiently) *p-decodable with average error δ* if there is a (polynomial time computable) decoding function $D : \Sigma^n \to \mathcal{M} \cup \{\perp\}$ such that for *every* error vector $e \in \Sigma^n$, the following holds for at least a fraction $(1 - \delta)$ of messages $m \in \mathcal{M}$: $D(\mathcal{E}(m) + e) = m$. □

The following is the main property that we are interested in achieving in this work. A generalization of this notion (see Definition 4 and Observation 3 below) is in fact stronger than the notion of codes for average error. Also, though we do not require it in the definition, our constructions of stochastic codes will also have the desirable property that when the number of errors exceeds pn, with high probability the decoder will output $\perp$ (a decoding failure) rather than decoding incorrectly.

Definition 3 (Stochastic codes and their decodability)**.** A stochastic binary code of rate R and block length n is given by an encoding function $\mathsf{Enc} : \{0,1\}^{Rn} \times \{0,1\}^b \to \{0,1\}^n$ which encodes the Rn message bits together with some additional random bits into an n-bit codeword.

Such a code is said to be (efficiently) *p-decodable with probability* $1 - \delta$ if there is a (deterministic polynomial time computable) decoding function $\mathsf{Dec} : \{0,1\}^n \to \{0,1\}^{Rn} \cup \{\perp\}$ such that for every

$m \in \{0,1\}^{Rn}$ and every error vector $e \in \{0,1\}^n$ of Hamming weight at most pn, with probability at least $1-\delta$ over the choice of a random string $\omega \in \{0,1\}^b$, we have $\mathsf{Dec}\big(\mathsf{Enc}(m,\omega)+e\big) = m$.

In this case, we also say that the stochastic code C allows reliable communication over the worst-case channel WEC_p with probability at least $1-\delta$. □

Definition 4 (Strongly decodable stochastic codes)**.** For a code as in Definition 3, if the decoding function correctly computes in addition to the message m also the randomness ω used at the encoder with probability at least $1-\delta$, the we say that the stochastic code is strongly p-decodable with probability $1-\delta$. □

Using a *strongly decodable* stochastic code we can get a code for average error by simply using the last few bits of the message as the randomness of the stochastic encoder. If the number of random bits used by the stochastic code is small compared to the message length, the rates of the codes in the two models are almost the same.

Observation 3. *A stochastic code* SSC *that is strongly p-decodable with probability $1-\delta$ gives a code* AVC *of the same block length that is p-decodable with average error δ. If the ratio of number of random bits to message bits in* SSC *is λ, the rate of* AVC *is $(1+\lambda)$ times the rate of* SSC.

5 List decoding implies codes for worst-case additive errors

In this section, we will demonstrate how to use good *linear* list-decodable codes to get good stochastic codes. The conversion uses the list-decodable code as a black-box and loses only a negligible amount in rate. In particular, by using binary *linear* codes that achieve list decoding capacity, we get stochastic codes which achieve the capacity of WEC_p (note that the linearity of the code is crucial for our analysis). The other ingredient we need for the construction is an authentication code ("MAC") that can detect *additive corruption* with high probability, which has been studied under the label of Algebraic manipulation detection (AMD) codes [3].

5.1 Algebraic manipulation detection (AMD) codes

The following is not the most general definition of AMD codes from [3], but will suffice for our purposes and is the one we will use.

Definition 5. Let $\mathcal{G} = (G_1, G_2, G_3)$ be a triple of abelian groups (whose group operations are written additively) and $\delta > 0$ be a real. Let $G = G_1 \times G_2 \times G_3$ be the product group (with componentwise addition). An $(\mathcal{G},\delta)$-algebraic manipulation code, or $(\mathcal{G},\delta)$-AMD code for short, is given by a map $f : G_1 \times G_2 \to G_3$ with the following property:

$$\text{For every } x \in G_1 \text{, and all } \Delta \in G, \quad \Pr_{r\in G_2}\big[D((x,r,f(x,r))+\Delta) \notin \{x,\perp\}\big] \leqslant \delta \ ,$$

where the decoding function $D : G \to G_1 \cup \{\perp\}$ is given by $D((x,r,s)) = x$ if $f(x,r) = s$ and $\perp$ otherwise. The *tag size* of the AMD code is defined as $\log|G_2| + \log|G_3|$ — it is the number of bits the AMD encoding appends to the source. □

Intuitively, the AMD allows one to authenticate x via a signed form $(x,r,f(x,r))$ so that an adversary who manipulates the signed value by adding an offset Δ cannot cause incorrect decoding of some $x' \neq x$. The following concrete scheme from [3] achieves near optimal tag size and we will make use of it.

Theorem 4. *Let $\mathbb{F}$ be a finite field of size q and characteristic p, and d be a positive integer such that $d+2$ is not divisible by p. Then the function $f^{(d)}_{\text{AMD}} : \mathbb{F}^d \times \mathbb{F} \to \mathbb{F}$ given by $f^{(d)}_{\text{AMD}}(x,r) = r^{d+2} + \sum_{i=1}^{d} x_i r^i$ is a $\big(\mathcal{G}, \frac{d+1}{q}\big)$-AMD code with tag size $2\log q$ where $\mathcal{G} = (\mathbb{F}^d, \mathbb{F}, \mathbb{F})$.*[3]

[3]Here we mean the additive group of the vector space $\mathbb{F}^d$.

5.2 Combining list decodable and AMD codes

Using a (p, L)-list decodable code C of length n, for any error pattern e of weight at most pn, we can recover a list of L messages that includes the correct message m. We would like to use the stochastic portion of the encoding to allow us to unambiguously pick out m from this short list. The key insight is that if C is a *linear* code, then the other (less than L) messages in the list are all fixed offsets of m that depend *only* on the error pattern e. So if prior to encoding by the list-decodable code C, the messages are themselves encodings as per a good AMD code, and the tag portion of the AMD code is good for these fixed L or fewer offsets, then we can uniquely detect m from the list using the AMD code. If the tag size of the AMD code is negligible compared to the message length, then the overall rate is essentially the same as that of the list-decodable code. Since there exist binary linear (p, L)-list-decodable codes of rate approaching $1 - H(p)$ for large L, this gives stochastic codes (in fact, *strongly decodable* stochastic codes) of rate approaching $1 - H(p)$ for the WEC_p channel.

Theorem 5 (Stochastic codes from list decoding and AMD)**.** *Let b, d be positive integers with d odd and $k = b(d + 2)$. Let $C : \mathbb{F}_2^k \to \mathbb{F}_2^n$ be the encoding function of a binary linear (p, L)-list decodable code. Let $f^{(d)}_{\text{AMD}}$ be the function from Theorem 4 for the choice $\mathbb{F} = \mathbb{F}_{2^b}$. Let C' be the stochastic binary code with encoding map $E : \{0, 1\}^{bd} \times \{0, 1\}^b \to \{0, 1\}^n$ given by*

$$E(m, r) = C\big(m, r, f^{(d)}_{\text{AMD}}(m, r)\big) .$$

Then if $\frac{d+1}{2^b} \leqslant \frac{\delta}{L}$, the stochastic code C' is strongly p-decodable with probability $1 - \delta$. If C is efficiently (p, L)-list decodable, then C' is efficiently (and strongly) p-decodable with probability $1 - \delta$.

Moreover, even when e has weight greater than pn, the decoder detects this and outputs $\perp$ (a decoding failure) with probability at least $1 - \delta$.

Note that the rate of C' is $\frac{d}{d+2}$ times the rate of C.

Proof. Fix an error vector $e \in \{0, 1\}^n$ and a message $m \in \{0, 1\}^{bd}$. Suppose we pick a random r and transmit $E(m, r)$, so that $y = E(m, r) + e$ was received.

The decoding function D, on input y, first runs the list decoding algorithm for C to find a list of $\ell \leqslant L$ messages $m'_1, \ldots, m'_\ell$ whose encodings are within distance pn of y. It then decomposes m'_i as (m_i, r_i, s_i) in the obvious way. The decoder then checks if there is a unique index $i \in \{1, 2, \ldots, \ell\}$ for which $f^{(d)}_{\text{AMD}}(m_i, r_i) = s_i$. If so, it outputs (m_i, r_i), otherwise it outputs $\perp$.

Let us now analyze the above decoder D. First consider the case when $\text{wt}(e) \leqslant pn$. In this case we want to argue that the decoder correctly outputs (m, r) with probability at least $1 - \delta$ (over the choice of r). Note that in this case one of the m'_i's equals $(m, r, f^{(d)}_{\text{AMD}}(m, r))$, say this happens for $i = 1$ w.l.o.g. Therefore, the condition $f^{(d)}_{\text{AMD}}(m_1, r_1) = s_1$ will be met and we only need to worry about this happening for some $i > 1$ also.

Let $e_i = y - C(m'_i)$ be the associated error vectors for the messages m'_i. Note that $e_1 = e$. By linearity of C, the e_i's only depend on e; indeed if $c'_1, \ldots, c'_\ell$ are all the codewords of C within distance pn from e, then $e_i = c'_i + e$. Let Δ_i be the preimage of c'_i, i.e., $c'_i = C(\Delta_i)$. Therefore we have $m'_i = m'_1 + \Delta_i$ where the Δ_i's only depend on e. By the AMD property, for each $i > 1$, the probability that $f^{(d)}_{\text{AMD}}(m_i, r_i) = s_i$ over the choice of r is at most $\frac{d+1}{2^b} \leqslant \delta/L$. Thus with probability at least $1 - \delta$, none of the checks $f^{(d)}_{\text{AMD}}(m_i, r_i) = s_i$ for $i > 1$ succeed, and the decoder thus correctly outputs $m_1 = m$.

In the case when $\text{wt}(e) > pn$, the same argument shows that the check $f^{(d)}_{\text{AMD}}(m_i, r_i) = s_i$ passes with probability at most δ/L for each i (including $i = 1$). So with probability at least $1 - \delta$ none of the checks pass, and the decoder outputs $\perp$. □

Plugging into the above theorem the existence of binary linear $(p, O(1/\varepsilon))$-list-decodable codes of rate $1 - H(p) - \varepsilon/2$, and picking $d = 2\lceil c_0/\varepsilon \rceil + 1$ for some absolute constant c_0, we can conclude the following result on existence of stochastic codes achieving capacity for reliable communication on the WEC_p channel.

Corollary 6. *For every p, $0 < p < 1/2$ and every $\varepsilon > 0$, there* exists *a family of stochastic codes of rate at least $1 - H(p) - \varepsilon$ and a deterministic (exponential time) decoder which enables reliable communication over WEC_p with probability of miscommunication at most $2^{-\Omega_{\varepsilon,p}(n)}$, where n is the block length. Moreover, when more than a fraction p of errors occur, the decoder is able to detect this and report a decoding failure with probability at least $1 - 2^{-\Omega_{\varepsilon,p}(n)}$.*
If we have explicit binary codes of rate R that can be efficiently list-decoded from fraction p of errors with list-size at most some constant $L = L(p)$, we can construct explicit stochastic codes with the above guarantee with rate R along with an efficient *decoder.*

Remark 1. Since the above stochastic codes are *strongly* p-decodable, by Observation 3 they also imply capacity achieving codes for the average error criterion: for every error vector, all but an exponentially small fraction of messages are communicated correctly.

6 Explicit Codes for Worst-case Additive Errors

6.1 Ingredients

Our construction uses a number of tools from coding theory and pseudorandomness. These are described in detail in Appendix A. Briefly, we use:

- A constant-rate explicit stochastic code $\mathsf{SC} : \{0,1\}^b \times \{0,1\}^b \to \{0,1\}^{c_0 b}$, defined on blocks of length $c_0 b = \Theta(\log N)$, that is efficiently strongly $p + O(\varepsilon)$ decodable with probability $1 - c_1/N$. This follows from Theorem 5 above.

- A rate $O(\varepsilon)$ Reed-Solomon code RS which encodes a message as the evaluation of a polynomial at points $\alpha_1, ..., \alpha_\ell$ in such a way that an efficient algorithm RS-DECODE can efficiently recover the message given at most $\varepsilon\ell/4$ correct symbols and at most $\varepsilon/24$ incorrect ones.

- A randomness-efficient *sampler* $\mathsf{Samp} : \{0,1\}^\sigma \to [N]^\ell$, such that for any subset $B \subseteq [N]$ of size at least μN, the output set of the sampler intersects with B in roughly a μ fraction of its size, that is $|\mathsf{Samp}(s) \cap B| \approx \mu|\mathsf{Samp}(s)|$, with high probability over $s \in \{0,1\}^\sigma$. We use an expander-based construction from Vadhan [25].

- A generator $\mathsf{KNR} : \{0,1\}^\sigma \to S_n$ for an (almost) t-wise independent family of permutations of the set $\{1, ..., n\}$, that uses a seed of $\sigma = O(t \log n)$ random bits (Kaplan, Naor, and Reingold [16]).

- A generator $\mathsf{POLY}_t : \{0,1\}^\sigma \to \{0,1\}^n$ for a t-wise independent distribution of bit strings of length n, that uses a seed of $\sigma = O(t \log n)$ random bits.

- An explicit efficiently decodable, rate $R = 1 - H(p) - O(\varepsilon)$ code $\mathsf{REC} : \{0,1\}^{Rn} \to \{0,1\}^n$ that can correct a p fraction of t-wise independent errors, that is: for every message $m \in \{0,1\}^{Rn}$, and every error vector $e \in \{0,1\}^n$ of Hamming weight at most pn, we have $\mathsf{Dec}(\mathsf{REC}(m) + \pi(e)) = m$ with probability at least $1 - 2^{-\Omega(\varepsilon^2 t)}$ over the choice of a permutation $\pi \in_R$ range(KNR). (Here $\pi(e)$ denotes the permuted vector: $\pi(e)_i = e_{\pi(i)}$.) A standard family of concatenated codes satisfies this property (Smith [24]).

Algorithm 1. ENCODE: On input parameters N, p, ε (with $p + \varepsilon < 1/2$), and message $m \in \{0,1\}^{R \cdot N}$, where $R = 1 - H(p) - O(\varepsilon)$.

```
1: $\Lambda \leftarrow 2c_0$    ▷ Here $c_0 = c_0(p+\varepsilon)$ is the constant in the stochastic code from Proposition 14 that can correct a fraction $p+\varepsilon$ of errors.
2: $n \leftarrow \frac{N}{\Lambda \log N}$    ▷ The final codeword consists of $n$ *blocks* of length $\Lambda \log N$.
3: $\ell \leftarrow 24\varepsilon N / \log N$    ▷ The control codeword is $\ell$ blocks long.
4: $n' \leftarrow n - \ell$ and $N' \leftarrow n' \cdot (\Lambda \log N)$    ▷ The payload codeword is $n'$ blocks long (i.e. $N'$ bits).
```

Phase 1: Generate control information

```
5: Select $s_\pi \leftarrow_R \{0,1\}^{\varepsilon^2 N}$.    ▷ $s_\pi$ is a seed for picking a permutation of $[N']$ from an almost $t$-wise independent family as per Proposition 17, where $t = \Omega(\varepsilon^2 N / \log N)$.
6: Select $s_\Delta \leftarrow_R \{0,1\}^{\varepsilon^2 N}$.    ▷ $s_\Delta$ is a seed for picking a $t'$-wise independent string $\Delta$, where $t' = \Omega(\varepsilon^2 N / \log N)$ as per Proposition 18.
7: Select $s_T \leftarrow_R \{0,1\}^{\varepsilon^2 N}$.    ▷ $s_T$ is a seed for sampling a pseudorandom subset $T \subset [n] = [n' + \ell]$ of size $\ell$ as per Proposition 16.
8: $\omega \leftarrow (s_\pi, s_\Delta, s_T)$    ▷ Total length $|\omega| = 3\varepsilon^2 N$.
```

Phase 2: Encode control information

```
9: Let $\mathbb{F} = \mathbb{F}_N$ and $S = (\alpha_1, \dots, \alpha_\ell) \subseteq \mathbb{F}$ be an arbitrary subset of size $\ell$.
   Compute $(a_1, ..., a_\ell) \leftarrow \mathsf{RS}_{\mathbb{F}, S, \ell, |\omega|/\log N}(\omega)$.
       ▷ RS (defined in (1)) is a rate $\varepsilon/8$ Reed-Solomon code of length $24\varepsilon N = \frac{8}{\varepsilon} \cdot |\omega|$ bits, i.e., $\ell = 24\varepsilon N / \log N$ field symbols.
10: for $i \leftarrow 1$ to $\ell$ do
11:     $A_i \leftarrow (\alpha_i, a_i)$
            ▷ Add location information to each RS symbol to get block $A_i$ of $2\log N$ bits.
12:     Set $C_i \leftarrow \mathsf{SC}(A_i, r_i)$, where $r_i \leftarrow_R \{0,1\}^{2\log N}$.
            ▷ Here $\mathsf{SC} = \mathsf{SC}_{2\log N, p+\varepsilon} : \{0,1\}^{2\log N} \times \{0,1\}^{2\log N} \to \{0,1\}^{\Lambda \log N}$ is a stochastic code that allows reliable communication over $\mathsf{WEC}_{p+\varepsilon}$ with probability $1 - c_1/N^2 > 1 - 1/N$ as per Proposition 14.
            ▷ The control information $\omega$ is thus encoded by a concatenated code with an outer Reed-Solomon code and inner code SC.
13: end for
```

Phase 3: Generate the payload codeword

```
14: $P \leftarrow \mathsf{REC}(m)$,    ▷ REC : $\{0,1\}^{R'N'} \to \{0,1\}^{N'}$ is a code that can correct a $p + 25\Lambda\varepsilon$ fraction of *t-wise independent* errors, as per Proposition 19. Here $R' = \frac{RN}{N'}$.
15: $\pi \leftarrow \mathsf{KNR}(s_\pi)$    ▷ Generate permutation $\pi : [n] \to [n]$ using Proposition 17.
16: $\Delta \leftarrow \mathsf{POLY}(s_\Delta)$    ▷ Generate random offset string $\Delta \in \{0,1\}^n$ as guaranteed by Proposition 18.
17: $\pi^{-1}(P) \leftarrow$ (bits of $P$ permuted according to $\pi^{-1}$)
18: $Q \leftarrow \pi^{-1}(P) \oplus \Delta$
19: Cut $Q$ into $n'$ blocks $B_1, ...B_{n'}$ of length $\Lambda \log N$ bits.    ▷ Recall that $n' = \frac{n}{\Lambda \log N}$.
```

Phase 4: Interleave blocks of payload codeword and control codeword

```
20: $T \leftarrow \mathsf{Samp}(s_T)$    ▷ Generate pseudorandom size-$\ell$ subset of $[n' + \ell]$ as locations for control blocks, using sampler of Proposition 16.
21: Interleave blocks $C_1, ..., C_\ell$ with blocks $B_1, ..., B_{n'}$, using $C_i$ blocks in positions from $T$ and $B_i$ blocks in positions from $\overline{T} = [n' + \ell] \setminus T$.
```

Algorithm 2. DECODE: On input a received word x of the length output by ENCODE.

```
▷ The decoder's pseudocode is annotated with statements about performance. These claims assume that $x = \text{ENCODE}(m;\omega, r_1, \ldots, r_\ell) + e$ where $e$ contains at most a fraction $p$ of ones and the random string $(\omega; r_1, r_2, \ldots, r_\ell)$ is uniform and independent of the pair $(m, e)$.
1: Cut $x$ into $n' + \ell$ blocks $x_1, ..., x_{n'+\ell}$ of length $\Lambda \log(n)$ each.
2: for $i \leftarrow 1$ to $n' + \ell$ do
3:     $\tilde{F}_i \leftarrow$ SC-DECODE$(x_i)$.
           ▷ Run the decoder for the stochastic code SC used to encode the symbols of the RS codeword encoding the control blocks.
           ▷ With high prob, non-control blocks are rejected (Lemma 10), and control blocks are either correctly decoded or discarded (Lemma 9).
4:     if $\tilde{F}_i \neq \perp$ then
5:         Parse $\tilde{F}_i$ as $(\tilde{\alpha}_i, \tilde{a}_i)$, where $\tilde{\alpha}_i, \tilde{a}_i \in \mathbb{F}_N$.
6:     end if
7: end for
8: $(\tilde{s}_T, \tilde{s}_\Delta, \tilde{s}_\pi) \leftarrow$ RS-DECODE$\Big($pairs $(\tilde{\alpha}_i, \tilde{a}_i)$ output above$\Big)$.
           ▷ With high prob., enough control blocks are decoded correctly to recover the control information (Lemma 11).
9: $\tilde{T} \leftarrow \mathsf{Samp}(\tilde{s}_T)$,
   $\tilde{\Delta} \leftarrow \mathsf{POLY}(\tilde{s}_\Delta)$
   $\tilde{\pi} \leftarrow \mathsf{KNR}(\tilde{s}_\pi)$
10: $\tilde{Q} \leftarrow$ concatenation of blocks $x_i$ in $[n' + k] \setminus \tilde{T}$
           ▷ Fraction of errors in $\tilde{Q}$ is at most $p + O(\varepsilon)$.
11: $\tilde{P} \leftarrow \pi(\tilde{Q} \oplus \tilde{\Delta})$
           ▷ If control info is correct, then errors in $\tilde{P}$ are almost $t$-wise independent.
12: $\tilde{m} \leftarrow$ REC-DECODE$(\tilde{P})$
           ▷ Run the decoder from Proposition 19.
           ▷ With high prob., $\tilde{m} = m$
```

6.2 Main Theorem

We are now ready to prove our main result (Theorem 1), which we restate below.

Theorem 7. *For every $p \in (0, 1/2)$, and every $\varepsilon > 0$, the functions* ENCODE, DECODE *(Algorithms 1 and 2) form an explicit, efficiently encodable and decodable stochastic code with rate $R = 1 - H(p) - \varepsilon$ which allows reliable communication over* WEC_p *with probability $1 - \exp(-\Omega(\varepsilon^2 N/\log^2 N)))$, where N is the block length of the code.*

With all the ingredients described in Appendix A in place, we can describe and analyze the code of Theorem 7. The encoding algorithm is given in Algorithm 1 (page 9). The corresponding decoder is given in Algorithm 2 (page 10). Also, a schematic illustration of the encoding is in Figure 1. The reader might find it useful to keep in mind the high level description from Section 3 when reading the formal description.

Proof of Theorem 7: The rate R of the overall code is almost equal to the rate R' of the code REC used to encode the actual message bits m, since the encoded control information has length $O(\varepsilon N)$ which is much smaller than the number of message bits (by picking ε small enough). The code REC needs to correct a fraction $p + 25\Lambda\varepsilon$ of t-wise independent errors, so we can pick $R' \geqslant 1 - H(p) - O(\varepsilon)$. Now the rate $R = \frac{R'N'}{N} = R'(1 - 24\Lambda\varepsilon) \geqslant 1 - H(p) - O(\varepsilon)$ (for small enough $\varepsilon > 0$).

We now turn to the analysis of the decoder. Fix a message $m \in \{0,1\}^{R \cdot N}$ and an error vector $e \in \{0,1\}^N$ with Hamming weight at most pN. Suppose that we run ENCODE on m and coins ω chosen *independently* of the pair m, e, and let $x = \text{ENCODE}(m;\omega) + e$. The decoder parses x into blocks $x_1, ..., x_{n'+\ell}$ of length $\Lambda \log N$, corresponding to the blocks output by the encoder.

We first prove that the control information is correctly recovered with high probability. This is content of the four lemmas below which are proved in the next section (Section 6.3). Conditioning on correct recovery of the control information, we then use the analysis from [24] to show that the payload message is correctly recovered.

Definition 6. A sampled set T is *good* for error vector e if the fraction of control blocks with relative error rate at most $p + \varepsilon$ is at least $\frac{\varepsilon}{2}$. $\square$

Lemma 8. *For any error vector e of relative weight at most p, with probability at least* $1 - \exp(-\Omega(\varepsilon^3 N/\log N)$ *over the choice of sampler seed s_T, the set T is good for e.*

Lemma 9. *For any e, T such that T is good for e, with probability at least* $1 - \exp(-\Omega(\varepsilon^3 N/\log N))$ *over the random coins $(r_1, r_2, \dots, r_\ell)$ used by the ℓ* SC *encodings, we have:*

1. *The number of control blocks correctly decoded by* SC-DECODE *is at least $\frac{\varepsilon\ell}{4}$.*
2. *The number of* erroneously *decoded control blocks is less than $\frac{\varepsilon\ell}{24}$.*
   (By erroneously decoded, we mean that SC-DECODE *outputs neither $\perp$ nor the correct message.)*

Lemma 10. *For every m, e, s_T, s_π, with probability at least* $1 - \exp(-\Omega(\varepsilon^2 N/\log^2 N)))$ *over the offset seed s_Δ, the number of* payload *blocks incorrectly accepted as control blocks by* SC-DECODE *is less than* $\frac{\varepsilon\ell}{24}$.

Lemma 11. *For any m and e, with probability* $1 - \exp(-\Omega(\varepsilon^2 N/\log^2 N))$ *over the choice of the control information and the coins of* SC*, the control information is correctly recovered, that is $\tilde{r} = r$.*

We defer the proof these lemmas to the next section. First, we use them to prove that our stochastic code corrects worst-case additive errors at rates arbitrarily close to capacity (Theorem 7).

The lemmas show that the control information r is correctly recovered with high probability. Suppose for a moment that they are recovered correctly with probability *exactly one* — that is, assume that the correct control information is magically handed directly to the decoder (we will subsequently adjust the argument to deal with conditioning on this event).

Fix a message m, error vector e, and sampler seed s_T, and let e_Q be the restriction of e to the payload codeword, i.e., blocks not in T. The relative weight of e_Q is at most $\frac{pN}{N'} = p\frac{N'+\ell\Lambda\log N}{N'} = p(1+24\varepsilon\Lambda\frac{N}{N'}) \leqslant p(1 + 25\Lambda\varepsilon)$ (for sufficiently small ε).

Now since s_π is selected independently from T, and since the control information is assumed to be correct with probability 1, the permutation π is independent of the payload error e_Q. Consider the string $\tilde{P}$ that is input the the REC decoder. We can write $\tilde{P} = \tilde{\pi}(\tilde{Q} \oplus \tilde{\Delta}) = \pi(Q \oplus e_Q \oplus \Delta)$. Because a permutation of the bit positions is a linear permutation of $\mathbb{Z}_2^{N'}$, we get $\tilde{P} = \pi(Q + \Delta) \oplus \pi(e_Q) = P \oplus \pi(e_Q)$.

Thus the input to REC is corrupted by a fraction of at most $p(1+25\Lambda\varepsilon)$ errors which are t-wise independent, in the sense of Proposition 19 [24]. Thus, with probability at least $1 - e^{-\Omega(\varepsilon^2 t)} = 1 - e^{-\Omega(\varepsilon^4 N/\log N)}$, the message m is correctly recovered by DECODE.

In the actual construction, the control information is not handed directly to the decoder. Instead, we must condition our analysis on the control information being decoded correctly, which introduces dependencies between e and π. Nevertheless, conditioning on an event of probability q can increase the likelihood of any other event by a factor of at most $1/q$. Hence, the probability of incorrectly decoding the message m conditioned on the control information being correctly recovered is at most

$$\frac{\Pr(\text{REC decodes } m \text{ incorrectly given correct } r)}{\Pr(r \text{ is correctly recovered})} \leqslant 2\Pr(\text{REC decodes } m \text{ incorrectly given correct } r)\,.$$

Overall, the error probability is $\exp(-\Omega(\varepsilon^2 N/\log^2 N)))$ (the probability of incorrectly recovering r from Lemma 11), plus $\exp(-\Omega(\varepsilon^4 N/\log N))$ (the probability that REC erroneously decodes m). Because ε is a constant relative to $\log N$, it is the former probability that dominates. This completes the analysis of the decoder and the proof of Theorem 7.

Remark 2. It will be interesting to achieve an error probability of $2^{-\Omega_\varepsilon(N)}$, i.e., a positive "error exponent," in Theorem 7 instead of the $2^{-\Omega_\varepsilon(N/\log^2 N)}$ bound we get. A more careful analysis (perhaps one that works with *almost* t'-wise independent offset Δ) can probably improve our error probability to $2^{-\Omega_\varepsilon(N/\log N)}$, but going further using our approach seems difficult. The existential result due to Csiszár and Narayan [4] achieves a positive error exponent for all rates less than capacity, as does our existence proof using list decoding in Section 5.2.

6.3 Proofs of Control Information Lemmas

We now prove the lemmas that bound the probability of the control information being correctly decoded.

Proof of Lemma 8. Let $B \subset [n] = [n' + \ell]$ be the set of blocks that contain a $(p+\varepsilon)$ or smaller fraction of errors. We first prove that B must occupy at least an ε fraction of total number of blocks: to see why, let γ be the proportion of blocks which have error rate at most $(p+\varepsilon)$. The total fraction of errors in x is then at least $(1-\gamma)(p+\varepsilon)$. Since this fraction is at most p by assumption, we must have $1-\gamma \leqslant p/(p+\varepsilon)$. So $\gamma \geqslant \varepsilon/(p+\varepsilon) > \varepsilon$.

Next, we show that the number of *control blocks* that have error rate at most $p+\varepsilon$ cannot be too small. The error e is fixed before the encoding algorithm is run, and so the sampler seed s_T is chosen independently of the set B. Thus, the fraction of control blocks in B will be roughly ε. Specifically, we can apply Proposition 16 with $\mu = \varepsilon$ (since B occupies at least an ε fraction of the set of blocks), $\theta = \varepsilon/2$ and $\sigma = \varepsilon^2 N$. We get that the error probability γ is $\exp(-\Omega(\theta^2\ell)) = \exp(-\Omega(\varepsilon^3 N/\log N)$. (Note that for constant ε, the seed length $\sigma = \varepsilon^2 N \gg \log N + \ell\log(1/\varepsilon)$ is large enough for the proposition to apply.) □

Proof of Lemma 9. Fix e and the sampled set T which is good for e. Consider a particular received block x_i that corresponds to control block j, that is, $x_i = C_j + e_i$. The key observation is that the error vector e_i depends on e and the sampler seed T, but it is *independent* of the randomness used by SC to generate C_j. Given this observation, we can apply Proposition 14 directly:

(a) If block i has error rate at most $p+\varepsilon$, then SC-DECODE decodes correctly with probability at least $1 - c_1/N^2 \geqslant 1 - 1/N$ over the coins of SC.

(b) If block i has error rate more than $p+\varepsilon$, then SC-DECODE outputs $\perp$ with probability at least $1-c_1/N^2 \geqslant 1-1/N$ over the coins of SC.

Note that in both statements (a) and (b), the probability need only be taken over the coins of SC.

Consider $\mathbf{Y}$, the the number of control blocks that either (i) have "low" error rate ($\leqslant p+\varepsilon$) yet are not correctly decoded, or (ii) have high error rate, and are not decoded as $\perp$. Because statements (a) and (b) above depend only on the coins of SC, and these coins are chosen independently in each block, the variable $\mathbf{Y}$ is statistically dominated by a sum of independent Bernoulli variables with probability $1/N$ of being 1. Thus $E[\mathbf{Y}] \leqslant \ell/N < 1$. By a standard additive Chernoff bound, the probability that Y exceeds $\varepsilon\ell/24$ is at most $\exp(-\Omega(\varepsilon^2\ell))$. The bound on $\mathbf{Y}$ implies both the bounds in the lemma. □

Proof of Lemma 10. Consider a block x_i that corresponds to payload block j, that is, $x_i = B_j + e_i$. Fix e, s_T, and s_π. The offset Δ is independent of these, and so we may write $x_i = y_i + \Delta_i$, where y_i is fixed independently of Δ_i. Since Δ is a t'-wise independent string with $t' = \Omega(\varepsilon^2 N/\log N)$ much greater than the size $\Lambda \log N$ of each block, the string Δ_i is uniformly random in $\{0,1\}^{\Lambda \log N}$. Hence, so is x_i. By Proposition 14 we know that on input a random string, SC-DECODE outputs $\perp$ with probability at least $1-c_1/N^2 \geqslant 1-1/N$

Moreover, the t'-wise independence of the *bits* of Δ implies $\frac{t'}{\Lambda \log N}$-wise independence of the *blocks* of Δ. Define $t'_{blocks} = \min\{\frac{t'}{\Lambda \log N}, \frac{\varepsilon\ell}{96}\}$. Note that $\Omega\big(\frac{\varepsilon^2 N}{\log^2 N}\big) \leqslant t'_{blocks} \leqslant \frac{\varepsilon\ell}{96}$. The decisions made by SC-DECODE on payload blocks are t'_{blocks}-wise independent. Let $\mathbf{Z}$ denote the number of payload blocks that are incorrectly accepted as control blocks by SC-DECODE. We have $E[\mathbf{Z}] \leqslant \frac{n'}{N} \leqslant \varepsilon\ell/48$ (for large enough N).

We can apply a concentration bound of Bellare and Rompel [2, Lemma 2.3] using $t = t'_{blocks}$, $\mu = E[\mathbf{Z}] \leqslant \frac{\varepsilon\ell}{48}$, $A = \frac{\varepsilon\ell}{48}$, to obtain the bound

$$\Pr[\mathbf{Z} \geqslant \tfrac{\varepsilon\ell}{24}] \leqslant 8\left(\frac{t'_{blocks}\cdot\mu + (t'_{blocks})^2}{(\varepsilon\ell/48)^2}\right)^{t'_{blocks}/2} \leqslant (\log N)^{-\Omega(t'_{blocks})} \leqslant e^{-\Omega(\varepsilon^2 N \log\log N/\log^2 N)} .$$

This bound implies the lemma statement. □

Proof of Lemma 11. Suppose the events of Lemmas 9 and 10 occur, that is, for at least $\varepsilon\ell/4$ of the control blocks the recovered value $\tilde{F}_i$ is correct, at most $\varepsilon\ell/24$ of the control blocks are erroneously decoded, and at most $\varepsilon\ell/24$ of the payload blocks are mistaken for control blocks.

Because the blocks of the control information come with the (possibly incorrect) evaluation points $\tilde{\alpha}_i$, we are effectively given a codeword in the Reed-Solomon code defined for the related point set $\{\tilde{\alpha}_i\}$. Now, the degree of the polynomial used for the original RS encoding is $d^* = |\omega|/\log(N) - 1 < 3\varepsilon^2 N/\log N = \varepsilon\ell/8$. Of the pairs $(\tilde{\alpha}_i, \tilde{a}_i)$ decoded by SC-DECODE, we know at least $\frac{\varepsilon\ell}{4}$ are correct (these pairs will be distinct), and at most $2\cdot\frac{\varepsilon\ell}{24}$ are incorrect (some of these pairs may occur more than once, or even collide with one of the correct). If we eliminate any duplicate pairs and then run the decoding algorithm from Proposition 15, the control information ω will be correctly recovered as long as the number of correct symbols exceeds the number of wrong symbols by at least d^*+1. This requirement is met if $\frac{\varepsilon\ell}{4} - 2\times\frac{\varepsilon\ell}{24} \geqslant d^*+1$. This is indeed the case since $d^* < \varepsilon\ell/8$.

Taking a union bound over the events of Lemmas 9 and 10, we get that the probability that the control information is correctly decoded is at least $1-\exp(-\Omega(\varepsilon^2 N/\log^2 N))$, as desired. □

7 Explicit capacity-achieving codes for the average error criterion

As mentioned in the introduction, much of the research on arbitrarily varying channels has considered the average error of deterministic codes for uniformly distributed messages. By Observation 3, we can obtain deterministic codes with good average error by constructing *strongly* decodable stochastic codes (Definition 4). Recall that the decoder of a strongly decodable code recovers both the message and the random bits used by the encoder with high probability.

The construction of the previous section can be modified to be strongly decodable, proving Theorem 2, which we restate below.

Theorem 12. *For every $p \in (0, 1/2)$, and every $\varepsilon > 0$, there is an explicit family of binary codes of rate at least $1 - H(p) - \varepsilon$ that are efficiently p-decodable with average error $\exp(-\Omega(\varepsilon^2 N/\log^2 N))$, where N is the block length of the code.*

Proof. We need to modify the stochastic code so that the decoder for the stochastic code can also recover all the random bits used at the encoding. We already showed (Lemma 11) that the random string ω comprising the control information is in fact correctly recovered with high probability. However, there is no hope to recover all the random strings $r_1, r_2, \ldots, r_\ell$ used by the various SC encodings, since some of these control blocks could be completely erased.

We solve this problem by using correlated random strings r_i for the ℓ encodings $\mathsf{SC}(A_i, r_i)$ in Step 12. Specifically, we generate a random string ρ of the same length as the control information ω, that is, $3\varepsilon^2 N$ bits. We compute the strings r_i by encoding ρ with (almost) the same Reed-Solomon code used to encode ω. Because the r_i's need to be $2\log N$ bits long, we increase the alphabet size to N^2 (from N); this halves the rate of the code, bringing it to $\frac{\varepsilon}{16}$. The RS encoding provides both redundancy (so we can recover all of the r_i given only a few of them) and enough independence so that the number of incorrectly decoded control blocks is still tightly concentrated around its mean.

Step 12 of the encoding algorithm becomes:

```
12 a: $\rho \leftarrow \{0,1\}^{3\varepsilon^2 N}$.
12 b: $(r_1, ..., r_\ell) \leftarrow \mathsf{RS}_{\mathbb{F}^2, S, \ell, |\omega|/2\log N}(\rho)$.
12 c: Set $C_i \leftarrow \mathsf{SC}(A_i, r_i)$.
```

Suppose for a moment that the decoder correctly recovers enough of the pairs (A_i, r_i) to compute the control information ω correctly. Then the decoder can also compute the string ρ since the r_i are encoded using the same code. The strings ω and ρ are the only random bits used by the encoder, so the decoder's task is then complete.

To analyze the modified code, we can therefore mimic the analysis of the original code. Lemmas 8, 10 and 11 require no modification; their proofs do not depend on the distribution of the random bits used to encode the control blocks. Only Lemma 9, which bounds the number of correctly and incorrectly decoded control blocks, needs a new proof. The main idea is that the symbols of the Reed-Solomon encoding of a random string of t symbols are t-wise independent.

Lemma 13 (Modified Lemma 9)**.** *Suppose that Step 12 of stochastic encoder is modified as above. Then for any e, T such that T is good for e, with probability at least $1 - \exp(-\Omega(\varepsilon^2 N/\log N))$ over the random coins $(r_1, r_2, \ldots, r_\ell)$ used by the ℓ SC encodings, we have:*

1. *The number of control blocks correctly decoded by* SC-DECODE *is at least $\frac{\varepsilon \ell}{4}$.*

2. *The number of* erroneously *decoded control blocks is less than* $\frac{\varepsilon\ell}{24}$.
   (By erroneously decoded, we mean that SC-DECODE *outputs neither* $\perp$ *nor the correct message.)*

Proof. Let the indicator random variable Y_i be 1 if control block i either (i) has "low" error rate (at most $p+\varepsilon$) and yet is not correctly decoded or (ii) has high error rate, and is not decoded as $\perp$. Let $\mathbf{Y}$ be the sum of the Y_i's. As in the original proof, it suffices to bound $\mathbf{Y}$ by $\frac{\varepsilon\ell}{24}$ to imply events 1 and 2 of the statement.

Individually, the r_i's are still uniform and independent of the error pattern, so each Y_i has probability at most $1/N$ of being 1. The r_i's are no longer mutually independent, but they are $\frac{\varepsilon\ell}{16}$-wise independent, since the symbols of a RS codeword of rate $\frac{\varepsilon}{16}$ and length ℓ are $\frac{\varepsilon\ell}{16}$-wise independent (they are the evaluations of a random polynomial of degree $\frac{\varepsilon\ell}{16}-1$).

We can apply a concentration bound of Bellare and Rompel [2, Lemma 2.3] for sums of t-wise independent variables. In turns out to be convenient to use a slightly smaller value of t than we have available, in order to fit the form of the existing bounds. Set $t=\frac{\varepsilon\ell}{48}$. Using $\mu=E[\mathbf{Y}]\leqslant\frac{\ell}{N}$ and $A=\frac{\varepsilon\ell}{24}$, we obtain the bound

$$\Pr[\mathbf{Y}\geqslant\tfrac{\varepsilon\ell}{24}]\leqslant\left(\frac{t\mu+t^2}{A^2}\right)^{t/2}=\left(\frac{\frac{\varepsilon\ell^2}{48N}+\frac{\varepsilon^2\ell^2}{48^2}}{\frac{\varepsilon^2\ell^2}{24^2}}\right)^{\varepsilon\ell/96}\leqslant\exp(-\Omega(\varepsilon\ell))=\exp(-\Omega(\varepsilon^2N/\log N))\,.$$

This completes the proof of the lemma, and hence the proof of the theorem. □

□

References

[1] N. Alon, L. Babai, and A. Itai. A fast and simple randomized parallel algorithm for the maximal independent set problem. *J. Algorithms*, 7(4):567–583, 1986.

[2] M. Bellare and J. Rompel. Randomness-efficient oblivious sampling. In *Proceedings of the 35th Annual IEEE Symposium on Foundations of Computer Science*, pages 276–287, 1994.

[3] R. Cramer, Y. Dodis, S. Fehr, C. Padró, and D. Wichs. Detection of algebraic manipulation with applications to robust secret sharing and fuzzy extractors. In *Advances in Cryptology - EUROCRYPT, 27th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 471–488, 2008.

[4] I. Csiszár and P. Narayan. The capacity of the arbitrarily varying channel revisited: Positivity, constraints. *IEEE Transactions on Information Theory*, 34(2):181–193, 1988.

[5] I. Csiszár and P. Narayan. Capacity and decoding rules for classes of arbitrarily varying channels. *IEEE Transactions on Information Theory*, 35(4):752–769, 1989.

[6] Y. Z. Ding, P. Gopalan, and R. J. Lipton. Error correction against computationally bounded adversaries. *Manuscript*, 2004.

[7] P. Elias. List decoding for noisy channels. *Technical Report 335, Research Laboratory of Electronics, MIT*, 1957.

[8] P. Elias. Error-correcting codes for list decoding. *IEEE Transactions on Information Theory*, 37:5–12, 1991.

[9] G. D. Forney. *Concatenated Codes*. MIT Press, Cambridge, MA, 1966.

[10] Z. Galil, R. J. Lipton, X. Yu, and M. Yung. Computational error-correcting codes achieve shannon's bound explicitly. *Manuscript*, 1995.

[11] V. Guruswami. List decoding with side information. In *Proceedings of the 18th IEEE Conference on Computational Complexity (CCC)*, pages 300–309, July 2003.

[12] V. Guruswami. *Algorithmic Results in List Decoding*, volume 2 of *Foundations and Trends in Theoretical Computer Science (FnT-TCS)*. NOW publishers, January 2007.

[13] V. Guruswami, J. Hastad, M. Sudan, and D. Zuckerman. Combinatorial bounds for list decoding. *IEEE Transactions on Information Theory*, 48(5):1021–1035, 2002.

[14] V. Guruswami and A. Rudra. Explicit codes achieving list decoding capacity: Error-correction with optimal redundancy. *IEEE Transactions on Information Theory*, 54(1):135–150, January 2008.

[15] V. Guruswami and A. Rudra. Better binary list-decodable codes via multilevel concatenation. *IEEE Transactions on Information Theory*, 55(1):19–26, January 2009.

[16] E. Kaplan, M. Naor, and O. Reingold. Derandomized constructions of k-wise (almost) independent permutations. *Electronic Colloquium on Computational Complexity (ECCC)*, (002), 2006.

[17] M. Langberg. Private codes or succinct random codes that are (almost) perfect. In *Proceedings of the 45th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 325–334, 2004.

[18] M. Langberg. Oblivious communication channels and their capacity. *IEEE Transactions on Information Theory*, 54(1):424–429, 2008.

[19] A. Lapidoth and P. Narayan. Reliable communication under channel uncertainty. *IEEE Transactions on Information Theory*, 44(6):2148–2177, 1998.

[20] R. J. Lipton. A new approach to information theory. In *Proceedings of the 11th Annual Symposium on Theoretical Aspects of Computer Science*, pages 699–708, 1994.

[21] S. Micali, C. Peikert, M. Sudan, and D. A. Wilson. Optimal error correction against computationally bounded noise. In *Proceedings of the 2nd Theory of Cryptography Conference*, pages 1–16, 2005.

[22] W. W. Peterson. Encoding and error-correction procedures for Bose-Chaudhuri codes. *IEEE Transactions on Information Theory*, 6:459–470, 1960.

[23] C. E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423, 623–656, 1948.

[24] A. Smith. Scrambling adversarial errors using few random bits, optimal information reconciliation, and better private codes. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 395–404, 2007.

[25] S. P. Vadhan. Constructing locally computable extractors and cryptosystems in the bounded-storage model. *J. Cryptology*, 17(1):43–77, 2004.

[26] J. M. Wozencraft. List Decoding. *Quarterly Progress Report, Research Laboratory of Electronics, MIT*, 48:90–95, 1958.

[27] V. V. Zyablov and M. S. Pinsker. List cascade decoding. *Problems of Information Transmission*, 17(4):29–34, 1981 (in Russian); pp. 236-240 (in English), 1982.

A Ingredients for Main Construction

In this section, we will describe the various ingredients that we will need in our construction of capacity achieving AVC codes.

A.1 Constant rate codes for average error

By plugging in an appropriate explicit construction of list-decodable codes (with sub-optimal rate) into Theorem 5, we can also get the following explicit constructions of stochastic codes, albeit not at capacity. We will make use of these codes to encode blocks of logarithmic length control information in our final capacity-achieving explicit construction. The total number of bits in all these control blocks together will only be a small fraction of the total message length. So the stochastic codes encoding these blocks can have any constant rate, and this allows us to use any off-the-shelf explicit constant rate list-decodable code in Theorem 5 (in particular, we do *not* need a brute-force search for small list-decodable codes of logarithmic block length). We get the following claim by choosing $d = 1$ and picking C to be a binary linear $(\alpha, c_1(\alpha)/2)$-list decodable code in Theorem 5.

Proposition 14. *For every α, $0 < \alpha < 1/2$, there exists $c_0 = c_0(\alpha) > 0$ and $c_1 = c_1(\alpha) < \infty$ such that for all large enough integers b, there is an explicit stochastic code $\mathsf{SC}_{k,\alpha}$ of rate $1/c_0$ with encoding $E : \{0,1\}^b \times \{0,1\}^b \to \{0,1\}^{c_0 b}$ that is efficiently strongly α-decodable with probability $1 - c_1 2^{-b}$.*

Moreover, for every message and every error pattern of more than a fraction α of errors, the decoder for $\mathsf{SC}_{k,\alpha}$ returns $\perp$ and reports a decoding failure with probability $1 - c_1 2^{-b}$. Further, on input a uniformly random string y from $\{0,1\}^{c_0 b}$, the decoder for $\mathsf{SC}_{k,\alpha}$ returns $\perp$ with probability at least $1 - c_1 2^{-b}$ (over the choice of y).

Proof. The claim follows by choosing $d = 1$ and picking C to be a binary linear $(\alpha, c_1(\alpha)/2)$-list decodable code in Theorem 5. The claim about decoding a uniformly random input follows since the number of strings y which differ from some valid output of the encoder E is at most a fraction α of positions is at most $2^{2b} 2^{H(\alpha) c_0 b}$. By standard entropy arguments, we have $(1 - H(\alpha)) c_0 b + \log(c_1(\alpha)/2) \geqslant 3b$ (since the code encodes $3b$ bits, the capacity is $1 - H(\alpha)$, and at most $\log(c_1(\alpha)/2)$ additional bits of side information are necessary to disambiguate the true message from the list). We conclude that the probability that a random string gets accepted by the decoder is at most $2^{-b} \cdot 2^{\log(c_1(\alpha)/2)} \leqslant c_1 2^{-b}$. □

A.2 Reed-Solomon codes

If $\mathbb{F}$ is a finite field with at least n elements, and $S = (\alpha_1, \alpha_2, \ldots, \alpha_n)$ is a sequence of n *distinct* elements from $\mathbb{F}$, the Reed-Solomon encoding, $\mathsf{RS}_{\mathbb{F},S,n,k}(m)$, or just $\mathsf{RS}(m)$ when the other parameters are implied, of a message $m = (m_0, m_1, \ldots, m_{k-1}) \in \mathbb{F}^k$ is given by

$$\mathsf{RS}_{\mathbb{F},S,n,k}(m) = (f(\alpha_1), f(\alpha_2), \cdots, f(\alpha_n)) \; . \tag{1}$$

where $f(X) = m_0 + m_1 X + ... + m_{k-1} X^{k-1}$. The following is a classic result on unique decoding Reed-Solomon codes [22], stated as a noisy polynomial reconstruction algorithm.

Proposition 15. *There is an efficient algorithm with running time polynomial in n and $\log |\mathbb{F}|$ that given n distinct pairs $(\alpha_i, a_i) \in \mathbb{F}^2$, $1 \leqslant i \leqslant n$, and an integer $k < n$, finds the unique polynomial f of degree at most k, if any, that satisfies $f(\alpha_i) = a_i$ for more than $\frac{n+k}{2}$ values of i. Note that this condition can also be expressed as $\left|\{i : f(\alpha_i) = a_i\}\right| - \left|\{i : f(\alpha_i) \neq a_i)\}\right| > k$.*

A.3 Pseudorandom constructs

A.3.1 Samplers

Let $[N] = \{1, 2, \ldots, N\}$. If $B \subseteq [N] \to \{0,1\}$ has density μ (i.e., μN elements), then standard tail bounds imply that for a random subset $T \subseteq [N]$ of size ℓ, the density of $B \cap T$ is within $\pm\theta$ of μ with overwhelming probability (at least $1 - \exp(-c_\theta \ell)$). But picking a random subset of size ℓ requires $\approx \ell \log(N/\ell)$ random bits. The following shows that a similar effect can be achieved by a sampling procedure that uses fewer random bits. The idea is the well known one of using random walks of length ℓ in a low-degree expander on N vertices. This could lead to repeated samples while we would like ℓ distinct samples. This can be achieved by picking slightly more than ℓ samples and discarding the repeated ones. The result below appears in this form as Lemma 8.2 in [25].

Proposition 16. *For every $N \in \mathbb{N}$, $0 < \theta < \mu < 1$, $\gamma > 0$, and integer $\ell \geqslant \ell_0 = \Omega(\frac{1}{\theta^2}\log(1/\gamma))$, there exists an explicit efficiently computable function $\mathsf{Samp} : \{0,1\}^\sigma \to [N]^\ell$ where $\sigma \leqslant O(\log N + \ell \log(1/\theta))$ with the following property:*

For every $B \subseteq [N]$ of size at least μN, with probability at least $1 - \gamma$ over the choice of a random $s \in \{0,1\}^\sigma$, $|\mathsf{Samp}(s) \cap B| \geqslant (\mu - \theta)|\mathsf{Samp}(s)|$.

We will use the above samplers to pick the random positions in which the blocks holding encoded control information are interspersed with the data blocks. The sampling guarantee will ensure that a reasonable fraction of the control blocks have no more than a fraction $p + \varepsilon$ of errors when the total fraction of errors is at most p.

A.3.2 Almost t-wise independent permutations

Definition 7. A distribution $\mathcal{D}$ on S_n (the set of permutations of $\{1, 2, \ldots, n\}$) is said to *almost t-wise independent* if for every $1 \leqslant i_1 < i_2 < \cdots < i_t \leqslant n$, the distribution of $(\pi(i_1), \pi(i_2), \ldots, \pi(i_t))$ for π chosen according to $\mathcal{D}$ has statistical distance at most 2^{-t} for the uniform distribution on t-tuples of t distinct elements from $\{1, 2, \ldots, n\}$. □

A uniformly random permutation of $\{1, 2, \ldots, n\}$ takes $\log n! = \Theta(n \log n)$ bits to describe. The following result shows that almost t-wise independent permutations can have much shorter descriptions.

Proposition 17 ([16]). *For all integers $1 \leqslant t \leqslant n$, there exists $D = O(t \log n)$ and an explicit map $\mathsf{KNR} : \{0,1\}^\sigma \to S_n$, computable in time polynomial in n, such that the distribution $\mathsf{KNR}(s)$ for random $s \in \{0,1\}^\sigma$ is almost t-wise independent.*

A.3.3 t-wise independent bit strings

We will also need small sample spaces of binary strings in $\{0,1\}^n$ which look uniform for any t positions.

Definition 8. A distribution $\mathcal{D}$ on $\{0,1\}^n$ is said to *t-wise independent* if for every $1 \leqslant i_1 < i_2 < \cdots < i_t \leqslant n$, the distribution of $(x_{i_1}, x_{i_2}, \ldots, x_{i_t})$ for $x = (x_1, x_2, \ldots, x_n)$ chosen according to $\mathcal{D}$ equals the uniform distribution on $\{0,1\}^t$. □

Using evaluations of degree t polynomials over a field of characteristic 2, the following well known fact can be shown. We remark that the optimal seed length is about $\frac{t}{2}\log n$ and was achieved in [1], but we can work with the weaker $O(t\log n)$ seed length.

Proposition 18. *Let n be a positive integer, and let $t \leqslant n$. There exists $\sigma \leqslant O(t\log n)$ and an explicit map $\mathsf{POLY}_t : \{0,1\}^\sigma \to \{0,1\}^n$, computable in time polynomial in n, such that the distribution $\mathsf{POLY}_t(s)$ for random $s \in \{0,1\}^\sigma$ is t-wise independent.*

A.4 Capacity achieving codes for t-wise independent errors

Forney [9] constructed binary linear concatenated codes that achieve the capacity of the binary symmetric channel BSC_p. Smith [24] showed that these codes also correct patterns of at most a fraction p of errors w.h.p. when the error locations are distributed in a t-wise independent manner for large enough t. The precise result is the following.

Proposition 19. *For every p, $0 < p < 1/2$ and every $\varepsilon > 0$, there is an explicit family of binary linear codes of rate $R \geqslant 1 - H(p) - \varepsilon$ such that a code $\mathsf{REC} : \{0,1\}^{Rn} \to \{0,1\}^n$ of block length n in the family provides the following guarantee. There is a polynomial time decoding algorithm Dec such that for every message $m \in \{0,1\}^{Rn}$, every error vector $e \in \{0,1\}^n$ of Hamming weight at most pn, and every almost t-wise independent distribution $\mathcal{D}$ of permutations of $\{1,2,\ldots,n\}$, we have*

$$\mathsf{Dec}(\mathsf{REC}(m) + \pi(e)) = m$$

with probability at least $1 - 2^{-\Omega(\varepsilon^2 t)}$ over the choice of a permutation $\pi \in_R \mathcal{D}$, as long as $\omega(\log n) < t < \varepsilon n/10$. (Here $\pi(e)$ denotes the permuted vector: $\pi(e)_i = e_{\pi(i)}$.)

We will use the above codes (which we denote REC, for "random-error code") to encode the actual data in our stochastic code construction.