Optimal Rate Code Constructions for Computationally Simple Channels

VENKATESAN GURUSWAMI, Carnegie Mellon University
ADAM SMITH, Pennsylvania State University

We consider coding schemes for *computationally bounded* channels, which can introduce an arbitrary set of errors as long as (a) the fraction of errors is bounded with high probability by a parameter p and (b) the process that adds the errors can be described by a sufficiently "simple" circuit. Codes for such channel models are attractive since, like codes for standard adversarial errors, they can handle channels whose true behavior is *unknown* or *varying* over time.

For two classes of channels, we provide explicit, efficiently encodable/decodable codes of optimal rate where only *in*efficiently decodable codes were previously known. In each case, we provide one encoder/decoder that works for *every* channel in the class. The encoders are randomized, and probabilities are taken over the (local, unknown to the decoder) coins of the encoder and those of the channel.

Unique decoding for additive errors: We give the first construction of a polynomial-time encodable/decodable code for *additive* (a.k.a. *oblivious*) channels that achieve the Shannon capacity $1 - H(p)$. These are channels that add an arbitrary error vector $e \in \{0,1\}^N$ of weight at most pN to the transmitted word; the vector e can depend on the code but not on the randomness of the encoder or the particular transmitted word. Such channels capture binary symmetric errors and burst errors as special cases.

List decoding for polynomial-time channels: For every constant $c > 0$, we construct codes with optimal rate (arbitrarily close to $1 - H(p)$) that efficiently recover a short list containing the correct message with high probability for channels describable by circuits of size at most N^c. Our construction is not fully explicit but rather Monte Carlo (we give an algorithm that, with high probability, produces an encoder/decoder pair that works for all time N^c channels). We are not aware of any channel models considered in the information theory literature other than purely adversarial channels, which require more than linear-size circuits to implement. We justify the relaxation to list decoding with an impossibility result showing that, in a large range of parameters ($p > 1/4$), codes that are uniquely decodable for a modest class of channels (online, memoryless, nonuniform channels) cannot have positive rate.

Categories and Subject Descriptors: E.4 [**Coding and Information Theory**]: Error Control Codes; F.1.3 [**Computation by Abstract Devices**]: Complexity Measures and Classes

General Terms: Algorithms, Theory

Additional Key Words and Phrases: Computationally bounded channels, error-correcting codes, pseudorandomness

V. G. was supported by a Packard Fellowship and NSF Grants No. CCF-0953155 and No. CCF-0963975. A. S. was supported by NSF Grants No. CCF-0747294 and No. CCF-0729171. An extended abstract appeared in the *Proceedings of the 51st Annual IEEE Symposium on Foundations of Computer Science* (FOCS), October 2010. The most recent full version of this article is maintained at http://arxiv.org/abs/1004.4017.
Authors' addresses: V. Guruswami, Computer Science Department, Carnegie Mellon University, Pittsburgh, PA 15213, USA; email: guruswami@cmu.edu; A. Smith, Computer Science and Engineering Department, Pennsylvania State University, University Park, PA 16802, USA; email: asmith@psu.edu.

DOI: http://dx.doi.org/10.1145/2936015

ACM Reference Format:
Venkatesan Guruswami and Adam Smith. 2016. Optimal rate code constructions for computationally simple channels. J. ACM 63, 4, Article 35 (September 2016), 37 pages.
DOI: http://dx.doi.org/10.1145/2936015

1. INTRODUCTION

The theory of error-correcting codes has had two divergent schools of thought, dating back to its origins, based on the underlying model of the noisy channel. Shannon's theory [Shannon 1948] modeled the channel as a stochastic process with a known probability law. Hamming's work [Hamming 1950] suggested a worst-case/adversarial model, where the channel is subject only to a limit on the number of errors it may cause.

These two approaches share several common tools; however, in terms of quantitative results, the classical results in the harsher Hamming model are much weaker. For instance, for the binary symmetric channel that flips each transmitted bit independently with probability $p < 1/2$, the optimal rate of reliable transmission is known to be the Shannon capacity $1 - H(p)$, where $H(\cdot)$ is the binary entropy function [Shannon 1948]. Concatenated codes [Forney 1966] and polar codes [Arikan 2009] can transmit at rates arbitrarily close to this capacity and are efficiently decodable. In contrast, for adversarial channels that can corrupt up to a fraction p of symbols in an *arbitrary* manner, the optimal rate is unknown in general, though it is known for all $p \in (0, \frac{1}{2})$ that the rate has to be much smaller than the Shannon capacity. In fact, for $p \in [\frac{1}{4}, \frac{1}{2})$, the achievable rate over an adversarial channel is asymptotically zero, while the Shannon capacity $1 - H(p)$ remains positive.

Codes that tolerate worst-case errors are attractive because they assume nothing about the distribution of the errors introduced by the channel, only a bound on the number of errors. Thus, they can be used to transmit information reliably over a large range of channels whose true behavior is unknown or varies over time. In contrast, codes tailored to a specific channel model tend to fail when the model changes. For example, concatenated codes with a high rate outer code, which can transmit efficiently and reliably at the Shannon capacity with i.i.d. errors, fail miserably in the presence of *burst* errors that occur in long runs.

In this article, we consider several intermediate models of uncertain channels, as a meaningful and well-motivated bridge between the Shannon and Hamming models. Specifically, we consider *computationally bounded* channels, which can introduce an arbitrary set of errors as long as (a) the total fraction of errors is bounded by p with high probability and (b) the process which adds the errors can be described by a sufficiently "simple" circuit. The idea and motivation behind these models is that natural processes may be mercurial but are not computationally intensive. These models are powerful enough to capture natural settings like i.i.d. and burst errors but weak enough to allow efficient communication arbitrarily close to the Shannon capacity $1 - H(p)$. The models we study, or close variants, have been considered previously-see Section 2 for a discussion of related work. The computational perspective we espouse is inspired by the works of Lipton [1994] and Micali et al. [2005].

For two classes of channels, we construct efficiently encodable and decodable codes of optimal rate (arbitrarily close to $1 - H(p)$) where only *in*efficiently decodable codes were previously known. In each case, we provide one encoder/decoder that works for *every* channel in the class. In particular, our results apply even when the channel's behavior depends on the code. One of our constructions is fully explicit, while the other is probabilistic ("Monte Carlo").

Structure of this article. We first describe the models and our results briefly (Section 1.1) and outline our main technical contributions (Section 1.2). In Section 2, we describe related lines of work aimed at handling (partly) adversarial errors with rates

near Shannon capacity. We state precise definitions in Section 3 and results in Section 4. Section 5 describes our list-decoding-based code constructions for recovery from additive errors. As the needed list-decodable codes are not explicitly known, this only gives an existential result. Section 6 describes the approach behind our efficient constructions at a high level. The remainder of the article describes and analyzes the constructions in detail, in order of increasing channel strength: additive errors (Section 7) and time-bounded errors (Section 8). The appendices contain extra details on the building blocks in our constructions (A), results for the "average" error criterion (B), and our impossibility result showing the necessity of list decoding for error fractions exceeding 1/4 (C), respectively.

1.1. Summary of Results

The encoders we construct are *stochastic* (that is, randomized). The encoder/decoder pair are required to succeed with high probability over the local (unknown to the decoder) coins of the encoder and the choices of the channel. The same encoder/decoder pair must work for all channels in a given family and for all messages of a given length; in particular, the message may be chosen adversarially and known to the channel. Our results do not assume any setup or shared randomness between the encoder and decoder. We provide a precise definition in Section 3.

Unique decoding for additive channels. We give the first explicit construction of stochastic codes with polynomial-time encoding/decoding algorithms that approach the Shannon capacity $1 - H(p)$ for *additive* (a.k.a. *oblivious*) channels. These are channels that add an arbitrary error vector $e \in \{0,1\}^N$ of Hamming weight at most pN to the transmitted codeword (of length N). The error vector may depend on the code and the message but, crucially, not on the encoder's local random coins. Additive errors capture binary symmetric errors as well as certain models of correlated errors, like burst errors. For a deterministic encoder, the additive error model is equivalent to the usual adversarial error model. A randomized encoder is thus necessary to achieve the Shannon capacity.

We also provide a novel, simple proof that (inefficient) capacity-achieving codes *exist* for additive channels. We do so by combining linear list-decodable codes with rate approaching $1 - H(p)$ (known to exist, but not known to be efficiently decodable) with a special type of authentication scheme. Previous existential proofs relied on complex random coding arguments [Csiszár and Narayan 1988a; Langberg 2008]; see the discussion of related work below.

Necessity of list decoding for richer channel models. The additive errors model postulates that the error vector has to be picked obliviously before seeing the codeword. To model more complex processes, we will allow the channel to see the codeword and decide on the error as a function of the codeword. We will stipulate a computational restriction on how the channel might decide what error pattern to cause. The simplest restriction (beyond the additive/oblivious case) is a "memoryless" channel that decides the action on the ith bit based only on that bit (and perhaps some internal state that is oblivious to the codeword).

First, we show that even against such memoryless channels, reliable *unique* decoding with positive rate is impossible when $p > 1/4$. The idea is that even a memoryless adversary can make the transmitted codeword difficult to distinguish from a different, random codeword. The proof relies on the requirement that a single code must work for all channels, since the "hard" channel depends on the code.

Thus, to communicate at a rate close to $1 - H(p)$ for all p, we consider the relaxation to *list decoding*: The decoder is allowed to output a small list of messages, one of which is correct. List-decodable codes with rate approaching $1 - H(p)$ are known to exist even for adversarial errors [Zyablov and Pinsker 1982; Elias 1991]. However, constructing

efficient (i.e., polynomial-time encodable and decodable) binary codes for list decoding with near-optimal rate is a major open problem.[1] Therefore, our goal is to exploit the computational restrictions on the channel to get efficiently list-decodable codes.

Computationally bounded channels. We consider channels whose behavior on N-bit inputs is described by a circuit of size $\mathsf{T}(N)$ (for example, $\mathsf{T}(N) = O(N^2)$). We sometimes call this the *time-bounded* model and refer to $\mathsf{T}(\cdot)$ as a time bound. We do not know of any channel models considered in the information theory literature other than purely adversarial channels, which require more than linear time to implement.

We also discuss *(online) space-bounded* channels; these channels make a single pass over the transmitted codeword, deciding which locations to corrupt as they go, and are limited to storing at most $S(N)$ bits (that is, they can be describe by one-pass branching programs of width at most $2^{S(N)}$). Logarithmic space channels, in particular, can be realized by polynomial-size circuits.

List decoding for polynomial-time channels. Our main contribution for time-bounded channels is a construction of polynomial-time encodable and list-decodable codes that approach the optimal rate for channels whose time bound is a polynomial in the block length N. Specifically, we give an efficiently computable encoding function, Enc, that stochastically encodes the message m into $\mathsf{Enc}(m;r)$, where r is *private* randomness chosen at the encoder, such that for every message m and time N^c-bounded channel W, the decoder takes $\mathsf{W}(\mathsf{Enc}(m;r))$ as input and returns a small list of messages that, with high probability over r and the coins of the channel, contains the real message m. The size of the list is polynomial in $1/\varepsilon$, where $N(1-H(p)-\varepsilon)$ is the length of the transmitted messages. We stress that the decoder does *not* know the choice of random bits r made at the encoder.

The construction of our encoding function $\mathsf{Enc}(\cdot,\cdot)$ is *Monte Carlo*—we give a randomized algorithm that, with high probability, produces an encoder/decoder pair $\mathsf{Enc}/\mathsf{Dec}$ reliably communicate across all time-N^c-bounded channels (Definition 3.3). The randomized construction is polynomial time, and the resulting function $\mathsf{Enc}(m;r)$ can be computed from m, r in deterministic polynomial time. However, we do not know how to efficiently check and certify that a particular encoder Enc has the claimed properties. Obtaining fully explicit constructions remains an intriguing open problem. In the case of polynomial-time bounded channels, it seems unlikely that one can solve the problem without additional complexity assumptions (such as the existence of certain pseudorandom generators). In the case of online logspace-bounded channel, it may be possible to obtain fully explicit constructions without complexity assumptions (using, for example, Nisan's pseudorandom generators for logspace). We will elaborate on this aspect in Section 9.

One technicality is that the decoder need *not* return all words within a distance pN of the received word (as is the case for the standard "combinatorial" notion of list decoding), but it *must* return the correct message as one of the candidates with high probability. This notion of list decoding is natural for stochastic codes in communication applications. In fact, it is exactly the notion that is needed in constructions that "sieve" the list, such as in Guruswami [2003] and Micali et al. [2005], and one application of our results is a *uniquely* decodable code for the public-key model (assuming a specific polynomial-time bound), strengthening results of Micali et al. [2005]. See the discussion of related work in Section 2 for a more precise statement.

For both the additive and time-bounded models, our constructions require new methods for applying tools from cryptography and derandomization to coding-theoretic problems. We give a brief high-level discussion of these techniques next. An expanded overview the approach behind our code construction appears in Section 6.

[1]Over large alphabets, explicit optimal rate list-decodable codes *are* known.

1.2. Techniques

Control/payload construction. In our constructions, we develop several new techniques. The first is a novel "reduction" from the standard coding setting with no setup to the setting of *shared secret randomness*. In models in which errors are distributed evenly, such a reduction is relatively simple [Ahlswede 1978]; however, this reduction fails against adversarial errors. Instead, we show how to *hide* the secret randomness (the *control information*) inside the main codeword (the *payload*) in such a way that the decoder can learn the control information but (a) the control information remains hidden to a bounded channel and (b) its encoding is robust to a certain, weaker class of errors. We feel this technique should be useful in other settings of bounded adversarial behavior.

Our reduction can also be viewed as a novel way of bootstrapping from "small" codes, which can be decoded by brute force, to "large" codes, which can be decoded efficiently. The standard way to do this is via concatenation; unfortunately, concatenation does not work well even against mildly unpredictable models, such as the additive error model.

Pseudorandomness. Second, our results further develop a surprising connection between coding and pseudorandomness. *Hiding* the "control information" from the channel requires us to make different settings of the control information *indistinguishable* from the channel's point of view. Thus, our proofs apply techniques from cryptography together with constructions of pseudorandom objects (generators, permutations, and samplers) from derandomization. Typically, the "tests" that must be fooled are compositions of the channel (which we assume has low complexity) with some subroutine of the decoder (which we design to have low complexity). The connection to pseudorandomness appeared in a simpler form in the previous work on bounded channels [Lipton 1994; Galil et al. 1995; Micali et al. 2005]; our use of this connection is significantly more delicate.

The necessary pseudorandom permutations and samplers can be explicitly constructed, and the construction of the needed complexity-theoretic pseudorandom generators is where we uses randomness in our construction. For the case of additive errors, information-theoretic objects that we can construct deterministically efficiently (such as t-wise-independent strings) suffice and we get a fully explicit construction.

2. BACKGROUND AND RELATED PREVIOUS WORK

There are several lines of work aimed at handling adversarial, or partly adversarial, errors with rates near the Shannon capacity. We survey them briefly here and highlight the relationship to our results.

List decoding. List decoding was introduced in the late 1950s [Elias 1957; Wozencraft 1958] and has witnessed a lot of recent algorithmic work (cf. the survey [Guruswami 2007]). Under list decoding, the decoder outputs a small list of messages that must include the correct message. Random coding arguments demonstrate that there exist binary codes of rate $1 - H(p) - \varepsilon$ that can tolerate pN adversarial errors if the decoder is allowed to output a list of size $O(1/\varepsilon)$ [Elias 1991; Zyablov and Pinsker 1982; Guruswami et al. 2002]. The explicit construction of binary list-decodable codes with rate close to $1 - H(p)$, however, remains a major open question. We provide such codes for the special case of corruptions introduced by space- or time-bounded channels.

Adding Setup—Shared Randomness. Another relaxation is to allow randomized coding strategies where the sender and receiver share "secret" randomness, *hidden from the channel*, which is used to pick a particular, deterministic code at random from a family of codes. Such randomized strategies were called *private codes* in Langberg [2004]. Using this secret shared randomness, one can transmit at rates approaching

$1 - H(p)$ against adversarial errors (for example, by randomly permuting the symbols and adding a random offset [Lipton 1994; Langberg 2004]). Using explicit codes achieving capacity on the BSC_p [Forney 1966], one can even get such randomized codes of rate approaching $1 - H(p)$ explicitly (although getting an explicit construction with $o(n)$ randomness remains an open problem [Smith 2007]). A related notion of setup is the *public key* model of Micali et al. [2005], in which the sender generates a public key that is known to the receiver and possibly to the channel. This model only makes sense for computationally bounded channels, discussed below.

Our constructions are the first (for all three models) that achieve rate $1 - H(p)$ with efficient decoding and no setup assumptions.

AVCs: Oblivious, nonuniform errors. A different approach to modeling uncertain channels is embodied by the rich literature on *arbitrarily varying channels* (AVCs), surveyed in Lapidoth and Narayan [1998]. Despite being extensively investigated in the information theory literature, AVCs have not received much algorithmic attention.

An AVC is specified by a finite state space $\mathcal{S}$ and a family of memoryless channels $\{W_s : s \in \mathcal{S}\}$. The channel's behavior is governed by its state, which is allowed to vary arbitrarily. The AVC's behavior in a particular execution is specified by a vector $\vec{s} = (s_1, \ldots, s_N) \in \mathcal{S}^N$: The channel applies the operation W_{s_i} to the ith bit of the codeword. A code for the AVC is required to transmit reliably with high probability for every sequence $\vec{s}$, possibly subject to some state constraint. Thus AVCs model uncertainty via the *nonuniform* choice of the state vector $\vec{s} \in \mathcal{S}^N$. However—and this is the one of the key differences that makes the bounded space model more powerful—the choice of vector $\vec{s}$ in an AVC is *oblivious* to the codeword; that is, the channel cannot look at the codeword to decide the state sequence.

The additive errors channel we consider *is* captured by the AVC framework. Indeed, consider the simple AVC where $\mathcal{S} = \{0, 1\}$ and when in state s, the channel adds s mod 2 to the input bit. With the state constraint $\sum_{i=1}^{N} s_i \leqslant pN$ on the state sequence $(s_1, s_2, \ldots, s_N)$ of the AVC, this models additive errors, where an *arbitrary* error vector e with at most p fraction of 1's is added to the codeword by the channel, but e is chosen *obliviously* of the codeword.

Csiszár and Narayan determined the capacity of AVCs with state constraints [Csiszár and Narayan 1988b, 1989]. In particular, for the additive case, they showed that random codes can achieve rate approaching $1 - H(p)$ while correcting any specific error pattern e of weight pN with high probability.[2] Note that codes providing this guarantee *cannot* be linear, since the bad error vectors are the same for all codewords in a linear code. The decoding rule used in Csiszár and Narayan [1988b] to prove this claim was quite complex, and it was simplified to the more natural closest codeword rule in Csiszár and Narayan [1989]. Langberg [2008] revisited this special case (which he called an *oblivious channel*) and gave another proof of the above claim, based on a different random coding argument.

As outlined above, we provide two results for this model. First, we give a new, more modular existential proof. More importantly, we provide the first explicit constructions of codes for this model that achieve the optimal rate $1 - H(p)$.

[2]The AVC literature usually discusses the "average error criterion," in which the code is deterministic but the message is assumed to be uniformly random and unknown to the channel. We prefer the "stochastic encoding" model, in which the message is chosen adversarially, but the encoder has local random coins. The stochastic encoding model strictly stronger than the Average error model as long as the decoder recovers the encoder's random coins along with message. The arguments of Csiszár and Narayan [1988b] and Langberg [2008] also apply to the stronger model.

Polynomial-time-bounded channels. In a different vein, Lipton [1994] considered channels whose behavior can be described by a polynomial-time algorithm. He showed how a small amount of secret shared randomness (the seed for a pseudorandom generator) could be used to communicate at the Shannon capacity over any polynomial-time channel that introduces a bounded number of errors. Micali et al. [2005] gave a similar result in a public key model; however, their result relies on efficiently list-decodable codes, which are only known with sub-optimal rate. Both results assume the existence of one-way functions and some kind of setup. On the positive side, in both cases the channel's time bound need not be known explicitly ahead of time; one gets a tradeoff between the channel's time and its probability of success.

Our list decoding result removes the setup assumptions of Lipton [1994] and Micali et al. [2005] at the price of imposing a specific polynomial bound on the channel's running time and relaxing to list decoding. However, our result also implies stronger *unique decoding* results in the public-key model [Micali et al. 2005]. Specifically, our codes can be plugged into the construction of Micali et al. to get unique decoding at rates up to the Shannon capacity when the sender has a public key known to the decoder (and possibly to the channel). The idea, roughly, is to sign messages before encoding them; see Micali et al. [2005] for details. We remark that while they make use of list-decodable codes in the usual sense where all close-by codewords can be found, our weaker notion (where we find a list that includes the original message with high probability) suffices for the application.

Ostrovsky et al. [2007] and Hemenway and Ostrovsky [2008] considered the construction of *locally* decodable codes in the presence of computationally bounded errors assuming some setup (private [Ostrovsky et al. 2007] and public [Hemenway and Ostrovsky 2008] keys, respectively). The techniques used for locally decodable codes differ substantially from those used in more traditional coding settings; we do not know if the ideas from our constructions cacn be used to remove the setup assumptions from Ostrovsky et al. [2007] and Hemenway and Ostrovsky [2008].

Logarithmic-space channels. Galil et al. [1995] considered a slightly weaker model, logarithmic space, that still captures most physically realizable channels. They modeled the channel as a finite automaton with polynomially many states. Using Nisan's generator for log-space machines [Nisan 1992], they removed the assumption of one-way functions from Lipton's construction in the shared randomness model [Lipton 1994].

In the initial version of this article, we considered nonuniform generalization of their model that also generalizes arbitrarily varying channels. Our result for polynomial-time-bounded channels, which implies a construction for logarithmic-space channels, removes the assumption of shared setup in the model of Galil et al. [1995] but achieves only list decoding. This relaxation is necessary for some parameter ranges, since unique decoding in this model is impossible when $p > 1/4$.

Causal channels. The logarithmic-space channel can also be seen as a restriction of *online*, or *causal*, channels, recently studied by Dey et al. [2008] and Langberg et al. [2009]. These channels make a single-pass through the codeword, introducing errors as they go. They are not restricted in either space usage or computation time. It is known that codes for online channels cannot achieve the Shannon rate; specifically, the achievable asymptotic rate is at most $\max(1 - 4p, 0)$ [Dey et al. 2008]. Our impossibility result, which shows that the rate of codes for time- or space-bounded channels, is asymptotically 0 for $p > \frac{1}{4}$, can be seen as a partial extension of the online channels results of Dey et al. [2008], Langberg et al. [2009], and Dey et al. [2012] to computationally bounded channels, though our proof technique differs considerably. Recent work ([Dey et al. 2012; Chen et al. 2015]), subsequent to the original appearance of

this article, gives matching upper and lower bounds for the achievable rate of online channels.

3. DEFINITIONS

A *stochastic binary code of rate* $R \in (0,1)$*, randomness length* b *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 b additional random bits, into an N-bit codeword. Here, N and b are integers, and we assume for simplicity that RN is an integer. We normally think of the code as being paired with a decoding algorithm Dec which maps $\{0,1\}^N$ to either a single string (for unique decoding), or a list of strings (for list decoding), in $\{0,1\}^{RN}$.

Stochastic codes are useful in the context of communication against certain unknown channels (see, e.g., the survey [Lapidoth and Narayan 1998]). For some such channels, the capacity of deterministic codes over a particular class of channels may be smaller under the maximum error probability criterion than under the average error probability criterion (that is, it may be much harder to transmit the worst-case message than a random one). Randomization at the encoder can allow achieving the larger capacity even with a worst-case message.

A *channel* is a possibly randomized process W that maps $\{0,1\}^N$ into (distributions on) $\{0,1\}^N$. A channel W is pN*-bounded* if, for all inputs c, the string $W(c) \oplus c$ has weight at most pN.

We discuss several families of (bounded) channels, listed here in increasing strength:

—Additive channels: channels W_e such that $\mathsf{W}_e(c) = c \oplus e$ for a fixed string $e \in \{0,1\}^N$. The channel is bounded iff weight$(e) \leqslant pN$.
—Time-T channels: channels W implementable by a randomized circuit with at most T gates.
—Adversarial channels (denoted ADV_p): the family of all pN-bounded channels.

Definition 3.1. A stochastic code $(\mathsf{Enc}, \mathsf{Dec})$ *uniquely decodes errors from a channel* W *with probability* $1-\delta$ if the decoder outputs a single string and, for all $m \in \{0,1\}^{RN}$,

$$\Pr_{\omega, coins_{\mathsf{W}}} (\mathsf{Dec}(\mathsf{W}(\mathsf{Enc}(m;\omega))) = m) \geqslant 1-\delta\,,$$

where the probability is taken over $\omega \leftarrow \{0,1\}^b$ and any randomness used by W. A stochastic code *strongly* decodes a channel if the decoder recovers both m and the randomness ω of the encoder.

A stochastic code $(\mathsf{Enc}, \mathsf{Dec})$ *L-list decodes errors from a channel* W *with probability* $1-\delta$ if the decoder outputs a set of L strings and, for all $m \in \{0,1\}^{RN}$,

$$\Pr_{\omega, coins_{\mathsf{W}}} (m \in \mathsf{Dec}(\mathsf{W}(\mathsf{Enc}(m;\omega)))) \geqslant 1-\delta\,,$$

where the probability is taken over $\omega \leftarrow \{0,1\}^b$ and any randomness used by W.

Finally, a stochastic code (uniquely or list) decodes errors from a family $\mathcal{W}$ of channels if the condition holds for all channels $\mathsf{W} \in \mathcal{W}$ in the family (with the same decoder Dec).

If a code decodes a family of channels, then the message m may depend on the channel (and the description of the encoder and decoder), since decoding must work for all pairs (m, W) in $\{0,1\}^{RN} \times \mathcal{W}$.

Relation to combinatorial notions. The notion of stochastic list decodability does not generally imply the "usual" notion of list decoding, since the decoder need not return all codewords within a given distance—it need only generate a list that contains the true message with high probability.

Nevertheless, for the adversarial channel family ADV_p, the existence of a stochastic encoder/decoder pair implies the existence of codes that satisfy the usual combinatorial notions of unique and list decodability. We recall the classical definition for completeness.

Definition 3.2 (*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 $2pn$.

Monte Carlo constructions. For some settings, our code constructions are not fully explicit but rather "Monte Carlo": We give a polynomial-time-randomized algorithm that outputs, with high probability, an encoder/decoder pair that works for all codes in a given family and for all messages of a given length. More precisely,

Definition 3.3. A polynomial-time algorithm A is a *Monte Carlo construction* of an N-bit, rate R, correctness probability $1 - \delta$ uniquely (respectively, list) decodable code for a family of channels $\mathcal{W}$ if, on inputs 1^N and 1^k and a string of random bits r, A outputs an N-bit, rate R stochastic code and

$$\Pr_{r \leftarrow \{0,1\}^*} \left(\begin{array}{c} (\mathsf{Enc}, \mathsf{Dec}) \text{ uniquely (respectively, list) decodes} \\ \text{errors from } \mathcal{W} \text{ with probability } 1 - \delta \\ \text{where } (\mathsf{Enc}, \mathsf{Dec}) = A(1^N, 1^k, r) \end{array} \right) \geqslant 1 - \frac{1}{k}.$$

We can think of the string r as a string chosen once and for all when the code is designed and available as part of the code description. Alternatively, we may think of r as shared *public* randomness—that is, r is uniformly random but may be known to the channel.

4. MAIN RESULTS

4.1. Codes for Worst-Case Additive Errors

Existential result via list decoding. We give a novel construction of stochastic codes for additive errors by combining *linear* list-decodable codes with a certain kind of authentication code called algebraic manipulation detection (AMD) codes. Such AMD codes can detect *additive corruption* with high probability and were defined and constructed for cryptographic applications in Cramer et al. [2008]. 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 and depend only on the error vector (and not on m, r). An additional 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.

The formal result is stated below. Details can be found in Section 5. The notation $\Omega_{p,\varepsilon}$ expresses an asymptotic lower bound in which p and ε are held constant.

THEOREM 4.1. *For every p, $0 < p < 1/2$ and every $\varepsilon > 0$, there exists a family of stochastic codes of rate $R \geqslant 1 - H(p) - \varepsilon$ and a deterministic (exponential time) decoder $\mathsf{Dec} : \{0,1\}^N \to \{0,1\}^{RN} \cup \{\perp\}$ such that for every $m \in \{0,1\}^{RN}$ and every error vector $e \in \{0,1\}^N$ of Hamming weight at most pN, $\Pr_r[\mathsf{Dec}(\mathsf{Enc}(m, r) + e) = m] \geqslant 1 - 2^{-\Omega_{p,\varepsilon}(N)}$. Moreover, when more than a fraction p of errors occur, the decoder is able to detect this and report a decoding failure ($\perp$) with probability at least $1 - 2^{-\Omega_{p,\varepsilon}(N)}$.*

Given an explicit family of linear binary codes of rate R that can be efficiently list-decoded from fraction p of errors with list-size bounded by a polynomial function in N, one can construct an explicit stochastic code of rate $R - o(1)$ with the above guarantee along with an efficient decoder.

Explicit, efficient codes achieving capacity. Explicit binary list-decodable codes of optimal rate are not known, so one cannot use the above connection to construct *explicit* stochastic codes of rate $\approx 1 - H(p)$ for pN additive errors. Nevertheless, we give an explicit construction of capacity-achieving stochastic codes against worst-case additive errors. The construction is described at a high level in Section 6 and in further detail in Section 7.

THEOREM 4.2. *For every $p \in (0, 1/2)$, every $\varepsilon > 0$, and infinitely many N, there is an explicit, efficient stochastic code of block length N and rate $R \geqslant 1 - H(p) - \varepsilon$ that corrects a p fraction of additive errors with probability $1 - o(1)$. Specifically, there are polynomial-time algorithms* Enc *and* Dec *such that for every message $m \in \{0,1\}^{RN}$ and every error vector e of Hamming weight at most pN, we have* $\Pr_r[\mathsf{Dec}(\mathsf{Enc}(m;r) + e) = m] = 1 - \exp(-\Omega_{p,\varepsilon}(N/\log^2 N))$.

A slight modification of our construction gives codes for the "average error criterion," in which the code is deterministic but the message is assumed to be uniformly random and unknown to the channel (see Appendix B).

4.2. Unique Decoding Is Impossible for Nonoblivious Channels when $p > \frac{1}{4}$

We exhibit a very simple "zero space" (aka memoryless) channel that rules out achieving any positive rate (i.e., the capacity is zero) when $p > 1/4$. In each position, the channel either leaves the transmitted bit alone, sets it to 0, or sets it to 1. The channel works by "pushing" the transmitted codeword towards a different valid codeword (selected at random). This simple channel adds at most $n/4$ errors *in expectation*. We can get a channel with a hard bound on the number of errors by allowing it logarithmic space. Our impossibility result can be seen as strengthening a result by Dey et al. [2008] for online channels in the special case where $p > 1/4$, though our proof technique, adapted from Ahlswede [1978], differs considerably. Appendix C contains the proof.

THEOREM 4.3 (IMPOSSIBILITY FOR $p > \frac{1}{4}$). *For every stochastic binary code* (Enc, Dec) *with block length n and rate $R = \omega(1/n)$ (that is, the size of the message space tends to infinity with n),*

(1) *there is a distribution over memoryless channels that alters at most $n/4$ bits in expectation and causes a decoding error for a uniformly random message with probability at least $\frac{1}{2} - o(1)$.*
(2) *for every $0 < \nu < \frac{1}{4}$, there is an online space-$\lceil \log(n) \rceil$ channel W_2 that alters at most $n(\frac{1}{4} + \nu)$ bits (with probability 1) and causes a decoding error for a uniformly random message with probability $\Omega(\nu)$.*

4.3. List-Decodable Codes for Polynomial-Time Channels

We next consider a very general noise model. The channel can look at the whole codeword, and effect any error pattern, with the only restriction being that the channel must compute the error pattern in polynomial time given the original codeword as input. In fact, we will even allow non-uniformity and require that the error pattern be computable by a polynomial size circuit.

THEOREM 4.4. *For all constants $\varepsilon > 0$, $p \in (0, 1/2)$, and $c \geqslant 1$, and for infinitely many integers N, there exists a Monte Carlo construction (succeeding with probability $1 - N^{-\Omega(1)}$) of a stochastic code of block length N and rate $R \geqslant 1 - H(p) - \varepsilon$ with $N^{O(c)}$ time-encoding / list-decoding algorithms* (Enc, Dec) *that have the following property: For all messages $m \in \{0,1\}^{RN}$, and all pN-bounded channels* W *that are implementable by a size $O(N^c)$ circuit,* $\mathsf{Dec}(\mathsf{W}(\mathsf{Enc}(m; r)))$ *outputs a list of at most* $\mathrm{poly}(1/\varepsilon)$ *messages that includes the real message m with probability at least $1 - N^{-\Omega(1)}$.*

5. SIMPLE, NONEXPLICIT 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 that achieve the capacity for additive errors. The linearity of the code is crucial for this construction. The other ingredient we need for the construction is a particular kind of authentication code, called an algebraic manipulation detection (AMD) code, that can detect *additive corruption* with high probability [Cramer et al. 2008].

Though we do not require it in the definition, our constructions in this section of stochastic codes from list-decodable codes will also have the desirable property that when the number of errors exceeds pn, with high probability the decoder will output a decoding failure rather than decoding incorrectly.

5.1. Algebraic Manipulation Detection Codes

The following is not the most general definition of AMD codes from Cramer et al. [2008] but suffices for our purposes.

Definition 5.1. 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 number. Let $G = G_1 \times G_2 \times G_3$ be the product group (with component-wise 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:

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

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 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 Cramer et al. [2008] achieves near optimal tag size and we will make use of it.

THEOREM 5.2. *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 $f^{(d)}_{\mathrm{AMD}} : \mathbb{F}^d \times \mathbb{F} \to \mathbb{F}$ given by $f^{(d)}_{\mathrm{AMD}}(x, r) = r^{d+2} + \sum_{i=1}^{d} x_i r^i$ is a $(\mathcal{G}, \frac{d+1}{q})$-AMD code with tag size $2 \log q$, where $\mathcal{G} = (\mathbb{F}^d, \mathbb{F}, \mathbb{F})$.*[3]

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

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

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 correcting up to a fraction p of worst-case additive errors.

THEOREM 5.3 (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 $f_{\mathrm{AMD}}^{(d)}$ be the function from Theorem 5.2 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*

$$E(m, r) = C\big(m, r, f_{\mathrm{AMD}}^{(d)}(m, r)\big).$$

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

Moreover, 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 $y = E(m, r) + e$ 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 $f_{\mathrm{AMD}}^{(d)}(m_i, r_i) = s_i$. If so, then it outputs (m_i, r_i), otherwise it outputs $\perp$.

Let us now analyze the above decoder D. First, consider the case when $\mathrm{wt}(e) \leqslant pn$. 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 $(m, r, f_{\mathrm{AMD}}^{(d)}(m, r))$; say, this happens for $i = 1$ w.l.o.g. Therefore, the condition $f_{\mathrm{AMD}}^{(d)}(m_1, r_1) = s_1$ 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 pre-image of c'_i, that is, $c'_i = C(\Delta_i)$. 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 $f_{\mathrm{AMD}}^{(d)}(m_i, r_i) = s_i$ 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 $f_{\mathrm{AMD}}^{(d)}(m_i, r_i) = s_i$ for $i > 1$ succeed, and the decoder thus correctly outputs $m_1 = m$.

In the case when $\mathrm{wt}(e) > pn$, the same argument shows that the check $f_{\mathrm{AMD}}^{(d)}(m_i, r_i) = s_i$ 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 against additive errors.

COROLLARY 5.4. *For every p, $0 < p < 1/2$ and every $\varepsilon > 0$, there exists a family of stochastic codes of rate at least $1 - H(p) - \varepsilon$, which strongly decode pN-bounded additive errors with probability at least $1 - 2^{-c(\varepsilon,p)n}$ where n is the block length and $c(\varepsilon, p)$ is a constant depending only on ε and p.*

Moreover, when more than a fraction p of errors occur, the code is able to detect this and report a decoding failure with probability at least $1 - 2^{-c(\varepsilon,p)n}$.

Remark 5.5. For the above construction, if the decoding succeeds, then it correctly computes in addition to the message m also the randomness r used at the encoder. So the construction also gives deterministic codes for the "average error criterion," where, for every error vector, all but an exponentially small fraction of messages are communicated correctly. See Appendix B for a discussion of codes for this model and their relation to stochastic codes for additive errors.

6. OVERVIEW OF EXPLICIT CONSTRUCTIONS

Codes for Additive Errors. Our result is obtained by combining several ingredients from pseudorandomness and coding theory. At a high level, the idea (introduced by Lipton [1994] in the context of shared randomness) 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 [Forney 1966]) can be used to communicate on ADV_p after the codeword's 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 used for 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, based solely 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 locations of 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, blocks

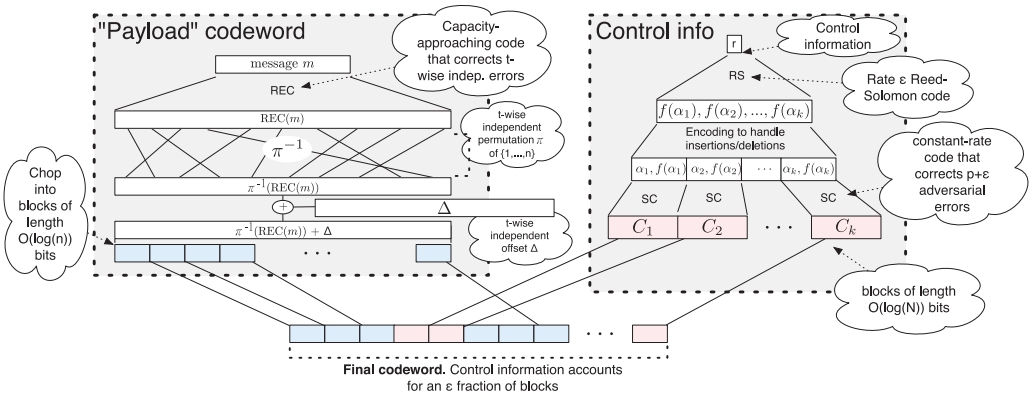

Fig. 1. Schematic description of encoder from Algorithm 1.

with too many errors will be flagged as erasures 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.

An important issue we have glossed over is that a uniformly random permutation of the n bits of the payload codeword would take $\Omega(n \log n)$ bits to specify. This would make the control information too big compared to the message length; we need it to be a tiny fraction of the message length. We therefore 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 [Smith 2007].

Extending the Construction to Poly-Time Channels. The construction for additive channels does not work against more powerful channels for (at least) two reasons:

(i) A more powerful channel may inject a large number of correctly formatted control blocks into the transmitted word (recall, each of the blocks is quite small). Even if the real control blocks are uncorrupted, the decoder will have trouble determining which of the correct-looking control blocks is in fact legitimate.
(ii) Since the channel can decide the errors after seeing the codeword, it may be able to learn which blocks of the codeword contain the control information and concentrate errors on those blocks. Similarly, we have to ensure that the channel does not learn about the permutation used to scramble the payload codeword and thus cause a bad error pattern that cannot be decoded by the standard decoder for the concatenated code.

The first obstacle is the easier one to get around, and we do so by using list decoding: Although the channel may inject spurious possibilities for the control information, the total number of such spurious candidates will be bounded. This ensures that after list decoding, provided at least a small fraction of the true control blocks do not incur too many errors, the list of candidates will include the correct control information with high probability.

To overcome the second obstacle, we make sure, using appropriate pseudorandom generators and employing a "hybrid" argument, that the encoding of the message is *indistinguishable from a random string* by a channel limited to a given time bound T, even when the channel has knowledge of the message and certain parts of the control information. One then uses this to ensure that the *distribution of errors* caused by a time-T channel on the codeword is indistinguishable in polynomial time from the distribution caused by the same channel on a uniformly random string. The latter distribution is independent of the codeword. If these error distributions were in fact *statistically close* (and not just close w.r.t. time-T tests), then successful decoding under oblivious errors would also imply successful decoding under the error distribution caused by the time-bounded channel. To show that closeness w.r.t. time-T tests does indeed suffice, we need to consider each of the conditions for correct decoding separately.

The condition that enough control blocks have at most a fraction $p + \varepsilon$ of errors can be checked in polynomial time given nonuniform advice, namely the locations of the control blocks. We use this together with the above indistinguishability to prove that enough control blocks are correctly list decoded, and thus the correct control information is among the candidates obtained by list decoding the control code.

The next step is to show that the payload codeword is correctly decoded given knowledge of the correct control information. The idea is that there is a set of error patterns such that (1) membership in the set can be checked in linear time, (2) the set has high probability under any oblivious error distribution, and (3) any error pattern in the set is correctly decoded with high probability by the concatenated code. Given these properties, one can show that if the concatenated code errs with noticeable probability on the actual error distribution, one can build a low-complexity distinguisher for the error distributions, thus contradicting their computational indistinguishability.

Remark 6.1. An earlier version of this article had a more complicated argument for online logspace channels, replacing one of the random components of the construction with Nisan's explicit pseudorandom generator for logspace. Nisan's generator only ensures that the error distribution caused by the channel is indistinguishable from oblivious errors by *online* space-bounded machines. Therefore, in the above argument, in order to arrive at a contradiction, we need to build a distinguisher than runs in online logspace. However, the unscrambling of the error vector (according to the permutation that was applied to the payload codeword) cannot be done in an online fashion. So we had to resort to an indirect argument based on showing limited independence of certain events related to the payload decoding. As pointed out by a reviewer, since the final construction is anyway randomized, in the current version, we simply use a random construction of pseudorandom generators for polynomial size circuits to build codes resilient to polynomial-time-bounded channels (and hence also logspace channels).

7. EXPLICIT CODES OF OPTIMAL RATE FOR ADDITIVE ERRORS

This section describes our construction of codes for additive errors.

7.1. 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 decodable with probability $1 - c_1/N$ from a fraction $p + O(\varepsilon)$ of *additive* errors decodable with probability $1 - c_1/N$. These codes are obtained via Theorem 4.1 (see Proposition A.1 in the appendix).
—A rate $O(\varepsilon)$ Reed-Solomon code RS that encodes a message as the evaluation of a polynomial at points $\alpha_1, \ldots, \alpha_\ell$ 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 [2004].
—A generator $\mathsf{KNR} : \{0,1\}^\sigma \to S_n$ for an (almost) t-wise-independent family of permutations of the set $\{1, \ldots, n\}$, that uses a seed of $\sigma = O(t \log n)$ random bits (Kaplan et al. [2006]).
—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 REC-DECODE$(\mathsf{REC}(m) + \pi(e)) = m$ 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 (see, *for example*, Smith [2007]).

7.2. Analysis

The following (Theorem 4.2, restated) is our result on explicit construction of capacity-achieving codes for additive errors.

THEOREM 7.1. *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 $R = 1 - H(p) - \varepsilon$ such that for every $m \in \{0,1\}^{RN}$ and error vector $e \in \{0,1\}^N$ of Hamming weight at most pN, we have $\Pr_\omega[\text{DECODE}(\text{ENCODE}(m;\omega) + e) = m] \geqslant 1 - \exp(-\Omega(\varepsilon^2 N/\log^2 N))$, where N is the block length of the code.*

With all the ingredients described in Section A in place, we can describe and analyze the code of Theorem 7.1. The encoding algorithm is given in Algorithm 1. The corresponding decoder is given in Algorithm 2. 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 6 when reading the formal description.

Starting the Proof of Theorem 7.1. First, note that the rate R of the overall code approaches the Shannon bound: R 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 $O(\varepsilon N)$. 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 Enc on m and coins ω chosen *independently* of the pair m, e, and let $x = \mathsf{Enc}(m;\omega) + e$. The decoder parses x into blocks $x_1, \ldots, x_{n'+\ell}$ of length $\Lambda \log N$, corresponding to the blocks output by the encoder.

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 is the expansion of the stochastic code from Theorem 4.1 that can correct a fraction $p + \varepsilon$ of errors.
$n \leftarrow \frac{N}{\Lambda \log N}$ The final codeword consists of n *blocks* of length $\Lambda \log N$.
$\ell \leftarrow 24\varepsilon N / \log N$ The control codeword is ℓ blocks long.
$n' \leftarrow n - \ell$ and $N' \leftarrow n' \cdot (\Lambda \log N)$ ▷ The payload codeword is n' blocks long (i.e., N' bits).

Phase 1: Generate control information

2: **Select seeds** s_π, s_Δ, s_V uniformly in $\{0,1\}^{\varepsilon^2 N}$.

3: $\omega \leftarrow (s_\pi, s_\Delta, s_V)$ Total length $|\omega| = 3\varepsilon^2 N$.

Phase 2: Encode control information

4: **Encode** ω **with a Reed-Solomon code** RS to get symbols $(a_1, \ldots, a_\ell)$.
▷ RS is a rate $\frac{\varepsilon}{8}$ Reed-Solomon code of length $24\varepsilon N = \frac{8}{\varepsilon} \cdot |\omega|$ bits which evaluates polynomials at points $(\alpha_1, \ldots, \alpha_\ell)$ in a field $\mathbb{F}$ of size $\approx N$.

5: **Encode each symbol together with its evaluation point**: For $i = 1, \ldots, \ell$, do

— $A_i \leftarrow (\alpha_i, a_i)$We add location information to each RS symbol to handle insertions and deletions.
— $C_i \leftarrow \mathsf{SC}(A_i, r_i)$, where r_i is random of length $2 \log N$ bits.
$\mathsf{SC} = \mathsf{SC}_{2\log N, p+\varepsilon} : \{0,1\}^{2\log N} \times \{0,1\}^{2\log N} \to \{0,1\}^{\Lambda \log N}$ is a stochastic code that can correct a fraction $(p + \varepsilon)$ of additive errors with probability $1 - c_1/N^2 > 1 - 1/N$ as per Proposition A.1.

Phase 3: Generate the payload codeword

6: **Encode** m **using a code that corrects *random* errors**:

— $P \leftarrow \mathsf{REC}(m)$, $\mathsf{REC} : \{0,1\}^{R'N'} \to \{0,1\}^{N'}$ is a code that corrects a $p + 25\Lambda\varepsilon$ fraction of *t-wise-independent* errors, as per Proposition A.9 . Here $R' = \frac{RN}{N'}$.

7: **Expand the seeds** s_V, s_Δ, s_π to get: ▷ See Section 7.1 for summary of subroutines.

— a set $V = \mathsf{Samp}(s_V) \subseteq [n]$ of size ℓ,
▷ Samp is a expander-based sampler (Proposition A.4)
— offset $\Delta = \mathsf{POLY}(s_\Delta) \in \{0,1\}^{N'}$,
▷ POLY is an almost t-wise-independent bit generator (Proposition A.8)
— permutation $\pi = \mathsf{KNR}(s_\pi)$ from $[N']$ to $[N']$.
▷ KNR is an almost t-wise-independent permutation generator (Proposition A.6)

8: **Scramble the payload codeword:**

— $\pi^{-1}(P) \leftarrow$ (bits of P permuted according to π^{-1})
— $Q \leftarrow \pi^{-1}(P) \oplus \Delta$
— Cut Q into n' blocks $B_1, \ldots, B_{n'}$ of length $\Lambda \log N$ bits.

Phase 4: Interleave blocks of payload codeword and control codeword

9: **Interleave** control blocks $C_1, \ldots, C_\ell$ with payload blocks $B_1, \ldots, B_{n'}$, using control blocks in positions from V and payload blocks in remaining positions.

Algorithm 2. DECODE: On Input x of Length N:

1: Cut x into $n' + \ell$ blocks $x_1, \ldots, x_{n'+\ell}$ of length $\Lambda \log(n)$ each.

2: **Attempt to decode control blocks**: For $i = 1, \ldots, n' + \ell$, do

— $\tilde{F}_i \leftarrow$ SC-DECODE(x_i).
With high prob, non-control blocks are rejected (Lemma 7.5), and control blocks are either correctly decoded or discarded (Lemma 7.4).

— If $\tilde{F}_i \neq \perp$, then parse $\tilde{F}_i$ as $(\tilde{\alpha}_i, \tilde{a}_i)$, where $\tilde{\alpha}_i, \tilde{a}_i \in \mathbb{F}$.

3: $(\tilde{s}_V, \tilde{s}_\Delta, \tilde{s}_\pi) \leftarrow$ RS-DECODE(pairs $(\tilde{\alpha}_i, \tilde{a}_i)$ output above).
Control information is recovered w.h.p. (Lemma 7.6).

4: Expand the seeds $\tilde{s}_V, \tilde{s}_\Delta, \tilde{s}_\pi$ to get set $\tilde{V}$, offset $\tilde{\Delta}$, and permutation $\tilde{\pi}$.

5: $\tilde{Q} \leftarrow$ concatenation of blocks x_i not in $\tilde{V}$
Fraction of errors in $\tilde{Q}$ is at most $p + O(\varepsilon)$.

6: $\tilde{P} \leftarrow \pi(\tilde{Q} \oplus \tilde{\Delta})$If control info is correct, then errors in $\tilde{P}$ are almost t-wise independent.

7: $\tilde{m} \leftarrow$ REC-DECODE$(\tilde{P})$Run the decoder from Proposition A.9.

The four lemmas below, proved in Section 7.3, show that the decoder recovers the control information correctly with high probability. We then show that the payload message is correctly recovered. The proof of the theorem is completed in Section 7.4.

The lemmas illuminate the roles of the main pseudorandom objects in the construction. First, the sampler seed is used to ensure that errors are not concentrated on the control blocks, as captured in the next lemma:

Definition 7.2 (*Good Sampler Seeds*). A sampled set V is *good* for error vector e if the fraction of control blocks with relative error rate at most $p + \varepsilon$ is at least $\frac{\varepsilon}{2}$.

LEMMA 7.3 (GOOD SAMPLER LEMMA). *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_V, the set V is good for e.*

Given a good sampler seed, the properties of the stochastic code SC guarantee that many control blocks are correctly interpreted. Specifically:

LEMMA 7.4 (CONTROL BLOCKS LEMMA). *For all e, V such that V is good for e, with probability at least $1 - \exp(-\Omega(\varepsilon^3 N/\log N))$ over the random coins $(r_1, r_2, \ldots, r_\ell)$ used by the ℓ SC encodings, we have the following: (i) The number of control blocks correctly decoded by* SC-DECODE *is at least $\frac{\varepsilon\ell}{4}$, and (ii) 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.)*

The offset Δ is then used to ensure that payload blocks are not mistaken for control blocks:

LEMMA 7.5 (PAYLOAD BLOCKS LEMMA). *For all m, e, s_V, s_π, with probability at least $1 - 2^{-\Omega(\varepsilon^2 N/\log^2 N))}$ over the offset seed s_Δ, the number of payload blocks incorrectly accepted as control blocks by* SC-DECODE *is less than $\frac{\varepsilon\ell}{24}$.*

The two previous lemmas imply that the Reed-Solomon decoder will, with high probability, be able to recover the control information. Specifically:

LEMMA 7.6 (CONTROL INFORMATION LEMMA). *For any m and e, with probability $1 - 2^{-\Omega(\varepsilon^2 N/\log^2 N)}$ over the choice of the control information and the coins of* SC*, the control information is correctly recovered, that is,* $(\tilde{s}_V, \tilde{s}_\Delta, \tilde{s}_\pi) = (s_V, s_\Delta, s_\pi)$.

Remark 7.7. It would be interesting to achieve an error probability of $2^{-\Omega_\varepsilon(N)}$, that is, a positive "error exponent," in Theorem 7.1 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 [1988b] achieves a positive error exponent for all rates less than capacity, as does our existence proof using list decoding in Section 5.2.

Remark 7.8. A slight modification of our construction give codes for the "average error criterion," in which the code is deterministic but the message is assumed to be unknown to the channel and the goal is to ensure that for every error vector most messages are correctly decoded; see Theorem B.4 in Appendix B.

7.3. Proofs of Lemmas Used in Theorem 7.1

PROOF OF LEMMA 7.3. 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_V is chosen independently of the set B. Thus, the fraction of control blocks in B will be roughly ε. Specifically, we can apply Proposition A.4 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.) □

POOOF OF LEMMA 7.4. Fix e and the sampled set V 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 V, but it is *independent* of the randomness used by SC to generate C_j. Given this observation, we can apply Proposition A.1 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 7.5. Consider a block x_i that corresponds to payload block j, that is, $x_i = B_j + e_i$. Fix e, s_V, 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 A.1 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 [1994, 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 7.6 Suppose the events of Lemmas 7.4 and 7.5 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 A.2, then 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 7.4 and 7.5, 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. □

7.4. Completing the Proof of Main Theorem 7.1

Proof of Theorem 7.1. We will first prove that the decoding of the payload codeword succeeds assuming the correct control information $\omega = (s_\pi, s_\Delta, s_V)$ is handed directly to the decoder, that is, in the "shared randomness" setting. We will then account for the fact that we must condition on the correct recovery of the control information ω by the first stage of the decoder.

Fix a message m, error vector e, and sampler seed s_V, and let e_Q be the restriction of e to the payload codeword, that is, blocks not in V. 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 V, 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 that are t-wise independent, in the sense of Proposition A.9 [Smith 2007]. 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 decoding, the control information is not handed directly to the decoder. Let $\tilde{\omega}$ be the candidate control information recovered by the decoder (in Step 8 of the algorithm). The above suite of lemmas (Lemmas 7.3, 7.4, 7.5, and 7.6) show that the control information is correctly recovered, that is, $\tilde{\omega} = \omega$, with probability at least $\exp(-\Omega(\varepsilon^2 N/\log^2 N)))$.

The overall probability of success is given by

$$\Pr_{\omega}[\text{payload decoding succeeds with control information } \tilde{\omega}],$$

which is at least

$$\begin{aligned}
&\Pr_{\omega}[\tilde{\omega} = \omega \wedge \text{payload decoding succeeds with control information } \tilde{\omega}] \\
&= \Pr_{\omega}[\tilde{\omega} = \omega \wedge \text{payload decoding succeeds with control information } \omega] \\
&\geqslant 1 - \Pr_{\omega}[\tilde{\omega} \neq \omega] - \Pr_{\omega}[\text{payload decoding succeeds given } \omega] \\
&\geqslant 1 - \exp(-\Omega(\varepsilon^2 N/\log^2 N)) - \exp(-\Omega(\varepsilon^4 N/\log N)).
\end{aligned}$$

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 7.1. □

8. CAPACITY-ACHIEVING CODES FOR TIME-BOUNDED CHANNELS

In this section, we outline a Monte Carlo algorithm that, for any desired error fraction $p \in (0, 1/2)$, produces a code of rate close to $1 - H(p)$ which can be efficiently *list decoded* from errors caused by an arbitrary randomized polynomial-time channel that corrupts at most a fraction p of symbols with high probability. Recall that for $p > 1/4$, resorting to list decoding is necessary even for very simple (constant space) channels (Theorem 4.3).

We will use the same high-level approach from our construction for the additive errors case, with some components changed. The main difference is that the codeword will be pseudorandom to bounded distinguishers, allowing us to "reduce" to the case of oblivious errors. (In fact, we will show that the errors are *indistinguishable from* oblivious errors, which will turn out to suffice). We will make repeated use of the following definition:

Definition 8.1 (*Indistinguishability*). For a given (possibly randomized) Boolean function $\mathcal{A}$ on some domain D and two random variables X, Y taking values in D, we write $X \overset{\eta}{\approx}_{\mathcal{A}} Y$ if

$$|\Pr(\mathcal{A}(X) = 1) - \Pr(\mathcal{A}(Y) = 1)| \leqslant \eta.$$

We write $X \overset{\eta}{\approx}_{\text{time T}} Y$ to indicate that $X \overset{\eta}{\approx}_{\mathcal{A}} Y$ for all circuits of size at most T.

Definition 8.2 (*Pseudorandom Generators*). A map $G : \{0,1\}^s \to \{0,1\}^n$ is said to be $(\mathsf{T}, \varepsilon)$-pseudorandom if

$$G(U_s) \overset{\varepsilon}{\approx}_{\text{time T}} U_b,$$

where U_m denotes the uniform distribution on $\{0,1\}^m$.

We begin the section with an overview of the code construction. We then develop several key technical results: a construction of small "pseudorandom codes," a useful intermediate result, the Hiding Lemma, and, finally, we show how to use these to analyze the decoder's performance.

8.1. Code Construction and Ingredients

Parameters. Input parameters of the construction: N, p, S, ε, where

(1) N is the block length of the final code,
(2) pN is the bound on the number of errors introduced ($0 < p < 1/2$) by the channel w.h.p.
(3) T_0 is a bound on the circuit size (think "running time") of the channel. We will eventually set $\mathsf{T}_0 = N^c$ for a constant $c > 1$, though most results hold for any $\mathsf{T}_0 > N$.
(4) ε is a measure of how far the rate is from the optimal bound of $1 - H(p)$ (that is, the rate must be at least $1 - H(p) - \varepsilon$). We will assume $0 < 2\varepsilon < 1/2 - p$.

The seeds/control information. The control information ω consists of three randomly chosen strings s_π, s_V, s_Γ where s_π, s_V are as in the additive errors case. We take the lengths of s_π, s_V be ζN where $\zeta = \zeta(p, \varepsilon)$ will be chosen small enough compared to ε.

The third string $s_\Gamma \in \{0,1\}^{\gamma N}$ is the seed of a pseudorandom generator Gen that outputs N bits and fools circuits of size (roughly) T_0. (A shorter seed would suffice here, but we make all seeds the same length for simplicity—see below.) The offset $\Gamma \leftarrow \mathsf{Gen}(s_\Gamma)$ will be used to fool the polynomial-time channel. We do not need to add the t'-wise-independent offset Δ as we did in the additive errors case.

Encoding the message. The payload codeword encoding the message m will be $\pi^{-1}(\mathsf{REC}(m)) \oplus \Gamma$, which is the same as the encoding for the additive channel, with the offset Γ added to break dependencies in the time-bounded channel instead of the t'-wise-independent offset Δ.

The code REC is the same as the code for the case of additive errors. We will denote by β_{REC} the maximum, over fixed error patterns e, of the probability, over permutations π, that REC does not correctly decode the pattern $\pi(e)$. As noted in Section 7.1, $\beta_{\mathsf{REC}} \leqslant 2^{-\Omega(\varepsilon^2 \zeta N/\log N)}$. We will need the following additional property of REC: There is a circuit of size $O_\varepsilon(N)$ that takes as input an error pattern e, permutation π, and set of control positions V and checks whether or not REC will decode the error pattern $\pi(e)$ (restricted to positions outside of V) correctly. (This circuit works by counting the number of blocks of the concatenated code which the inner code decodes incorrectly.)

For the offset Γ, we need a pseudorandom generator $\mathsf{PolyPRG}$ that is computable in time poly(N) and secure against circuits of size T with polynomially small error, with seed length at most γN. Such generators can be constructed in a Monte Carlo fashion (a random function from $c \log N$ to N bits will do for a large-enough constant c):

Proposition 8.3 (Folklore (Also Follows From Proposition 8.4)). *For every constant $\zeta > 0$ and polynomial $\mathsf{T}(N) \geqslant N$, for sufficiently large N there exists a poly-time Monte Carlo construction of a polynomial-time computable function* $\mathsf{PolyPRG}$ *from ζN to N bits that is* $(\mathsf{T}, \frac{1}{\mathsf{T}})$ *pseudorandom with probability at least* $1 - 1/\mathsf{T}$.

This construction can be made explicit assuming either that one-way functions exist [Yao 1982; Håstad et al. 1999] or that $\mathsf{E} \not\subseteq \mathsf{SIZE}(2^{\varepsilon_0 n})$ for some absolute constant $\varepsilon_0 > 0$ [Impagliazzo and Wigderson 1997] (where $\mathsf{E} = \mathrm{TIME}(2^{O(n)})$ and $\mathsf{SIZE}(2^{\varepsilon_0 n})$ denotes the class of languages that have size $O(2^{\varepsilon_0 n})$ circuits). For *space-bounded* (as opposed to time-bounded) distinguishers, there is even an explicit construction that makes no assumptions [Nisan 1992]. However, as noted by one reviewer, we require

a Monte Carlo algorithm to construct pseudorandom codes (as required by Proposition 8.4 below), even for space-bounded channels. Therefore, we use a Monte Carlo construction of PolyPRG and get a single statement covering time- and space-bounded channels.

Encoding the seeds. The control information (consisting of the seeds s_π, s_V, s_Γ) will be encoded by a similar structure to the solution for the additive channel: a Reed-Solomon code of rate $R^{\mathsf{RS}} = R^{\mathsf{RS}}(p, \varepsilon)$ concatenated with an inner stochastic code. But the stochastic code SC (of Proposition A.1) will now be replaced by a *pseudorandom code* PRC that satisfies two requirements: First, it has good list decoding properties and, second, the (stochastic) encoding of any given message is indistinguishable from a random string by a randomized time-bounded channel. The construction of the necessary stochastic code is guaranteed by the following lemma.

PROPOSITION 8.4 (INNER CONTROL CODES EXIST). *For some fixed positive integer Λ_0 the following holds. For all δ, $0 < \delta < 1/2$ and polynomials $\mathsf{T} = \mathsf{T}(N) \geqslant N$, for sufficiently large N there exist $R = R(\delta) \geqslant (1/2 - \delta)^{\Omega(1)} > 0$ and a positive integer $L = L(\delta) \leqslant 1/(1/2-\delta)^{O(1)}$ such that there exists a* poly(T) *time-randomized Monte Carlo algorithm A that takes a random string of length poly(N, T) and outputs a stochastic code* (Enc, Dec) *with block length $b = \Lambda_0 \log(\mathsf{T})$ and rate $k/b \geqslant R$ that, with probability at least $1 - 2^{-b}$, satisfies:*

—(Enc, Dec) *is (δ, L)-list decodable: For every $y \in \{0,1\}^b$, there are at most L pairs (m, r) such that $E(m, r)$ is within Hamming distance δb of y;*
—(Enc, Dec) *is $(\mathsf{T}, 1/\mathsf{T})$-pseudorandom: For every $m \in \{0,1\}^k$, we have $E(m, U_s) \overset{1/\mathsf{T}}{\approx}_{time\ \mathsf{T}} U_b$; and*
—*there exists a deterministic decoding procedure running in time* poly(T) *that, given a string $y \in \{0,1\}^b$, recovers the complete list of at most L pairs (m, r) whose encodings $E(m, r)$ are within Hamming distance at most δb from y.*

Proposition 8.4 is proved in Section 8.3. An interesting direction for future work is the design of *explicit* pseudorandom stochastic codes along the lines of Proposition 8.4. See Section 9.

Full Code. To construct the final code, we will apply the Monte Carlo constructions of the previous section with time $\mathsf{T}_2 = \mathsf{T}_0 + O(N \max\{\log N, 2^{\mathrm{poly}(1/\varepsilon)}\})$ (the exact value of T_2 will be clear from the analysis). For constant ε, it suffices to take $\mathsf{T}_2 = 2T_0$ when $\mathsf{T}_0$is a large-enough polynomial in N. This means the control blocks have length $\Lambda_0 \log T_2 = \Theta(\log T_0)$. The control blocks occupy an ε fraction of the whole codeword, which means the number of real control blocks is $n_{\mathsf{ctrl}} = \Theta(\frac{\varepsilon N}{\log T_0})$.

As in the additive errors case, the control blocks will be interspersed with the payload blocks at locations specified by the sampler's output on s_V.

Rate of the code. The code encoding the control information is of some small constant rate $R_{\mathsf{ctrl}}(p, \varepsilon)$, but the control information consists only of $O(\zeta N)$ bits. Given ε, we can select ζ small enough so the control portion of the codeword only adds $\varepsilon N/2$ bits to the overall encoding. The rate of REC is at least $(1-\varepsilon/10)(1-H(p)-\varepsilon/10) \geqslant 1-H(p)-\varepsilon/5$. So the rate of the overall stochastic code is at least $1 - H(p) - \varepsilon$ as desired.

8.2. List Decoding Algorithm for Full Encoding

The decoding algorithm for the full encoding will be similar to the additive case with the principal difference being that the inner stochastic codes will be decoded using the procedure guaranteed in Proposition 8.4. For each block, we obtain a list of L possible

pairs of the form (α_i, a_i). This set of (at most NL) pairs is the fed into the polynomial-time Reed-Solomon list-decoding algorithm (guaranteed by Proposition A.3), which returns a list of poly$(1/\varepsilon)$ values for the control information. This comprises the *first phase* of the decoder.

Once a list of control vectors is recovered, the *second phase* of the decoder will run the decoding algorithm for REC for each of these choices of the control information and recover a list of possible messages.

The steps to decode each of inner stochastic codes takes time poly(T) and decoding the Reed-Solomon code as well as REC takes time polynomial in N. So the overall runtime is polynomial in N and T.

THEOREM 8.5. *Let* $\mathsf{W}_{\mathsf{T}_0}$ *be an arbitrary randomized time-*T_0 *channel on* N *input bits that is* pN *bounded. Consider the code construction described in Section 8.1.*

The resulting code has rate at least $1 - H(p) - \varepsilon$. *For every message* m*, with high probability over the choice of control information* s_π, s_V, s_Γ*, the coins of* PRC*, and the coins of* $\mathsf{W}_{\mathsf{T}_0}$*, the list output by the decoding algorithm has size* poly$(1/\varepsilon)$ *and includes the message* m *with probability at least*

$$1 - O(N/\mathsf{T}_0).$$

The running time of the decoding algorithm is polynomial in N *and* T_0.

The novelty compared to the additive errors case is in the analysis of the decoder, which is more subtle since we have to deal with a much more powerful channel. The remainder of this section deals with this analysis, which will establish the validity of Theorem 8.5.

8.3. Monte Carlo Constructions of Pseudorandom Codes

POOOF OF PROPOSITION 8.4. The codes we design have a specific structure: A binary stochastic code with encoding map E, where $E : \{0,1\}^k \times \{0,1\}^s \to \{0,1\}^b$, is said to be *decomposable* if there exist functions $E_1 : \{0,1\}^k \to \{0,1\}^b$ and $E_2 : \{0,1\}^s \to \{0,1\}^b$ such that $E(x,y) = E_1(x) \oplus E_2(y)$ for every x, y. We say that such an encoding decomposes as $E = [E_1, E_2]$.

The existence will be shown by a probabilistic construction with a decomposable encoding $E(m,r) = C(m) \oplus \mathsf{BPPRG}(r)$, where C will be (the encoding map of) a linear list-decodable code, and BPPRG will be a generator that fools size T circuits, obtained by picking $\mathsf{BPPRG}(r) \in \{0,1\}^b$ independently and uniformly at random for each seed r (our analysis relies on more than just the pseudorandomness of BPPRG—we need its domain to also have certain error correction properties). Here $b = \Lambda_0 \log(\mathsf{T})$ for a large-enough absolute constant Λ_0 as in the statement of the Proposition. Note that the construction time is $2^{O(b)} = \text{poly}(\mathsf{T})$.

LIST-DECODING PROPERTY. We adapt the proof that a truly random set is list decodable. Let $C \subseteq \{0,1\}^b$ be a linear $(\delta, L_C = L_C(\delta))$-list-decodable code; such codes exist for rates less than $1 - H(\delta)$ [Guruswami et al. 2002] and can be constructed explicitly with positive rate $R(\delta) \geqslant (1/2 - \delta)^{\Omega(1)} > 0$ for any constant $\delta < 1/2$ with a list size $L_C \leqslant 1/(1/2 - \delta)^{O(1)}$ [Guruswami and Sudan 2000]. We will show that the composed code E has constant list size with high probability *over the choice of* BPPRG as long as the rate of the combined code is strictly less than $1 - H(\delta)$.

Fix a ball B' of radius δb in $\{0,1\}^b$, and let X denote the size of the intersection of the image of E with B'. We can view the image of E as a union of 2^s sub-codes C_r, where C_r is the translated code $C \oplus \mathsf{BPPRG}(r)$ (for $r \in \{0,1\}^s$). Each sub-code C_r is (δ, L_C)-list decodable since it is a translation of C. We can then write $X = \sum_{r \in \{0,1\}^s} X_r$, where X_r is the size

of $C_r \cap B'$. The X_r are independent integer-valued random variables with range $[0, L_C]$ and expectation $\mathbb{E}[X_r] = |C| \cdot |B'|/2^b \leqslant 2^{-b(1-H(\delta)-R_C)}$, where R_C denotes the rate of C.

If we set $s = 10 \log T_0$, then we get:

$$\mathbb{E}[X] = 2^{10 \log(\mathsf{T})} 2^{-b(1-H(\delta)-R_C)} = 2^{-b(1-H(\delta)-R_C-10/\Lambda_0)}.$$

Suppose $R_C + 10/\Lambda_0 = 1 - H(\delta) - \alpha_0$, so $\mathbb{E}[X] = 2^{-\alpha_0 b}$. Let t be the ratio $L/\mathbb{E}[X]$, where $L = L(\delta)$ is the desired list-decoding bound for the composed code E. We will set $L = 3L_C/\alpha_0$. By the multiplicative Chernoff bound for bounded random variables, the probability (over the choice of BPPRG) that $X > L$ is at most $(\frac{t}{e})^{t\mathbb{E}[X]/L_C}$. Simplifying, we get $\Pr[X > L] \leqslant (\frac{L}{e})^2 2^{-3b} \leqslant 2^{-2b}$.

Taking a union bound over all 2^b possible balls B', we get that with probability at least $1-2^{-2b}$, the random choice of BPPRG satisfies the property that the decomposable stochastic code with encoding map $E = C \oplus \mathsf{BPPRG}$ is (δ, L)-list decodable.

Pseudorandomness. The proof of pseudorandomness is standard, but we include it here for completeness. It suffices to prove the pseudorandomness property against all deterministic circuits of size T, since a randomized circuit is just a distribution over deterministic ones.

Fix an arbitrary codeword $C(m)$. Consider the (multi)set $X_m = \{C(m) \oplus \mathsf{BPPRG}(r)\}$ as r varies over $\{0,1\}^s$. Each element of this set is chosen uniformly and independently at random from $\{0,1\}^b$. Fix a circuit B of size T. By a standard Chernoff bound, the probability, over the choice of X_m, that $\Pr_{x \in X_m}[B(x) = 1]$ deviates from the probability $\Pr[B(U_b) = 1]$ that B accepts a uniformly random string by more than ζ in absolute value is at most $\exp(-\Omega(\zeta^2 |X_m|))$. For $\zeta = 1/\mathsf{T}$ and $|X_m| = 2^s \geqslant \mathsf{T}^{10}$, this probability is at most $\exp(-\Omega(T_0^8))$.

The number of circuits of size T is $\exp(O(\mathsf{T} \log(\mathsf{T})))$. By a union bound over all these branching programs, we conclude that except with probability at most $\exp(-\operatorname{poly}(\mathsf{T}))$ over the choice of BPPRG, the following holds for *every* size-T circuit B: $|\Pr_{x \in X_m}[B(x) = 1] - \Pr[B(U_b) = 1]| \leqslant 1/\mathsf{T}$. Since m was arbitrary, we have proved that the constructed stochastic code is $(\mathsf{T}, 1/\mathsf{T})$-pseudorandom with probability at least $1 - 2^{-2b}$.

Decoding. Finally, it remains to justify the claim about the decoding procedure. Given a string $y \in \{0,1\}^b$, the decoding algorithm will go over all $(m, r) \in \{0,1\}^k \times \{0,1\}^s$ by brute force and check for each whether $\operatorname{dist}(E(m,r), y) \leqslant \delta b$. By the list-decoding property, there will be at most L such pairs (m, r). The decoding complexity is $2^{O(k+s)} = 2^{O(b)}$. □

8.4. Analyzing Decoding: Main Steps

It will be convenient to explicitly name the different sources of error in the construction. The code ingredients are selected so each of these terms is at most $1/\mathsf{T}_0$.

$\beta_\Gamma(\mathsf{T})$	distinguishability of long generator (from uniform) by circuits of size T
$\beta_{\mathsf{PRC}}(\mathsf{T})$	distinguishability of PRC outputs by circuits of size T
β_V	max. probability (over choice of sampler set V) that a fixed error pattern will not be well-distributed among control blocks
β_π	max. probability that any fixed error pattern will cause a decoding error for code REC after π

Our analysis requires two main claims.

Lemma 8.6 (Few Control Candidates). *The decoder recovers a list of $L' \leqslant \operatorname{poly}(1/\varepsilon)$ candidate values of the control information. The list includes the correct value*

$\omega = (s_\pi, s_V, s_\Gamma)$ *of the control information used at the encoder with probability at least* $1 - \beta_{control}$*, where*

$$\beta_{control} \leqslant \beta_V + \beta_\Gamma(\mathsf{T}_2) + N \cdot \beta_{\mathsf{PRC}}(\mathsf{T}_2) \leqslant (N+3)/\mathsf{T}_0,$$

when $T_2 = T_0 + O(N \log N)$.

LEMMA 8.7 (PAYLOAD DECODING SUCCEEDS). *Given the correct control information* (π, V, Γ)*, the decoder recovers the correct message with high probability. Specifically, the probability of successful decoding is at least* $1 - \beta_{payload}$*, where*

$$\beta_{payload} \leqslant \beta_\pi + \beta_\Gamma(\mathsf{T}_2) + N \cdot \beta_{\mathsf{PRC}}(\mathsf{T}_2)) \leqslant (N+3)/\mathsf{T}_0$$

and $T_2 = T_0 + O(N 2^{\mathrm{poly}(1/\varepsilon)})$.

Combining these two lemmas, which we prove in the next two sections, we get that except with probability at most $\beta_{Control} + \beta_{payload} \leqslant \beta_V + \beta_\pi + 2\beta_\Gamma(\mathsf{T}_2) + 2N \cdot \beta_{\mathsf{PRC}}(\mathsf{T}_2) \leqslant 2(N+3)/\mathsf{T}_0$, the decoder recovers a list of at most $L' \leqslant \mathrm{poly}(1/\varepsilon)$ potential messages, one of which is the correct original message. This establishes Theorem 8.5.

8.5. The Hiding Lemma

Given a message m, and pseudorandom outputs π, V, Γ based on the seeds s_π, s_V, s_Γ, let

$$\mathsf{Enc}(m; \pi, V, \Gamma, r_1, \ldots, r_{n_{\mathsf{ctrl}}})$$

denote the output of the encoding algorithm when the r_i's are used as the random bits for the LSC encoding. Let $\mathsf{Enc}(m; \pi, V, \cdot)$ be a random encoding of the message m using a given π, V and selecting all other inputs at random.

LEMMA 8.8 (HIDING LEMMA). *For all messages* m*, sampler sets* V*, and permutations* π*, the random variable* $\mathsf{Enc}(m; \pi, V, \cdot)$ *is pseudorandom, namely:*

$$\mathsf{Enc}(m; \pi, V, \cdot) \overset{\beta}{\approx}_{time\ \mathsf{T}} U_N,$$

where $\beta \leqslant \beta_{Hide}(\mathsf{T}) \overset{def}{=\!\leqslant} N \cdot \beta_{\mathsf{PRC}}(\mathsf{T}') + \beta_\Gamma(\mathsf{T}')$ *and* $\mathsf{T}' = \mathsf{T} + N$.

For our purposes, the most important consequence of the Hiding Lemma that even with the knowledge of m, π, and V, the distribution of errors inflicted by a space-bounded channel on a codeword of our code and on a uniformly random string are indistinguishable by time-bounded tests.

In other words, the pseudorandomness of the codewords allows us to reduce the analysis of time-bounded errors to the additive case, as long as the events whose probability we seek to bound can be tested for by small circuits. We encapsulate this idea in the "Oblivious Errors Corollary" below. The proof of the Hiding Lemma follows that of the corollary.

Definition 8.9 (*Error Distribution*). Given a randomized channel W on N bits and a random variable D on $\{0,1\}^N$, let $\mathcal{E}_{\mathsf{W}}(D) = D \oplus \mathsf{W}(D)$ denote the error introduced by W when D is sent through the channel.

COROLLARY 8.10 (ERRORS ARE NEAR-OBLIVIOUS). *For every* m, π, V *and randomized time-*T *channel* W*. The error distributions for a real codeword and a random string are indistinguishable. That is, for every time-bound* T'*:*

$$\mathcal{E}_{\mathsf{W}}(U_N) \overset{\beta}{\approx}_{time\ \mathsf{T}'} \mathcal{E}_{\mathsf{W}}(\mathsf{Enc}(m; \pi, V, \cdot)),$$

where $\beta \leqslant \beta_{Hide}(\mathsf{T} + \mathsf{T}' + N)$ *and* $\beta_{Hide}(\cdot)$ *is the bound from the Hiding Lemma (8.8). Note that the distinguishing circuits here may depend on* m, π, V.

PROOF OF COROLLARY. One can compose a distinguisher for the two error random variables with the channel W to get a distinguisher for the original distributions of the Hiding Lemma. This composition requires the addition of a layer of XOR gates (thus adding N to the size of the circuit). □

PROOF OF THE HIDING LEMMA (LEMMA 8.8). The proof proceeds by a standard hybrid argument. Fix m, π, V, and recall that $n_{\mathsf{ctrl}} = |V| < N$ is the number of control blocks. Let D_0 be the random variable $\mathsf{Enc}(m; \pi, V, \cdot)$, and $D_{n_{\mathsf{ctrl}}+1}$ be the uniform distribution over $\{0,1\}^N$. We define intermediate random variables $D_1, D_2, \ldots, D_{n_{\mathsf{ctrl}}}$: In D_i, the first i control blocks from D_0 are replaced by random strings. For a given time-T circuit $\mathcal{A}$, let $p_i = \Pr[\mathcal{A}(D_i) = 1]$.

Note that for $i \in \{1, \ldots, n_{\mathsf{ctrl}}\}$, D_{i-1} and D_i are distributed identically in all blocks except the ith-control block. The pseudorandomness of PRC implies that, conditioned on any particular fixed value of the positions outside the ith control block, the distributions D_{i-1} and D_i are indistinguishable up to error $\beta_{\mathsf{PRC}}(\mathsf{T}) \leqslant \beta_{\mathsf{PRC}}(\mathsf{T}')$ by circuits of size T. Averaging over the possible values of the other blocks, we get $|p_i - p_{i-1}| \leqslant \beta_{\mathsf{PRC}}(\mathsf{T}')$.

To compare $D_{n_{\mathsf{ctrl}}}$ and $D_{n_{\mathsf{ctrl}}+1} = U_N$, note that the two distributions would be identical if the offset Γ were replaced by a truly random string. Since the control blocks are now random (and, hence, carry no information about s_Γ), we use a distinguisher for $D_{n_{\mathsf{ctrl}}}$ and $D_{n_{\mathsf{ctrl}}+1}$ (of size T) to get a distinguisher between Γ and $U_{N'}$ (of size T') by XORing the challenge string with the REC encoding of the message, permuted according to π. Because m, π, V can be fixed, the encoding can be hardwired into the new distinguisher, leading to a size increase of only $N' < N$ XOR gates. Thus, $|p_{n_{\mathsf{ctrl}}+1} - p_{n_{\mathsf{ctrl}}}| \leqslant \beta_\Gamma(\mathsf{T}')$, where $\mathsf{T}' = \mathsf{T} + N$.

Combining these bounds, we get $|p_{n_{\mathsf{ctrl}}+1} - p_0| \leqslant N\beta_{\mathsf{PRC}}(\mathsf{T}') + \beta_\Gamma(\mathsf{T}')$, as desired. □

8.6. Control Candidates Analysis

Armed with the Hiding Lemma, we can show that the decoder can recover a small list of candidate control strings, one of which is correct with high probability (Lemma 8.6).

There were four main lemmas in the analysis of additive errors. The first (Lemma 7.3) stated that the sampler set V is *good* error pattern e for V with high probability. A version of this lemma holds also for time-bounded errors. Recall that V is good for e (Definition 7.2) if at least a fraction ε of the n_{ctrl} control blocks have an error rate (fraction of flipped bits) bounded above by $p + \varepsilon$.

LEMMA 8.11 (GOOD SAMPLERS: TIME-BOUNDED ANALOGUE TO LEMMA 7.3). *For every pN-bounded time-T_0 channel with $\mathsf{T}_0 > N$ and for every message m and permutation seed s_π, the set V is good for the error pattern e introduced by the channel with probability at least $1 - (\beta_V + 2\beta_\Gamma(\mathsf{T}_2) + 2N \cdot \beta_{\mathsf{PRC}}(\mathsf{T}_2)) \geqslant 1 - 2(N+3)/\mathsf{T}_0$ over the choice of sampler seed s_V, the seed s_Γ for the pseudorandom offset, the coins of the control encoding and the coins of the channel. Here $\mathsf{T}_2 = \mathsf{T}_0 + N \log N$.*

PROOF. The crucial observation here is that Oblivious Errors Lemma implies that the error pattern introduced by a bounded channel is almost independent of V from the point of view of any time-T test. That is, by a simple averaging argument, the Oblivious Errors Lemma implies that for every distribution on m, π, V, we have

$$(m, \pi, V, \mathcal{E}_{\mathsf{W}}(U_N)) \overset{\beta}{\approx}_{\text{time } \mathsf{T}'} (m, \pi, V, \mathcal{E}_{\mathsf{W}}(\mathsf{Enc}(m; \pi, V, \cdot))). \tag{1}$$

The properties of the sampler imply that the probability of getting a good (control set, error) pair in the left-hand distribution is at least $1 - \beta_V$.

Thus, all we really have to do is show that "goodness" of the control set V can be tested efficiently, given e and V. Testing for goodness only involves counting the number of

errors in each of the control blocks and tallying the number of control blocks with too high a fraction of errors. This can be done in circuit size $O(N \log N)$ (in fact, quite a bit less but the optimization does not matter here).

By the Oblivious Errors Lemma, the probability of a good (control set, error) pair on the right-hand side of (1) is at least $1 - \beta_V - \beta_{Hide}(\mathsf{T} + O(N \log N))$, as desired. □

We now turn to the second lemma in the analysis of the control information decoding (Lemma 7.4 for additive errors), which stated that when the sampled positions V are good for the error pattern e, one can correctly recover a large number of control blocks. This is no longer true in our setting, but we require only the following weaker statement.

LEMMA 8.12 (CORRECT CONTROL BLOCKS—LIST-DECODING VERSION). *For any fixed e and V such that V is good for e, the decoding algorithm for the inner codes* PRC *outputs a list of L symbols containing the correct symbol a_i for at least $\frac{\varepsilon n_{\mathsf{ctrl}}}{2} = \Theta(\frac{\varepsilon^2 N}{\log \mathsf{T}_0})$ control blocks.*

PROOF. The list-decoding radius of the PRC code is set to be $\delta > p + \varepsilon$, so all blocks with an error rate below $p + \varepsilon$ produce a valid list. □

The third lemma from the analysis of additive errors (Lemma 7.5), which previously stated that very few payload blocks are mistaken for control blocks, requires a significant change, because it is possible for the time-bounded channel to inject fake control blocks into the codeword (by changing a block to some pre-determined codeword of PRC). Therefore, we can only say that the total number of candidate control blocks is small.

LEMMA 8.13 (BOUNDING MISTAKEN CONTROL BLOCKS). *For every m, e, ω, the total number of candidate control symbols is at most $\frac{NL}{b_{\mathsf{ctrl}}} = \Theta(\frac{NL}{\log T_0})$.*

PROOF. Since each candidate control block has $b_{\mathsf{ctrl}} = \Theta(\log \mathsf{T}_0)$ bits, there are $\frac{N}{\log \mathsf{T}_0}$ blocks considered by the decoder. The list decoding of each such block yields at most L candidate control symbols. □

PROOF OF LEMMA 8.6. Given Lemmas 8.11, 8.12, and 8.13, we only need to ensure that the rate R^{RS} of the Reed-Solomon code used at the outer level to encode the control information is small enough so list decoding is possible according to Proposition A.3 as long as (1) the number of data pairs n is at most $\frac{NL}{b_{ctrl}}$ and (2) the number of agreements t is at least $\Theta(\frac{\varepsilon^2 N}{\log \mathsf{T}_0})$. The claimed list decoding is possible with rate $R^{\mathsf{RS}} = O(\varepsilon^4/L)$, and the list decoder will return at most $O(L/\varepsilon^2)$ candidates for the control information. Since the list-decoding radius δ of PRC was chosen to be $\delta = p + \varepsilon < 1/2 - \varepsilon$, we have $L \leqslant 1/\varepsilon^{O(1)}$ by the guarantee of Proposition 8.4, so the output list size is bounded by a polynomial in $1/\varepsilon$. This proves Lemma 8.6. □

8.7. Payload Decoding Analysis

We now show Lemma 8.7: Given a magical, correct copy of the control information, the payload blocks will correctly be decoded to recover the message m. The assumption of correct control information essentially places us in the *shared randomness* setting.

As with the analysis of the control block decoding, we use the fact that errors are nearly oblivious (Corollary 8.10) to argue that the events that we needed to happen with high probability for successful decoding against additive errors (where the errors were oblivious to the codeword) will also happen with good probability with a time-bounded channel.

PROOF OF LEMMA 8.7. Fix the message m and the choice V of the control block locations. Recall that the probability for an *oblivious* error distribution that $\pi(e)$ causes REC to decode m incorrectly is at most $\beta_\pi \leqslant 1/\mathsf{T}_0$.

Note that given a permutation π, control set V and an error pattern e, one can easily check if the payload code REC will correctly decode the error pattern (one could run the full decoder for REC; alternatively, one can simply check that a sufficiently large number of the inner code blocks are decoded correctly by the inner code). There is a circuit of size $O(N2^{\mathrm{poly}(1/\varepsilon)})$ that verifies if decoding will occur correctly.

We can thus apply the Oblivious Errors Corollary (8.10) with $\mathsf{T} = \mathsf{T}_0$ and $\mathsf{T}' = O(N2^{\mathrm{poly}(1/\varepsilon)})$ to show the following: For every fixed m, V, the probability that a time-T_0 channel W introduces errors that induce a decoding mistake (by REC) is at most

$$\beta_\pi + \beta_{Hide}(\mathsf{T}_2) = \beta_\pi + \beta_\Gamma(\mathsf{T}_2) + N\beta_{\mathsf{PRC}}(\mathsf{T}_2)\,,$$

where $\mathsf{T}_2 = \mathsf{T}_0 + O(N2^{\mathrm{poly}(1/\varepsilon)})$, as desired. □

9. OPEN QUESTIONS

The code constructions of the previous sections leave several open questions

Uniquely decodable codes beyond the GV bound. The codes we design for time-bounded channels are list decodable but not necessarily uniquely decodable. This is inherent to the current analysis, since even a very simple adversary may inject valid control blocks into the codeword, potentially causing the decoder to come up with several seemingly valid control strings. For $p \geqslant 1/4$, we know the limitation is inherent to *any* construction, because our lower bound describes an attack that can be carried out by a very simple attacker. However, for $p < 1/4$, it may still be possible to design codes that lead the decoder to a unique, correct codeword with high probability. Since the initial version of this article was published, Haviv and Langberg [2011] showed that random codes can tolerate causal errors slightly beyond the Gilbert-Varshamov bound (regardless of the channel's complexity) in the low-error regime (i.e., they can achieve a rate better than $1 - H(2p)$ for correcting a fraction p of online errors, for small p). Those codes are neither explicit nor decodable in polynomial time, however.

(More) Explicit Constructions for Time-Bounded Channels. Our design of time-bounded channels uses Monte Carlo constructions in two places: for the pseudorandom code PRC and the generator Γ. Constructions for the generator Γ are in fact known based on worst-case hardness assumptions for circuits (say, that exponential time does not have subexponential size circuits [Impagliazzo and Wigderson 1997]). An interesting direction for future work is the design of *explicit* pseudorandom stochastic codes along the lines of Proposition 8.4 under such hardness assumptions. This would make the entire code construction explicit (conditionally).

Explicit Constructions for Online Space-Bounded Channels. Similarly to time-bounded channels, one can define *online space-bounded* channels. An online space-S channel is a width-2^S branching program that makes a single in-order pass over the transmitted codeword and outputs one bit for every bit that is read. Such channels were first considered by Galil et al. [1995] and are special case of both time-bounded channels (since a space-S channel can be implemented by a time-$N \cdot 2^S$ circuit) and of *causal channels* [Dey et al. 2008; Langberg et al. 2009].

Another direction for future work is the construction of fully explicit codes for space-bounded channels, without hardness assumptions or Monte Carlo constructions. We considered online space-bounded channels in the initial version of this article. The analysis of the codes for such channels was complex, because the reductions needed to preserve logarithmic space. Our original analysis, however, shows that

one only needs to construct logarithmic-length, pseudorandom stochastic codes for logarithmic-space channels in order to get a full construction (as one can use Nisan's generator [Nisan 1992] for Γ). The lemmas required for obtaining the full construction from these pieces can be found in version 3 of the arXiv version of this article (http://arxiv.org/abs/1004.4017v3). One advantage of considering space-bounded channels is that one could potentially have codes whose running time is a fixed polynomial, independent of the specific logarithmic space bound. Recent inefficient constructions of codes for general online channels [Chen et al. 2015] give reason to hope that such explicit codes for online logspace are possible.

APPENDIXES

A. INGREDIENTS FOR CODE CONSTRUCTION FOR ADDITIVE ERRORS

In this section, we will describe the various ingredients that we will need in our construction of capacity achieving AVC codes, expanding on the brief mention of these from Section 7.1.

A.1. Constant Rate Codes for Average Error

By plugging in an appropriate explicit construction of list-decodable codes (with suboptimal rate) into Theorem 5.3, 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 5.3 (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 5.3.

PROPOSITION A.1. *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 list decodable over αN-bounded additive channels 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, there exists an absolute constant $c_3 = c_3(\alpha)$ such that 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 5.3. The claim about decoding a uniformly random input follows since the number of strings y that 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 $\mathbb{F}$, then 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

$$\mathsf{RS}_{\mathbb{F},S,n,k}(m) = (f(\alpha_1), f(\alpha_2), \ldots, f(\alpha_n)), \tag{2}$$

where $f(X) = m_0 + m_1 X + \cdots + m_{k-1}X^{k-1}$. The following is a classic result on unique decoding Reed-Solomon codes [Peterson 1960], stated as a noisy polynomial reconstruction algorithm.

PROPOSITION A.2 (UNIQUE DECODING OF RS CODES). *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 $|\{i : f(\alpha_i) = a_i\}| - |\{i : f(\alpha_i) \neq a_i)\}| > k$.*

We also state a list-decoding generalization (the version of Sudan [1997] suffices for our purposes) that will be used in our result for space-bounded channels.

PROPOSITION A.3 (LIST DECODING OF RS CODES [SUDAN 1997]). *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 integer $k < n$, finds the set $\mathcal{L}$ of all polynomials f of degree at most k, if any, that satisfy $f(\alpha_i) = a_i$ for at least t values of i as long as $t > \sqrt{2kn}$. Moreover, there are at most $\sqrt{2n/k}$ polynomials in the set $\mathcal{L}$.*

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 $V \subseteq [N]$ of size ℓ, the density of $B \cap V$ 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 Vadhan [2004].

PROPOSITION A.4. *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 A.5. 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 A.6 (KAPLAN ET AL. [2006]). *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 A.7. 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 Alon et al. [1986], but we can work with the weaker $O(t \log n)$ seed length.

PROPOSITION A.8. *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 [1966] constructed binary linear concatenated codes that achieve the capacity of the binary symmetric channel BSC_p. Smith [2007] 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 A.9. *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.

B. CAPACITY-ACHIEVING CODES FOR AVERAGE ERROR

The average error criterion is an extensively studied topic in the literature on arbitrarily varying channels; see the survey in Lapidoth and Narayan [1998] and the many references therein. Here we assume the message is unknown to the channel and the decoding error probability is taken over a uniformly random choice of the message and the noise of the channel. The following defines this notion for the special case of the

additive errors. The idea is that we want every error vector to be bad for only a small fraction of messages.

Definition B.1 (*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 $e \in \Sigma^n$ of weight at most pN, the following holds for at least a fraction $(1-\delta)$ of messages $m \in \mathcal{M}$: $D(\mathcal{E}(m) + e) = m$.

B.1. Codes for Average Error from Stochastic Codes for Additive Errors

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, then the rates of the codes in the two models are almost the same.

Observation B.2. A stochastic code SSC that is strongly list decodable over pN-bounded additive channels 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, then the rate of AVC is $(1 + \lambda)$ times the rate of SSC.

B.2. Explicit Capacity-Achieving Codes for Average Error

We would now like to apply Observation B.2 to the stochastic codes constructed in Section 7 and also construct explicit codes achieving capacity for the average error criterion. For this, we need to ensure that the decoder for the stochastic code can also recover all the random bits used at the encoding. We already showed (Lemma 7.6) that the random string ω comprising the control information is in fact correctly recovered w.h.p. However, there is no hope to recover all the random strings $r_1, r_2, \ldots, r_\ell$ used by the various SC encodings. This is because some of these control blocks could incur much more than a fraction $p + \varepsilon$ of errors (or, in fact, be totally corrupted).

Our idea is to use the *same* random string r for each of the ℓ encodings $\mathsf{SC}(A_i, r)$ in Step 5. Since each run of SC-Decode is correct with probability at least $1 - c_1/N^2$, by a union bound over all n blocks, we can claim that all the following events occur with probability at least $1 - c_1/N$ (over the choice of r):

> Among the control blocks, all of the at least $\varepsilon\ell/2$ control blocks with at most a fraction $p + \varepsilon$ of errors are decoded correctly, along with the random string r, by SC-Decode. Further, SC-Decode outputs $\perp$ on all the other control blocks. Thus the correct random string r gets at least $\varepsilon\ell/2$ "votes."

By Lemma 7.5, with probability at least $1 - \exp(-\Omega(\varepsilon^2 N/\log^2 N)))$ (over the choice of ω), the number of payload blocks that get accepted as control blocks is at most $\varepsilon\ell/24$. (Note that this lemma only used the t'-wise independence of the offset string Δ.)

The above facts imply that the control information ω is recovered correctly with probability at least $1 - O(1/N)$ over the choice of (ω, r) (this is the analog of Lemma 7.6). Also r is the unique string that will get at least $\varepsilon\ell/2$ votes from the various runs of SC-Decode. Therefore, it can be correctly identified (with probability at least $1 - O(1/N)$ over the choice of (ω, r)) after running SC-Decode on all the n blocks. We can thus conclude the following result on capacity-achieving codes for average error (Definition B.1).

Lemma B.3 (Polynomially Small Average Error). *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 $O(1/N)$, where N is the block length of the code.*

One can reduce the error probability in this theorem by using redundant, but t-wise-independent, values r_i for the control block encodings. Specifically, let $(r_1, \ldots, r_\ell)$ be a random codeword from a Reed-Solomon code of dimension $\varepsilon\ell/8$ (the simpler construction above corresponds to a majority code). Then the r_i values are, in particular, $\varepsilon\ell/8$-wise independent. One can modify the proof of Lemma 7.4 (which states that sufficiently many control blocks are recovered) to rely on only this limited independence. Under the same conditions that the control information is correctly recovered, there is enough information to recover the entire vector $r_1, \ldots, r_\ell$. We can thus prove the following.

THEOREM B.4 (EXPONENTIALLY SMALL AVERAGE ERROR). *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 $\exp(-\Omega_\varepsilon(N/\log^2 N))$, where N is the block length of the code.*

C. IMPOSSIBILITY RESULTS FOR BIT-FIXING CHANNELS WHEN $p > \frac{1}{4}$

We prove Theorem 4.3, which shows that even very simple channels prevent reliable communication if they can introduce a fraction errors strictly greater than 1/4. In particular, this result (a) separates the additive (i.e., oblivious) error model from bounded-space channels when $p > 1/4$ and (b) shows that some relaxation of correctness is necessary to handle space- and time-bounded channels when $p > 1/4$.

Our proof adapts the impossibility results of Ahlswede [1978] on arbitrarily varying channels. We present a self-contained proof for completeness. Readers familiar with the AVCs literature will recognize the idea of *symmetrizability* from Ahlswede [1978].

The Swapping Channel. We begin by considering a simple *swapping* channel, whose behavior is specified by a *state* vector $s = (s_1, \ldots, s_n) \in \{0,1\}^n$. On input a transmitted word $c = (c_1, \ldots, c_n) \in \{0,1\}^n$, the channel W_s outputs c_i in all positions where $c_i = s_i$ and a random bit in all positions where $c_i \neq s_i$. The bits selected randomly by the channel at different positions are independent.

There are several equivalent characterizations that help to understand the channel's behavior. First, we may view the channel as outputting either c_i or s_i, independently for each position,

$$\mathsf{W}_s(c)_i = \begin{cases} c_i & \text{if } c_i = s_i \\ U \leftarrow \{0,1\} & \text{if } c_i \neq s_i \end{cases} = \begin{cases} c_i & \text{with prob. } 1/2 \\ s_i & \text{with prob. } 1/2 \end{cases}.$$

This view of the channel makes it obvious that the output distribution is symmetric with respect to the inversion of c and s. That is,

$$\mathsf{W}_s(c) \text{ and } \mathsf{W}_c(s) \text{ are identically distributed.} \tag{3}$$

The key idea behind our lower bounds is that if s is itself a valid codeword, then the decoder cannot tell whether c was sent with state s or s was sent with state c. If c and s code different messages, then the decoder will make a mistake with probability at least 1/2.

Note that the expected number of errors introduced by the channel is half of the Hamming distance $\mathrm{dist}(c, s)$; specifically, the number of errors is distributed as $\mathrm{Binomial}(\mathrm{dist}(c, s), \frac{1}{2})$. As long as $\mathrm{dist}(c, s)$ is close to $n/2$, then the number of errors will be less than $n(\frac{1}{4} + \nu)$ with high probability.

Hard Channel Distributions. Given an stochastic encoder $\mathsf{Enc}(\cdot;\cdot)$, consider the following distribution on swapping channels: Pick a random codeword in the image of

Enc and use it as the state,

$$\mathsf{W}^{main}(c): \begin{cases} \text{Select } m', r' \text{ uniformly at random} \\ \text{Compute } s \leftarrow \mathsf{Enc}(m', r') \\ \text{Output } \mathsf{W}_s(c) \end{cases}.$$

LEMMA C.1. *Under the conditions of Theorem 4.3, for channel* W^{main}*:*

(*a*) *The probability of a decoding error on a random message is* $\frac{1}{2} - o(1)$.
(*b*) *The expected number of bits altered by* $\mathsf{W}^{(main)}$ *is at most* $n/4$.

PROOF. (a) We are interested in bounding the probability of a decoding error:

$$\Pr(\text{correct decoding}) = \Pr_{\substack{m,r \\ \text{channel coins}}}(\mathsf{Dec}(\mathsf{W}^{main}(\mathsf{Enc}(m,r))) = m)$$
$$= \Pr_{\substack{m,r,m',r' \\ \text{swapping coins}}}(\mathsf{Dec}(\mathsf{W}_{\mathsf{Enc}(m',r')}(\mathsf{Enc}(m,r))) = m).$$

Because of the symmetry of the swapping channel, the right-hand side is equal to the probability that the decoder outputs m', rather than m. This is a decoding error as long as m' differs from m. We assumed that the size of the message space grows with n, so the probability that $m = m'$ goes to 0 with n. We use "right" and "wrong" and shorthand for the events that decoding is correct and incorrect, respectively.

$$\Pr(\text{right}) = \Pr_{m,m'}(\text{decoder outputs } m') \leqslant \Pr(\text{wrong} \vee m = m') \leqslant \Pr(\text{wrong}) + o(1).$$

Thus, the probability of correct decoding is at most $\frac{1}{2} - o(1)$. This proves part (a) of the Lemma.

It remains to show that the expected number of bit corruptions is at most $n/4$. This follows directly from the following fact, which is essentially the Plotkin bound from coding theory:

CLAIM C.2. *If* (m, r) *is independent of and identically distributed to* (m', r')*, then the expectation of the distance* $\mathrm{dist}(\mathsf{Enc}(m, r), \mathsf{Enc}(m', r'))$ *is at most* $n/2$.

PROOF. By linearity of expectation, the expected Hamming distance is the sum, over positions i, of the probability that $\mathsf{Enc}(m, r)$ and $\mathsf{Enc}(m', r')$ disagree in the ith positions. The probability that two i.i.d. bits disagree is at most $\frac{1}{2}$, so the expected distance is at most $\frac{n}{2}$. □

Part (b) of the lemma follows since the expected number of errors introduced by the swapping channel is half of the Hamming distance between the transmitted word and the state vector.

Bounding the Number of Errors. To prove part (2) of Theorem 4.3, we will find a (nonuniform) channel with a hard bound on the number of bits it alters. In logarithmic space, it is easy for the channel to count the number of bits it has flipped so far and stop altering bits when a threshold has been exceeded. The difficult part is to show that such a channel will still cause a significant probability of decryption error.

As before, the channel will select m', r' at random and run the swapping channel W_s with state $s = \mathsf{Enc}(m', r')$. In addition, however, it will stop altering bits once the threshold of $n(\frac{1}{4} + \nu)$ bits have been exceeded.

Consider now the transmission of a random codeword $c = \mathsf{Enc}(m, r)$. Let G be the event that $\mathrm{dist}(c, s) \leqslant n(\frac{1}{2} + \nu)$. By a Markov bound, the probability of $\overline{G}$ is at most $\frac{1/2}{1/2+\nu}$, and so the probability of G is $1 - \Pr(\bar{G}) \geqslant \frac{2\nu}{1+2\nu} \geqslant \nu$. Conditioned on G, the number

of bits altered by W_s on input c is dominated by Binomial($n(\frac{1}{2}+\nu)$, $\frac{1}{2}$). The probability that the number of bits altered exceeds $n(\frac{1}{4}+\nu)$ is therefore at most $\exp(-\Omega(\nu^2 n))$.

On the other hand, conditioned on G there is a significant probability of a decoding error. To see why this is the case, first note that conditioned on G the error-bounded channel will simulate $\mathsf{W}_s(c)$ nearly perfectly. Moreover, the event G is symmetric in c and s, and so conditioning on G does not help to distinguish $\mathsf{W}_c(s)$ from $\mathsf{W}_s(c)$. By the same reasoning as in the previous proof,

$$\Pr(\text{incorrect decoding}|G) \geqslant \frac{1}{2} - o(1)\,.$$

Since G has probability at least ν, the channel causes a decoding error with probability at least $\frac{\nu}{2} - o(1)$, in expectation over the choice of s. Hence, there exists a specific string s^* for which the channel causes a decoding error with probability $\frac{\nu}{2} - o(1)$. This completes the proof of Theorem 4.3.

ACKNOWLEDGMENTS

We are extremely grateful to an anonymous referee for observations that simultaneously strengthened our results and simplified the analysis of our code constructions. Specifically, the referee pointed out that the separate constructions we had for space- and time-bounded channels could be unified to obtain a single, unconditional result covering both types of channels.

REFERENCES

Rudolf Ahlswede. 1978. Elimination of correlation in random codes for arbitrarily varying channels. *Z. Wahrscheinlichkeitstheor. Verw. Gebiete* 44 (1978), 159–175.

Noga Alon, László Babai, and Alon Itai. 1986. A fast and simple randomized parallel algorithm for the maximal independent set problem. *J. Algorith.* 7, 4 (1986), 567–583.

Erdal Arikan. 2009. Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels. *IEEE Trans. Inform. Theor.* 55, 7 (2009), 3051–3073.

Mihir Bellare and John Rompel. 1994. Randomness-efficient oblivious sampling. In *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*. 276–287.

Zitan Chen, Sidharth Jaggi, and Michael Langberg. 2015. A characterization of the capacity of online (causal) binary channels. In *Proceedings of ACM Symposium on Theory of Computing (STOC'15)*. 287–296. DOI: http://dx.doi.org/10.1145/2746539.2746591

Ronald Cramer, Yevgeniy Dodis, Serge Fehr, Carles Padró, and Daniel Wichs. 2008. Detection of algebraic manipulation with applications to robust secret sharing and fuzzy extractors. In *Proceedings of EUROCRYPT*. 471–488.

Imre Csiszár and Prakash Narayan. 1988a. Arbitrarily varying channels with constrained inputs and states. *IEEE Trans. Inform. Theor.* 34, 1 (1988), 27–34.

Imre Csiszár and Prakash Narayan. 1988b. The capacity of the arbitrarily varying channel revisited: Positivity, constraints. *IEEE Trans. Inform. Theor.* 34, 2 (1988), 181–193.

Imre Csiszár and Prakash Narayan. 1989. Capacity and decoding rules for classes of arbitrarily varying channels. *IEEE Trans. Inform. Theor.* 35, 4 (1989), 752–769.

Bikash Kumar Dey, Sidharth Jaggi, and Michael Langberg. 2008. Codes against online adversaries. *CoRR* abs/0811.2850 (2008).

Bikash Kumar Dey, Sidharth Jaggi, Michael Langberg, and Anand D. Sarwate. 2012. Improved upper bounds on the capacity of binary channels with causal adversaries. In *ISIT*. IEEE, 681–685.

Peter Elias. 1957. *List Decoding for Noisy Channels*. Technical Report 335. Research Lab. of Electronics, MIT.

Peter Elias. 1991. Error-correcting codes for list decoding. *IEEE Trans. Inform. Theor.* 37 (1991), 5–12.

G. David Forney. 1966. *Concatenated Codes*. MIT Press, Cambridge, MA.

Zvi Galil, Richard J. Lipton, Xiangdong Yu, and Moti Yung. 1995. Computational error-correcting codes achieve Shannon's bound explicitly. *Manuscript* (1995).

Venkatesan Guruswami. 2003. List decoding with side information. In *Proceedings of the Conference on Computational Complexity (CCC)*. 300–309.

V. Guruswami. 2007. Algorithmic results in list decoding. *Foundations and Trends in Theoretical Computer Science (FnT-TCS)* 2, 2 (Jan. 2007).

Venkatesan Guruswami, Johan Håstad, Madhu Sudan, and David Zuckerman. 2002. Combinatorial bounds for list decoding. *IEEE Trans. Inform. Theor.* 48, 5 (2002), 1021–1035.

V. Guruswami and M. Sudan. 2000. List decoding algorithms for certain concatenated codes. In *Symposium on Theory of Computing (STOC)*. 181–190.

Richard W. Hamming. 1950. Error detecting and error correcting codes. *Bell Syst. Techn. J.* 29 (April 1950), 147–160.

Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. 1999. A pseudorandom generator from any one-way function. *SIAM J. Comput.* 28, 4 (1999), 1364–1396.

Ishay Haviv and Michael Langberg. 2011. Beating the Gilbert-Varshamov bound for online channels. In *ISIT*, Alexander Kuleshov, Vladimir Blinovsky, and Anthony Ephremides (Eds.). IEEE, 1392–1396.

Brett Hemenway and Rafail Ostrovsky. 2008. Public-key locally-decodable codes. In *CRYPTO (Lecture Notes in Computer Science)*, David Wagner (Ed.), Vol. 5157. Springer, 126–143.

Russell Impagliazzo and Avi Wigderson. 1997. P = BPP if E requires exponential circuits: Derandomizing the XOR lemma. In *Proceedings of the Symposium on the Theory of Computing*. 220–229.

Eyal Kaplan, Moni Naor, and Omer Reingold. 2006. Derandomized constructions of k-wise (almost) independent permutations. *Electronic Colloquium on Computational Complexity TR06-002* (2006).

Michael Langberg. 2004. Private codes or succinct random codes that are (almost) perfect. In *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*. 325–334.

Michael Langberg. 2008. Oblivious communication channels and their capacity. *IEEE Trans. Inform. Theor.* 54, 1 (2008), 424–429.

Michael Langberg, Sidharth Jaggi, and Bikash Kumar Dey. 2009. Binary causal-adversary channels. *CoRR* abs/0901.1853 (2009).

Amos Lapidoth and P. Narayan. 1998. Reliable communication under channel uncertainty. *IEEE Trans. Inform. Theor.* 44, 6 (1998), 2148–2177.

Richard J. Lipton. 1994. A new approach to information theory. In *Proceedings of the Symposium on Theoretical Aspects of Computer Science (STACS)*. 699–708.

Silvio Micali, Chris Peikert, Madhu Sudan, and David A. Wilson. 2005. Optimal error correction against computationally bounded noise. In *Proceedings of the 2nd Theory of Cryptography Conference*. 1–16.

Noam Nisan. 1992. Pseudorandom generators for space-bounded computation. *Combinatorica* 12, 4 (1992), 449–461.

Rafail Ostrovsky, Omkant Pandey, and Amit Sahai. 2007. Private locally decodable codes. In *ICALP (Lecture Notes in Computer Science)*, Lars Arge, Christian Cachin, Tomasz Jurdzinski, and Andrzej Tarlecki (Eds.), Vol. 4596. Springer, 387–398.

W. Wesley Peterson. 1960. Encoding and error-correction procedures for Bose-Chaudhuri codes. *IEEE Trans. Inform. Theor.* 6 (1960), 459–470.

Claude E. Shannon. 1948. A mathematical theory of communication. *Bell Syst. Techn. J.* 27 (1948), 379–423, 623–656.

Adam Smith. 2007. Scrambling adversarial errors using few random bits, optimal information reconciliation, and better private codes. In *Proceedings of the Symposium on Discrete Algorithms*. 395–404.

Madhu Sudan. 1997. Decoding of Reed-Solomon codes beyond the error-correction bound. *J. Complex.* 13, 1 (1997), 180–193.

Salil P. Vadhan. 2004. Constructing locally computable extractors and cryptosystems in the bounded-storage model. *J. Cryptol.* 17, 1 (2004), 43–77.

John M. Wozencraft. 1958. List decoding. *Quarterly Progress Report, Research Lab. of Electronics, MIT* 48 (1958), 90–95.

Andrew Chi-Chih Yao. 1982. Theory and applications of trapdoor functions. In *Proceedings of the 23rd IEEE Symposium on Foundations of Computer Science*. 80–91.

Victor V. Zyablov and Mark S. Pinsker. 1981 (in Russian); pp. 236-240 (in English), 1982. List cascade decoding. *Prob. Inform. Transm.* 17, 4 (1981 (in Russian); pp. 236–240 (in English), 1982), 29–34.

Received February 2011; revised February 2016; accepted May 2016