Electronic Colloquium on Computational Complexity, Report No. 77 (2010)

Codes for Computationally Simple Channels: Explicit Constructions with Optimal Rate

Venkatesan Guruswami* Adam Smith†

April 2010

Abstract

In this paper, 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 which adds the errors can be described by a sufficiently "simple" circuit. Codes for such channel models are attractive since, like codes for traditional adversarial errors, they can handle channels whose true behavior is *unknown* or *varying* over time.

For three 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 poly-time encodable/decodable binary codes for *additive* (a.k.a. *oblivious*) channels that achieve the Shannon capacity $1 - H(p)$. These are channels which 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 particular transmitted word. Such channels capture binary symmetric errors and burst errors as special cases.

List-decoding for log-space channels. A *space-$S(n)$ bounded* channel reads and modifies the transmitted codeword as a stream, using at most $S(n)$ bits of workspace on transmissions of n bits. For constant S, this captures many models from the literature, including *discrete channels with finite memory* and *arbitrarily varying channels*. We give an efficient binary code with optimal rate (up to $1 - H(p)$) that recovers a short list containing the correct message with high probability for channels limited to *logarithmic* space.

List-decoding for poly-time channels. For any constant c, assuming the existence of pseudorandom generators, we give a similar list-decoding result for channels describable by circuits of size at most n^c. We are not aware of any channel models considered in the information theory literature (other than purely adversarial channels) which require more than nonuniform linear time to implement.

*Computer Science Department, Carnegie Mellon University, Pittsburgh, PA 15213. guruswami@cmu.edu. Research supported by a Packard Fellowship and NSF CCF 0953155.

†Computer Science & Engr. Dept, Pennsylvania State University, University Park, PA 16802. Supported by NSF grants TF-0747294 and TF-0729171. asmith@cse.psu.edu

ISSN 1433-8092

Contents

1 Introduction

For the binary symmetric channel BSC_p which 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. Moreover, concatenated codes approach this capacity and are efficiently decodable (Forney [11]). In contrast, for adversarial channels that can corrupt up to a fraction p of symbols in an *arbitrary* manner, the optimal rate is unknown, though it is known that the rate has to be much smaller than the Shannon capacity. In particular, for $p > 1/4$, the achievable rate over an adversarial channel is zero, while $1 - H(p)$ remains positive. Determining the best asymptotic rate for error fraction p (equivalently, minimum relative distance $2p$) remains an important open question in combinatorial coding theory.

Codes that tolerate adversarial errors are attractive because they can model channels whose true behavior is unknown or varies over time, including, for example, burst errors and echo. In contrast, codes tailored to any one of these models tend to fail when the model changes. For example, concatenated codes, which can transmit efficiently and reliably at the Shannon capacity with random errors, fail miserably in the presence of *burst* errors that occur in long runs.

In this paper, we consider several intermediate models of uncertain channels. Specifically, we consider *computationally bounded* channels, which can introduce an arbitrary set of errors as long as (a) the total number of errors is bounded by pn with high probability and (b) the process which adds the errors can be described by a sufficiently "simple" circuit. The idea 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. The models we study, or close variants, have been considered previously; see Section 2 for a discussion of related work.

For three classes of channels, we provide efficiently encodable and decodable codes of optimal rate $1 - H(p)$ 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.)

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. Our results are stated formally in Section 3.

1.1 Our results

The encoders we construct are *stochastic* (that is, randomized). Probabilities are taken over the (local, unknown to the decoder) coins of the encoder and the choices of the channel; messages may be chosen adversarially and known to the channel. Our results assume *no* setup or shared randomness between the encoder and decoder.

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 which add an arbitrary error vector $e \in \{0,1\}^n$ of weight at most pn to the transmitted word. The error vector may depend on

the code 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 [5, 21]; see the discussion of related work below.

List decoding for space-bounded channels. The additive errors model postulates that the error vector has to be picked obliviously, before seeing the codeword $\mathsf{Enc}(m;r)$. To get a more powerful class of channels, consider a channel that processes the codeword as a stream, deciding as it goes which positions to corrupt. The channel's only limitation is a bound $S(n)$ on the amount of work space it can use. Roughly, we view the channel as a finite automaton with $2^{S(n)}$ states. More precisely, in order to allow *nonuniform* dependency on the code, we model the channel as a width-$2^{S(n)}$ branching program that outputs one bit for every input bit that it reads. Even for constant space S, this model captures a wide range of channels considered in coding theory, including additive channels, discrete channels with finite memory, echo, bounded delay and *arbitrarily varying channels* (see the discussion of related work, below, for definitions). We consider logarithmic space channels, since they constitute a more robust and expressive class than constant space channels (for example, a log-space channel can track the number of bits it has altered). As above, we assume that, with high probability, the channel introduces at most pn errors.

First, we show that reliable *unique* decoding with positive rate is impossible even for *memoryless* channels when $p > 1/4$. The issue is that a single code must work for all channels. 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 [34, 10]. However, constructing efficient (*i.e.*, polynomial-time encodable and decodable) codes for list decoding with near-optimal rate is a major open problem.

Our main contribution for space-bounded channels is a construction of polynomial-time list-decodable codes with optimal rate for channels whose space bound is logarithmic in the block length n. Specifically, for every message m and log-space 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.

Note that the decoder need not return *all* words within a distance pn of the received word (as is the case for the standard "combinatorial" meaning of list decoding for deterministic codes), but it must return the correct message as one of the candidates with high probability. From a communication viewpoint, this notion of list-decoding is natural for stochastic codes. In fact, it is exactly the notion that is needed in constructions which "sieve" the list, such as [13, 24]; see related work in Section 2.

Our results raise a compelling question: are there stochastic codes of rate approaching $1 - H(p)$ that can be uniquely decoded from pn log-space errors, when $p < 1/4$?

List decoding for polynomial time channels. More generally, one may consider channels whose behavior on n-bit inputs is described a circuit of size $T(n)$. Logarithmic space channels, in particular, can be realized by polynomial-size circuits. In fact, 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. Our construction of list-decodable codes for logarithmic space channels can be extended to handle channels with a given polynomial time bound $T(n) = n^c$, for any fixed $c > 1$, under an additional assumption, namely the existence of pseudorandom generators of constant stretch that output n pseudorandom bits and fool circuits of size n^c. Such generators exist, for example, if there are functions in E which have no subexponential-size circuits [26, 18], or if one-way functions exist [33, 17].

For all three models, our constructions require the development of new methods for applying tools from cryptography and derandomization to coding-theoretic problems. We give a brief discussion of these techniques next. A more detailed discussion of the approach behind our code construction appears in Section 4.

1.2 Techniques

Control/payload construction. In our constructions, we develop several new techniques for coding theory. 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 [1]; 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 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 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 [23, 12, 24]; our use of this connection is significantly more delicate.

2 Background and Relation to 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 [9, 32] and has witnessed a lot of recent algorithmic work (cf. the survey [14]). Under list decoding, the decoder outputs a small list of messages that must include the correct message. Random coding arguments assert the existence of binary codes of rate $1-H(p)-\varepsilon$ for which error-correction against adversarial errors is possible in this model when the decoder is allowed to output a list of size $O(1/\varepsilon)$ [10, 34, 15]. If a small amount of auxiliary information can be communicated on a noiseless side channel, then it becomes possible to pick the correct element from the list with high probability [13]. 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 space- or time-bounded channels. As mentioned above, the model we consider is slightly weaker than the standard one, in that we assume that the received word is obtained by corrupting a true output of the randomized encoder.

Adding Setup: Shared Randomness. Another relaxation that increases codes' power is to allow randomized coding strategies where the sender and receiver share "secret" randomness, *hidden from the channel*, which is used to pick a coding scheme at random from a family of codes (such codes were called *private codes* in [20]). Using such strategies, one can achieve the capacity $1 - H(p)$ against ADV_p (for example, by randomly permuting the symbols and adding a random offset [23, 20]). Using explicit codes achieving capacity on the BSC_p [11], 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 [29]). A related notion of setup is the *public key* model of Micali *et al.* [24], in which the sender generates a public key which is assumed to be known to the receiver and possibly the channel. This model only makes sense for computationally bounded channels, discussed below.

Our constructions are the first (for all three models) which 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 [22]. 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 $\vec{s} = (s_1, ..., s_n) \in \mathcal{S}^n$: 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 state vector in an AVC *oblivious* to the codeword and 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 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 [6, 7]. 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. [1] Note that codes providing this guarantee *cannot* be linear, since the bad error vectors for all codewords are the same in a linear code. The decoding rule used in [6] to prove this claim was quite complex, and it was simplified to the more natural closest codeword rule in [7]. Langberg [21] 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 and simpler existential proof. More importantly, we provide the first explicit constructions of codes for this model which achieve the optimal rate $1 - H(p)$.

Computationally bounded channels. In a different vein, Lipton [23] considered channels whose behavior can be described by a polynomial-time algorithm. Lipton 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.* [24] 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 trade-off between the channel's time and its probability of success.

Our list decoding result removes the setup assumptions of [23, 24] at the price of imposing a specific polynomial bound on the channels running time and relaxing to list-decoding.

However, our result also implies stronger *unique decoding* results in the public-key model [24]. 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 the channel). The idea, roughly, is to sign messages before encoding them; see [24] for details.

Logarithmic space channels. Galil *et al.* [12] 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 [25], they removed the assumption of one-way functions from Lipton's construction in the shared randomness model [23].

We add nonuniformity to their model to get a common generalization of arbitrarily varying channels. Our code construction for logarithmic space channels removes the assumption of shared

[1] 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 we consider the worst-case message, but allow the encoder some local random coins. This is a strict strengthening of the model as long as the decoder recovers the random coins r along with message m. The results of [6, 21] also apply to this stronger model.

setup in the model of [12], at the price of achieving list-decoding. This relaxation is in some sense necessary since unique decoding in this model is impossible when $p > 1/4$.

3 Statements of Results

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

3.1 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 a cryptographic motivation in [4]. The decoder does not have access to the randomness r to "sign" the message m. The linearity of the list-decodable code is therefore crucial to make the combination with AMD codes work. The linearity ensures that the spurious messages output by the list-decoder are all additive offsets of the true message that depend only on the error vector (and not on m, r). 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 and proved in Section 5.

Theorem 3.1. *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, $\Pr_r\big[\mathsf{Dec}\big(\mathsf{Enc}(m,r)+e\big) = m\big] \geqslant 1 - 2^{-\Omega_{\varepsilon,p}(n)}$. 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 $1 - 2^{-\Omega_{\varepsilon,p}(n)}$.*
Given an explicit family of linear binary codes of rate R that can be efficiently list-decoded from fraction p of errors with constant polynomial list-size $L = L(p,n)$, we can construct explicit stochastic codes with the above guarantee with rate R along with an efficient *decoder.*

Explicit, efficient codes achieving capacity. For 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 4 and in detail in Section 6.

Theorem 3.2. *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 $R \geqslant 1 - H(p) - \varepsilon$ which corrects a p fraction of additive errors with probability $1 - o(1)$. Specifically, there is a polynomial time encoder* Enc *and a polynomial time decoder* Dec *such that for* every *message $m \in \{0,1\}^{RN}$ and* every *error vector e of Hamming weight at most pN, we have* $\Pr_r(\mathsf{Dec}(\mathsf{Enc}(m;r) + e) = m) = 1 - \exp(-\Omega_\varepsilon(N/\log^2 N))$.

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 uniformly random and unknown to the channel (Theorem B.3).

3.2 Codes for online log-space-bounded channels

We generalize the model of Galil *et al.* [12] to capture both finite automaton-based models as well as arbitrarily varying channels. To model channels (as opposed to Boolean functions), we augment standard branching programs with the ability to output bits at each step.

Definition 1 (Space bounded channels)**.** *An* online-space-S channel *is a read-once branching program of width $\leqslant 2^S$ that outputs one bit at each computation step. Specifically, let $Q = \{0,1\}^S$ be a set of 2^S states. For input length N, the channel is given by a sequence of N transition functions $F_i : Q \times \{0,1\} \to Q \times \{0,1\}$, for $i = 1$ to N, along with a start state $q_0 \in Q$. On input $x = (x_1 x_2 \cdots x_N) \in \{0,1\}^N$, the channel computes $(q_i, y_i) = F_i(q_{i-1}, x_i)$ for $i = 1$ to N. The output of the channel, denoted $A(x)$, is $y = (y_1 y_2 \cdots y_N) \in \{0,1\}^N$.*

A randomized online-space-S channel *is a probability distribution over the space of deterministic online-space-S channels. For a given input x, such a channel induces a corresponding distribution on outputs. A randomized channel $\mathcal{A}$ is* pN-bounded with probability $1 - \beta$ *if, for all inputs $x \in \{0,1\}^N$, with probability at least $1 - \beta$, the channel flips fewer than pN bits of x, that is* $\Pr_{A \in \mathcal{A}}[\text{weight}(x \oplus A(x)) > pN] \leqslant \beta$.

We exhibit a very simple "zero space" 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, set it to 0, or set 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.* [8] for online channels in the special case where $p > 1/4$. See Appendix C for details.

Theorem 3.3 (Impossibility of unique decoding for $p > \frac{1}{4}$)**.** *For every pair of randomized encoding/decoding algorithms* $\mathsf{Enc}, \mathsf{Dec}$ *that make n uses of the channel and use a message space whose size tends to infinity with n, 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 and causes a uniformly random message to be incorrectly decoded with probability $\Omega(\nu)$.*

For list-decoding, we provide a positive result, namely, a construction of codes with rate approaching $1 - H(p)$ that efficiently recover a short list containing the correct message when the channel uses logarithmic space. For details of the construction and analysis, see Section 7. The structure of the code is similar to the uniquely decodable code for additive errors; however, additional work is needed to make the codewords appear pseudorandom to the channel, and the analysis is much more subtle.

Theorem 3.4 (Corollary of Theorem 7.3)**.** *For every $p \in (0, 1/2)$ and constant $\varepsilon > 0$, there is an efficient Monte Carlo construction of a stochastic code with encoding/decoding algorithms* $(\mathsf{Enc}, \mathsf{Dec})$ *such that for every message $m \in \{0,1\}^{(1-H(p)-\varepsilon)N}$ and every randomized online-space-S channel*

W_S *on* N *input bits that is* pN*-bounded (where* $\Omega(\log N) \leqslant S \leqslant o(N/\log N)$*), with high probability over the choice of coins* r *and the errors introduced by* W_S*,* $\mathsf{Dec}(\mathsf{W}_S(\mathsf{Enc}(m;r)))$ *outputs a list of at most* $\mathrm{poly}(1/\varepsilon)$ *messages that includes the real message* m*.*

The probability of incorrect decoding is at most $N2^{-\Omega(\varepsilon^3 S)} + 2^{-\Omega(\varepsilon^3 N/S)}$*, and the running time of* $(\mathsf{Enc}, \mathsf{Dec})$ *is is polynomial in* N *and* 2^S *(and therefore polynomial in* N *for log-space channels).*

3.3 List-decoding for Time-bounded Channels

Finally, we prove a similar result for time-bounded channels, assuming the existence of certain pseudorandom generators (which in turn follow from standard complexity assumptions). The model here is easy to describe: it suffices that the channel be implementable by a circuit of size N^c for some $c \geqslant 1$.

The details appear in Section 8.

Theorem 3.5. *Assume either* $\mathsf{E} \not\subseteq \mathsf{SIZE}(2^{\varepsilon_0 n})$ *for some* $\varepsilon_0 > 0$ *or the existence of one-way functions. 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* $(\mathsf{Enc}, \mathsf{Dec})$ *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* $1 - N^{-\Omega(1)}$*.*

4 Construction overview

Construction Ideas for Additive Errors. Our result is obtained by combining several ingredients in pseudorandomness and coding theory. At a high level the idea (introduced by Lipton [23] 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 construction [11]) can be used to communicate on ADV_p after the codewords symbols are randomly permuted. This allows one to achieve capacity against adversarial errors when the encoder and decoder share randomness that is unknown to the adversary causing the errors. But, crucially, this requires the decoder to know the random permutation that was used at the encoding.

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

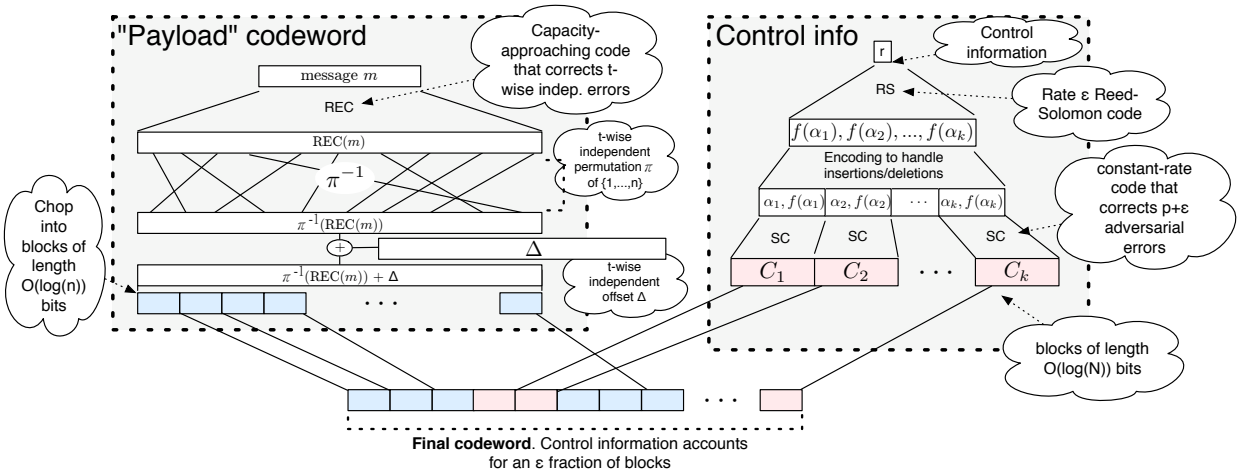

Figure 1: Schematic description of encoder from Algorithm 1.

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

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

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

One important issue we have glossed over is that a completely random permutation of the n bits of the payload codeword will take $\Omega(n \log n)$ bits to specify. This would make the control information too big compared to the message length (whereas we need it to be a tiny fraction of the message length). Therefore, we use almost t-wise independent permutations for $t \approx \varepsilon n / \log n$.

Such permutations can be sampled with $\approx \varepsilon n$ random bits. We then make use of the fact that C_{BSC} enables reliable decoding even when the error locations have such limited independence instead of being a uniformly random subset of all possible locations [29].

Extending the Construction to log-space and poly-time channels. The construction for additive channels does not work against more powerful channels for two reasons: first, the channel may be able to learn which blocks of the codeword contain the control information and concentrate errors on those blocks. Second, 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.

We overcome the first obstacle by making sure that transmitted word is indistinguishable from a random string by a log-space (resp. poly-time) test, even one which knows the message and certain parts of the control information. This ensures that the channel's placement of errors is "independent" (according to log-space tests) from the locations of the control blocks. It also ensures that errors are "randomly distributed" after unscrambling in the sense that the events that we needed to happen with high probability for successful decoding against oblivious errors will also happen with good probability against errors caused by the log-space (or poly-time-bounded) channel. Ensuring that this happens and leads to correct decoding constitutes the bulk of our analysis. This part is harder for the space-bounded case, since Nisan's generator only ensures that the error distribution caused by the channel is indistinguishable from oblivious errors by *online* space-bounded machines. 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 have to resort an indirect argument based on showing almost $\log n$-wise independence of certain events related to the payload decoding.

We overcome the second obstacle 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, and the list of candidates will include the correct control information with high probability.

5 List decoding implies codes for worst-case additive errors

In this section, we will demonstrate how to use good *linear* list-decodable codes to get good stochastic codes. The conversion uses the list-decodable code as a black-box and loses only a negligible amount in rate. In particular, by using binary *linear* codes that achieve list decoding capacity, we get stochastic codes which achieve the capacity for additive errors. The linearity of the code is crucial for this construction. The other ingredient we need for the construction is an authentication code ("MAC") that can detect *additive corruption* with high probability, which has been studied under the label of Algebraic manipulation detection (AMD) codes [4].

5.1 Some coding terminology

We begin with the definitions relating to list decoding and stochastic codes for additive errors.

Definition 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 $2pn$.* □

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

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

*Such a code is said to be (efficiently) p-*decodable with probability $1-\delta$ *if there is a (deterministic polynomial time computable) decoding function $\mathsf{Dec} : \{0,1\}^n \to \{0,1\}^{Rn} \cup \{\perp\}$ such that for every $m \in \{0,1\}^{Rn}$ and every error vector $e \in \{0,1\}^n$ of Hamming weight at most pn, with probability at least $1 - \delta$ over the choice of a random string $\omega \in \{0,1\}^b$, we have*

$$\mathsf{Dec}\big(\mathsf{Enc}(m, \omega) + e\big) = m \ .$$

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.

5.2 Algebraic manipulation detection (AMD) codes

The following is not the most general definition of AMD codes from [4], but will suffice for our purposes and is the one we will use.

Definition 4. *Let $\mathcal{G} = (G_1, G_2, G_3)$ be a triple of abelian groups (whose group operations are written additively) and $\delta > 0$ be a real. Let $G = G_1 \times G_2 \times G_3$ be the product group (with 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:*

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

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

Intuitively, the AMD allows one to authenticate x via a signed form $(x, r, f(x, r))$ so that an adversary who manipulates the signed value by adding an offset Δ cannot cause incorrect decoding of some $x' \neq x$. The following concrete scheme from [4] achieves near optimal tag size and we will make use of it.

Theorem 5.1. *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 $\left(\mathcal{G}, \frac{d+1}{q}\right)$-AMD code with tag size $2 \log q$ where $\mathcal{G} = (\mathbb{F}^d, \mathbb{F}, \mathbb{F})$.*[2]

5.3 Combining list decodable and AMD codes

Using a (p, L)-list decodable code C of length n, for any error pattern e of weight at most pn, we can recover a list of L messages that includes the correct message m. We would like to use the stochastic portion of the encoding to allow us to unambiguously pick out m from this short list. The key insight is that if C is a *linear* code, then the other (less than L) messages in the list are all fixed offsets of m that depend *only* on the error pattern e. So if prior to encoding by the list-decodable code C, the messages are themselves encodings as per a good AMD code, and the tag portion of the AMD code is good for these fixed L or fewer offsets, then we can uniquely detect m from the list using the AMD code. If the tag size of the DMD 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 5.2 (Stochastic codes from list decoding and AMD)**.** *Let b, d be positive integers with d odd and $k = b(d+2)$. Let $C : \mathbb{F}_2^k \to \mathbb{F}_2^n$ be the encoding function of a binary linear (p, L)-list decodable code. Let $f^{(d)}_{\mathrm{AMD}}$ be the function from Theorem 5.1 for the choice $\mathbb{F} = \mathbb{F}_{2^b}$. Let C' be the stochastic binary code with encoding map $E : \{0,1\}^{bd} \times \{0,1\}^b \to \{0,1\}^n$ given by*

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

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

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

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

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

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

Let us now analyze the above decoder D. First consider the case when $\mathrm{wt}(e) \leqslant pn$. 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 $(m, r, f^{(d)}_{\mathrm{AMD}}(m, r))$, say this happens for

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

$i = 1$ w.l.o.g. Therefore, the condition $f^{(d)}_{\mathrm{AMD}}(m_1, r_1) = s_1$ will be met and we only need to worry about this happening for some $i > 1$ also.

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

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

Corollary 5.3. *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 are strongly p-decodable 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 1. For the above construction, if the decoding succeeds, 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 Explicit Codes of Optimal Rate for Additive Errors

6.1 Ingredients

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

- A constant-rate explicit stochastic code $\mathsf{SC} : \{0,1\}^b \times \{0,1\}^b \to \{0,1\}^{c_0 b}$, defined on blocks of length $c_0 b = \Theta(\log N)$, that is efficiently 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 3.1 (detailed statement appears in the Appendix as Proposition A.1).
- A rate $O(\varepsilon)$ Reed-Solomon code RS which encodes a message as the evaluation of a polynomial at points $\alpha_1, ..., \alpha_\ell$ in such a way that an efficient algorithm RS-Decode can efficiently recover the message given at most $\varepsilon\ell/4$ correct symbols and at most $\varepsilon/24$ incorrect ones.

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

$\Lambda \leftarrow 2c_0$ ▷ Here $c_0 = c_0(p+\varepsilon)$ is the constant in the stochastic code from Proposition A.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

Select $s_\pi \leftarrow_R \{0,1\}^{\varepsilon^2 N}$. ▷ s_π is a seed for picking a permutation of $[N']$ from an almost t-wise independent family as per Proposition A.5, where $t = \Omega(\varepsilon^2 N / \log N)$.

Select $s_\Delta \leftarrow_R \{0,1\}^{\varepsilon^2 N}$. ▷ s_Δ is a seed for picking a t'-wise independent string Δ, where $t' = \Omega(\varepsilon^2 N / \log N)$ as per Proposition A.6.

Select $s_T \leftarrow_R \{0,1\}^{\varepsilon^2 N}$. ▷ s_T is a seed for sampling a pseudorandom subset $T \subset [n] = [n' + \ell]$ of size ℓ as per Proposition A.4.

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

Phase 2: Encode control information

Let $\mathbb{F} = \mathbb{F}_N$ and $S = (\alpha_1, \ldots, \alpha_\ell) \subseteq \mathbb{F}$ be an arbitrary subset of size ℓ.
Compute $(a_1, ..., a_\ell) \leftarrow \mathsf{RS}_{\mathbb{F},S,\ell,|\omega|/\log N}(\omega)$.

▷ RS (defined in (3)) is a rate $\varepsilon/8$ Reed-Solomon code of length $24\varepsilon N = \frac{8}{\varepsilon} \cdot |\omega|$ bits, i.e., $\ell = 24\varepsilon N / \log N$ field symbols.

for $i \leftarrow 1$ to ℓ **do**

  $A_i \leftarrow (\alpha_i, a_i)$

  ▷ Add location information to each RS symbol to get block A_i of $2 \log N$ bits.

  Set $C_i \leftarrow \mathsf{SC}(A_i, r_i)$, where $r_i \leftarrow_R \{0,1\}^{2 \log N}$.

  ▷ Here $\mathsf{SC} = \mathsf{SC}_{2\log N, p+\varepsilon} : \{0,1\}^{2\log N} \times \{0,1\}^{2\log N} \to \{0,1\}^{\Lambda \log N}$ is a stochastic code that can correct a fraction $(p+\varepsilon)$ of additive errors with probability $1 - c_1/N^2 > 1 - 1/N$ as per Proposition A.1.

  ▷ The control information ω is thus encoded by a concatenated code with an outer Reed-Solomon code and inner code SC.

end for

Phase 3: Generate the payload codeword

$P \leftarrow \mathsf{REC}(m)$, ▷ $\mathsf{REC} : \{0,1\}^{R'N'} \to \{0,1\}^{N'}$ is a code that can correct a $p + 25\Lambda\varepsilon$ fraction of *t-wise independent* errors, as per Proposition A.7. Here $R' = \frac{RN}{N'}$.

$\pi \leftarrow \mathsf{KNR}(s_\pi)$ ▷ Generate permutation $\pi : [n] \to [n]$ using Proposition A.5.

$\Delta \leftarrow \mathsf{POLY}(s_\Delta)$ ▷ Generate random offset string $\Delta \in \{0,1\}^n$ as guaranteed by Proposition A.6.

$\pi^{-1}(P) \leftarrow$ (bits of P permuted according to π^{-1})

$Q \leftarrow \pi^{-1}(P) \oplus \Delta$

Cut Q into n' blocks $B_1, ...B_{n'}$ of length $\Lambda \log N$ bits. ▷ Recall that $n' = \frac{n}{\Lambda \log N}$.

Phase 4: Interleave blocks of payload codeword and control codeword

$T \leftarrow \mathsf{Samp}(s_T)$ ▷ Generate pseudorandom size-ℓ subset of $[n' + \ell]$ as locations for control blocks, using sampler of Proposition A.4.

Interleave blocks $C_1, ..., C_\ell$ with blocks $B_1, ..., B_{n'}$, using C_i blocks in positions from T and B_i blocks in positions from $\overline{T} = [n' + \ell] \setminus T$.

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

▷ The decoder's pseudocode is annotated with statements about performance. These claims assume that $x = \mathsf{Enc}(m; \omega, r_1, \ldots, r_\ell) + e$ where e contains at most a fraction p of ones and the random string $(\omega; r_1, r_2, \ldots, r_\ell)$ is uniform and independent of the pair (m, e).

Cut x into $n' + \ell$ blocks $x_1, ..., x_{n'+\ell}$ of length $\Lambda \log(n)$ each.
for $i \leftarrow 1$ to $n' + \ell$ **do**
  $\tilde{F}_i \leftarrow$ SC-DECODE(x_i).

▷ Run the decoder for the stochastic code SC used to encode the symbols of the RS codeword encoding the control blocks.

▷ With high prob, non-control blocks are rejected (Lemma 6.4), and control blocks are either correctly decoded or discarded (Lemma 6.3).

  if $\tilde{F}_i \neq \perp$ **then**
    Parse $\tilde{F}_i$ as $(\tilde{\alpha}_i, \tilde{a}_i)$, where $\tilde{\alpha}_i, \tilde{a}_i \in \mathbb{F}_N$.
  end if
end for
$(\tilde{s}_T, \tilde{s}_\Delta, \tilde{s}_\pi) \leftarrow$ RS-DECODE$\Big($pairs $(\tilde{\alpha}_i, \tilde{a}_i)$ output above$\Big)$.

▷ With high prob., enough control blocks are decoded correctly to recover the control information (Lemma 6.5).

$\tilde{T} \leftarrow \mathsf{Samp}(\tilde{s}_T)$,
$\tilde{\Delta} \leftarrow \mathsf{POLY}(\tilde{s}_\Delta)$
$\tilde{\pi} \leftarrow \mathsf{KNR}(\tilde{s}_\pi)$
$\tilde{Q} \leftarrow$ concatenation of blocks x_i in $[n' + k] \setminus \tilde{T}$

▷ Fraction of errors in $\tilde{Q}$ is at most $p + O(\varepsilon)$.

$\tilde{P} \leftarrow \pi(\tilde{Q} \oplus \tilde{\Delta})$

▷ If control info is correct, then errors in $\tilde{P}$ are almost t-wise independent.

$\tilde{m} \leftarrow$ REC-DECODE$(\tilde{P})$

▷ Run the decoder from Proposition A.7.

▷ With high prob., $\tilde{m} = m$

- 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 [31].
- A generator $\mathsf{KNR} : \{0,1\}^\sigma \to S_n$ for an (almost) t-wise independent family of permutations of the set $\{1, ..., n\}$, that uses a seed of $\sigma = O(t \log n)$ random bits (Kaplan, Naor, and Reingold [19]).
- 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 $\pi \in_R \text{range}(\mathsf{KNR})$. (Here $\pi(e)$ denotes the permuted vector: $\pi(e)_i = e_{\pi(i)}$.) A standard family of concatenated codes satisfies this property (Smith [29]).

6.2 Main theorem on codes for additive error

The following (Theorem 3.2, restated) is our result on explicit construction of capacity-achieving codes for additive errors.

Theorem 6.1. *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 $\Pr_\omega\big[\text{DECODE}(\text{ENCODE}(m;\omega)+e) = m\big] \geqslant 1 - \exp(-\Omega(\varepsilon^2 N/\log^2 N)))$, 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 6.1. The encoding algorithm is given in Algorithm 1 (page 14). The corresponding decoder is given in Algorithm 2 (page 15). 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 4 when reading the formal description.

Starting the Proof of Theorem 6.1. The rate R of the overall code is almost equal to the rate R' of the code REC used to encode the actual message bits m, since the encoded control information has length $O(\varepsilon N)$ which is much smaller than the number of message bits (by picking ε small enough). The code REC needs to correct a fraction $p + 25\Lambda\varepsilon$ of t-wise independent errors, so we can pick $R' \geqslant 1 - H(p) - O(\varepsilon)$. Now the rate $R = \frac{R'N'}{N} = R'(1 - 24\Lambda\varepsilon) \geqslant 1 - H(p) - O(\varepsilon)$ (for small enough $\varepsilon > 0$).

We now turn to the analysis of the decoder. Fix a message $m \in \{0,1\}^{R \cdot N}$ and an error vector $e \in \{0,1\}^N$ with Hamming weight at most pN. Suppose that we run 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 $x_1, ..., x_{n'+\ell}$ of length $\Lambda \log N$, corresponding to the blocks output by the encoder.

The suite of four lemmas below which are proved in Section 6.3 show that the decoder recovers the control information correctly with high probability. Conditioning on correct recovery of the control information, we then show that the payload message is correctly recovered. The proof of the theorem is completed in Section 6.4.

Definition 5 (Good sampler seeds)**.** *A sampled set T is* good *for error vector e if the fraction of control blocks with relative error rate at most $p + \varepsilon$ is at least $\frac{\varepsilon}{2}$.* □

Lemma 6.2 (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_T, the set T is good for e.*

Lemma 6.3 (Control blocks lemma)**.** *For any e, T such that T is good for e, with probability at least $1 - \exp(-\Omega(\varepsilon^3 N/\log N))$ over the random coins $(r_1, r_2, \ldots, r_\ell)$ used by the ℓ SC encodings, we have:*

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

Lemma 6.4 (Payload blocks lemma)**.** *For every* m, e, s_T, 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* $\frac{\varepsilon\ell}{24}$.

Lemma 6.5 (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* $\tilde{\omega} = \omega$.

Remark 2. It would be interesting to achieve an error probability of $2^{-\Omega_\varepsilon(N)}$, i.e., a positive "error exponent," in Theorem 6.1 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 [6] achieves a positive error exponent for all rates less than capacity, as does our existence proof using list decoding in Section 5.3.

Remark 3. 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.3 in Appendix B.

6.3 Proofs of Lemmas used in Theorem 6.1

Proof of Lemma 6.2. Let $B \subset [n] = [n' + \ell]$ be the set of blocks that contain a $(p+\varepsilon)$ or smaller fraction of errors. We first prove that B must occupy at least an ε fraction of total number of blocks: to see why, let γ be the proportion of blocks which have error rate at most $(p+\varepsilon)$. The total fraction of errors in x is then at least $(1-\gamma)(p+\varepsilon)$. Since this fraction is at most p by assumption, we must have $1-\gamma \leqslant p/(p+\varepsilon)$. So $\gamma \geqslant \varepsilon/(p+\varepsilon) > \varepsilon$.

Next, we show that the number of *control blocks* that have error rate at most $p+\varepsilon$ cannot be too small. The error e is fixed before the encoding algorithm is run, and so the sampler seed s_T is chosen independently of the set B. Thus, the fraction of control blocks in B will be roughly ε. Specifically, we can apply Proposition 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.) □

Proof of Lemma 6.3. Fix e and the sampled set T which is good for e. Consider a particular received block x_i that corresponds to control block j, that is, $x_i = C_j + e_i$. The key observation is that the error vector e_i depends on e and the sampler seed T, but it is *independent* of the randomness used by SC to generate C_j. Given this observation, we can apply Proposition 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 6.4. Consider a block x_i that corresponds to payload block j, that is, $x_i = B_j + e_i$. Fix e, s_T, and s_π. The offset Δ is independent of these, and so we may write $x_i = y_i + \Delta_i$, where y_i is fixed independently of Δ_i. Since Δ is a t'-wise independent string with $t' = \Omega(\varepsilon^2 N/\log N)$ much greater than the size $\Lambda \log N$ of each block, the string Δ_i is uniformly random in $\{0,1\}^{\Lambda \log N}$. Hence, so is x_i. By Proposition 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 [3, 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 6.5. Suppose the events of Lemmas 6.3 and 6.4 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, 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 6.3 and 6.4, 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. □

6.4 Completing the Proof of Main Theorem 6.1

Proof of Theorem 6.1. We will first prove that the decoding of the payload codeword succeeds assuming the correct control information $\omega = (s_\pi, s_\Delta, s_T)$ is handed directly to the decoder, i.e., 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_T, and let e_Q be the restriction of e to the payload codeword, *i.e.*, blocks not in T. The relative weight of e_Q is at most $\frac{pN}{N'} = p\frac{N' + \ell\Lambda \log N}{N'} = p(1 + 24\varepsilon\Lambda\frac{N}{N'}) \leqslant p(1 + 25\Lambda\varepsilon)$ (for sufficiently small ε).

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

Thus the input to REC is corrupted by a fraction of at most $p(1 + 25\Lambda\varepsilon)$ errors which are t-wise independent, in the sense of Proposition A.7 [29]. Thus, with probability at least $1 - e^{-\Omega(\varepsilon^2 t)} = 1 - e^{-\Omega(\varepsilon^4 N/\log N)}$, the message m is correctly recovered by DECODE.

In the actual 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 6.2, 6.3, 6.4, and 6.5) show that the control information is correctly recovered, i.e., $\tilde{\omega} = \omega$, 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 6.1. □

7 Capacity-achieving codes for online space-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 online log-space 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. If the channel is allowed a space bound $S = o(N/\log N)$, our construction and decoding times are polynomial in the block length N of the code and 2^S.

7.1 Channel models and branching programs

We model space-bounded online channels as a restricted form of bounded-width branching programs. Galil *et al.* [12] formulated a uniform version of this model, with finite automata replacing branching programs. We use a nonuniform model since it simplifies several proofs and captures a broader class of channels, including AVC's.

Definition 6 (Branching programs)**.** *A* layered, oblivious branching program *of width 2^S and length ℓ is a sequence of ℓ transition functions $F_i : Q \times \{0,1\} \to Q$, where $Q = \{0,1\}^S$ is a set of 2^S states, together with a start state $q_0 \in Q$, an input map $I : [\ell] \to [N]$, and an output map $Out : Q \to \{0,1\}$. On input a binary string $(x_1 x_2 \cdots x_N) \in \{0,1\}^N$, the program computes $q_i = F_i(q_{i-1}, x_{I(i)})$ for $i = 1$ to ℓ. The (binary) output of the program is $Out(q_\ell)$.*

We define an online branching program *of width 2^S as a special case of the above, in which $\ell = N$ and I is the identity function, so that $q_i = F_i(q_{i-1}, x_i)$. Such a program is essentially a nonuniform finite automaton, for which the transition function can change from symbol to symbol. A* randomized online *branching program is a probability distribution over (deterministic) online branching programs (of the same input length and width bound).*

To model space-bounded *channels* (as opposed to Boolean functions), we augment the standard branching program model with the ability to output bits at each phase.

Definition 7 (Space bounded channels (Def. 1 restated))**.** *An* online-space-S channel *is a read-once width-2^S branching program that outputs one bit at each computation step. Specifically, let $Q = \{0,1\}^S$ be a set of 2^S states. For input length N, the channel is given by a sequence of N transition functions $F_i : Q \times \{0,1\} \to Q \times \{0,1\}$, for $i = 1$ to N, along with a start state $q_0 \in Q$. On input $x = (x_1 x_2 \cdots x_N) \in \{0,1\}^N$, the channel computes $(q_i, y_i) = F_i(q_{i-1}, x_i)$ for $i = 1$ to N. The output of the channel, which is denoted $A(x)$, is $y = (y_1 y_2 \cdots y_N) \in \{0,1\}^N$.*

A randomized (online-space-S channel *is a probability distribution over the space of deterministic (online-space-S) channels. For a given input x, such a channel induces a corresponding distribution on outputs. A randomized channel $\mathcal{A}$ is* pN-bounded with probability $1 - \beta$ *if, for all inputs $x \in \{0,1\}^N$, with probability at least $1 - \beta$, the channel flips fewer than pN bits of x, that is*

$$\Pr_{A \in \mathcal{A}}[\text{weight}(x \oplus A(x)) > pN] \leqslant \beta \ .$$

Remark 4 (Bounded Lookahead)**.** We may also consider a model in which the channel bases its decision about which bits to flip not only on positions seen so far, but some number of positions

in the future. A *channel with look-ahead t* is specified by $N + t$ transition functions. The input is augmented with t extra dummy bits, i.e. $x' = x\|0^t$, and the output of the channel is the last N bits produced by the transition functions, that is $y = y_{t+1}y_{t+2}\cdots y_{t+N}$. The results of this section extend directly to channels with $O(S)$ look-ahead.

7.2 Nisan's pseudorandom generator

The encoding function of our code will make use of Nisan's pseudorandom generator for fooling space bounded algorithms, or in the non-uniform setting, bounded width branching programs. We first define the notion of indistinguishability of two distributions relative to a function.

Definition 8 (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*

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

Theorem 7.1 (Nisan's PRG [25])**.** *For integers S', m, there exists $s \leqslant O(S' \log m)$ and a function $\mathsf{Nis} : \{0,1\}^{O(S' \log m)} \to \{0,1\}^m$ such that for every randomized online branching program $\mathcal{B}$ on m inputs of width $2^{S'}$,*

$$\mathsf{Nis}(U_s) \overset{2^{-S'}}{\approx}_{\mathcal{B}} U_m$$

where U_t denotes the uniform distribution on $\{0,1\}^t$. For such a generator, s is called its seed length, and $2^{-S'}$ its error.

7.3 Code construction and ingredients

We will use the same high level approach from our construction for the additive errors case, with some components changed, and with a seed in the control information to obtain an offset that looks uniformly random to online log-space bounded channels.

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

(i) N is the block length of the final code,

(ii) pN is the bound on the number of errors introduced ($0 < p < 1/2$) by the channel w.h.p.

(iii) S is a bound on the space of the online channel (as per Definition 7), such that S is both $\Omega(\log N)$ and $o(N/\log N)$.

(iv) ε is a measure of how far the rate is from the optimal bound of $1 - H(p)$ (that is, the rate is be at least $1 - H(p) - \varepsilon$). We will assume $0 < 2\varepsilon < 1/2 - p$.

The seeds/control information. The control information ω will consist of three randomly chosen strings s_π, s_T, s_Γ where s_π, s_T are as in the additive errors case. We will take the lengths of s_π, s_T be ζN where $\zeta = \zeta(p, \varepsilon)$ will be chosen small enough compared to ε (the exact choice of ζ is not too important, but we remark that choosing, say, $\zeta = \varepsilon^{10}$ should suffice).

The third string $s_\Gamma \in \{0,1\}^{\zeta N}$ will be a seed for Nisan's generator Nis for randomized online branching programs of width 2^{2S} with error $\eta_{\mathsf{Nis}} \leqslant 2^{-2S}$. Since $S \leqslant o(N/\log N)$, a seed length of ζN is sufficient for Nisan's generator from Theorem 7.1. The offset $\Gamma \leftarrow \mathsf{Nis}(s_\Gamma)$ will be used to fool the space-bounded channel. We won't need to add the t'-wise independent offset Δ as we did in the additive errors case.

Encoding the message. The payload codeword encoding a 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 log-space bounded channel instead of the t'-wise independent offset Δ.

We will use the fact that REC is a concatenated code with the following standard components:

- an outer code $\mathsf{REC}^{\mathrm{out}}$ (over an alphabet of size 2^a for some constant a) of rate at least $1-\varepsilon/10$ that can correct a constant fraction $\kappa = \kappa(\varepsilon) \geqslant \varepsilon/10$ of worst-case errors, and
- an inner code $\mathsf{REC}^{\mathrm{in}}$ of dimension a, block length $b_{\mathsf{data}} \leqslant O(\log(1/\kappa)/\varepsilon^2)$, and rate $a/b_{\mathsf{data}} \geqslant 1 - H(p) - \varepsilon/10$ that is capacity-achieving for the binary symmetric channel with crossover probability p with error probability at most $\kappa/10$. Such capacity-achieving codes exist by Shannon's theorem, and can be found in time $\exp(\mathrm{poly}(1/\varepsilon))$ (which is independent of n) by a brute-force search.

Thus the payload codeword will naturally be broken into n_{data} "inner chunks" of size b_{data}. This structure of REC will be important to argue that once the control information is correctly recovered, the decoder can find the actual message. This step is not as easy as in the additive errors case, since the error distribution is no longer a simple t-wise independent distribution but rather caused by an arbitrary space-bounded online channel.

For convenience, we will assume that b_{data} will divide the size of the control blocks b_{ctrl} (which will be $\Theta(\log N)$). Therefore, each block of the final codeword will either be a control block, or a payload block that consists of $b_{\mathsf{ctrl}}/b_{\mathsf{data}}$ consecutive "inner chunks" of the concatenated payload codeword.

Encoding the seeds. The control information (consisting of the seeds s_π, s_T, 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 *low-space pseudorandom code* LSC with good list decoding properties and such that the stochastic encoding of every message according to the code is indistinguishable from a random string by a randomized online space-S channel.

The formal properties needed from LSC are stated in Proposition 7.2 below — we will apply the Proposition with the choice $S_0 = 2S$ and list decoding radius $\delta = p + \varepsilon < 1/2 - \varepsilon$, obtaining a code of block length $2\Lambda_0 S$. The size b_{ctrl} of the control blocks will be set to be equal to the block length $2\Lambda_0 S$ of LSC. Let n_{ctrl} denote number of control blocks, which also equals the block length of the Reed-Solomon code. Note that

$$n_{\mathsf{ctrl}} \approx \varepsilon N / b_{\mathsf{ctrl}} = \Theta(\varepsilon N/S) \ .$$

The encoding of control blocks is exactly as in the additive errors case, with LSC replacing SC. 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_T.

Rate of the code. The code encoding the control information is of some small constant rate $R_{\mathsf{ctrl}}(p, \varepsilon)$ and the control information consists only of $O(\zeta N)$ bits. Given ε, we can select ζ small enough so that 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.

We next formally state and prove the properties of the code LSC needed above, thus completing the description of the code and encoding function.

7.4 Low-space pseudorandom stochastic code

Definition 9 (Decomposable stochastic code)**.** *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 a encoding decomposes as $E = [E_1, E_2]$.* □

Definition 10 (List-decodable low-space pseudorandom code)**.** *A decomposable binary stochastic code with encoding map $E : \{0,1\}^k \times \{0,1\}^s \to \{0,1\}^b$ that decomposes as $E = [E_1, E_2]$ is said to be a* (δ, L)-list decodable (S, γ)-pseudorandom code *if the following properties hold:*

1. *E is (δ, L)-list decodable, i.e., 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.*
2. *For every $m \in \{0,1\}^k$ and every randomized online branching program $\mathcal{B}$ (that can depend on $m, E_1(m)$) with b input bits and width 2^S, we have*

$$E(m, U_s) \overset{\gamma}{\approx}_{\mathcal{B}} U_b \ .$$

The rate *of such a code equals k/b, and its* seed complexity *is s.* □

The construction of the necessary stochastic code, whose codewords look random to low-space channels, is guaranteed by the following lemma.

Proposition 7.2 (Inner control codes)**.** *For some fixed positive integer Λ_0 the following holds. For all δ, $0 < \delta < 1/2$, 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 for all large enough integers S_0, there exists a $2^{O(S_0)}$ time randomized Monte Carlo construction of a decomposable stochastic code with encoding $E : \{0,1\}^k \times \{0,1\}^s \to \{0,1\}^u$ with $u = \Lambda_0 S_0$, $k \geqslant Ru$ and $s = 10S_0$, that is (δ, L)-list-decodable $(S_0, 2^{-S_0})$-pseudorandom with probability at least $1 - 2^{-u}$.*

Further, there exists a deterministic decoding procedure running in time $2^{O(S_0)}$ that, given a string $y \in \{0,1\}^u$, recovers the complete list of (at most L) pairs (m, r) whose encodings $E(m, r)$ are within Hamming distance at most δu from y.

Proof. 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 width 2^{S_0} branching programs, obtained by picking $\mathsf{BPPRG}(r) \in \{0,1\}^u$ independently and uniformly at random for each seed r. Here $u = \Lambda_0 S_0$ for a large enough absolute constant Λ_0 as in the statement of the Proposition. Note that the construction time is $2^{O(u)} = 2^{O(S_0)}$.

List-decoding property. We adapt the proof that a truly random set is list-decodable. Let $C \subseteq \{0,1\}^u$ be a linear $(\delta, L_C = L_C(\delta))$-list decodable code; such codes exist for rates less than $1 - H(\delta)$ [15], 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 $L_C \leqslant 1/(1/2-\delta)^{O(1)}$ [16]. 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 δu in $\{0,1\}^u$, 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 $\mathbb{E}[X_r] = |C| \cdot |B'|/2^u \leqslant 2^{-u(1-H(\delta)-R_C)}$ where R_C denotes the rate of C. Therefore,

$$\mathbb{E}[X] = 2^{10S_0} 2^{-u(1-H(\delta)-R_C)} = 2^{-u(1-H(\delta)-R_C-10/\Lambda_0)} .$$

Suppose $R_C + 10/\Lambda_0 = 1 - H(\delta) - \alpha_0$, so that $\mathbb{E}[X] = 2^{-\alpha_0 u}$. 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 $\left(\frac{t}{e}\right)^{t\mathbb{E}[X]/L_C}$. Simplifying, we get $\Pr[X > L] \leqslant (\frac{L}{e})^2 2^{-3u} \leqslant 2^{-2u}$.

Taking a union bound over all 2^u possible balls B', we get that with probability at least $1 - 2^{-2u}$, 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. We now establish the pseudorandomness claim. It suffices to prove the pseudorandomness property against all deterministic online branching programs of width 2^{S_0}, since a randomized online branching program is just a distribution over deterministic branching programs.

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 $\{0,1\}^u$. Fix an online width-2^{S_0} branching program B. 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 $\Pr[B(U_u) = 1]$ that B accepts a uniformly random string by more than ζ in absolute value, is at most $\exp(-\Omega(\zeta^2 |X_m|))$. For $\zeta = 2^{-S_0}$ and $|X_m| = 2^s \geqslant 2^{10S_0}$, this probability is at most $\exp(-\Omega(2^{8S_0}))$.

The number of online branching programs of width 2^{S_0} on u input bits is at most $\exp(O(S_0 u) 2^{S_0}) \leqslant \exp(O(2^{2S_0}))$. By a union bound over all these branching programs, we conclude that except with probability at most $\exp(-2^{\Omega(S_0)})$ over the choice of BPPRG, the following holds for *every* online width 2^{S_0} branching program B with u inputs bits:

$$\left| \Pr_{x \in X_m}[B(x) = 1] - \Pr[B(U_u) = 1] \right| \leqslant 2^{-S_0} .$$

Since m was arbitrary, we have proved that the constructed stochastic code is $(S_0, 2^{-S_0})$-pseudorandom with probability at least $1 - 2^{-2u}$.

DECODING. Finally, it remains to justify the claim about the decoding procedure. Given a string $y \in \{0,1\}^u$, 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 $\mathrm{dist}(E(m,r), y) \leqslant \delta u$. By the list-decoding property, there will be at most L such pairs (m, r). The decoding complexity is $2^{O(k+s)} = 2^{O(u)}$. □

7.5 List decoding algorithm

The decoding algorithm will be similar to the additive case with the principal difference being that the inner stochastic codes will be decoded as per the procedure guaranteed in Proposition 7.2 (using time $2^{O(S)}$ for each block). For each block, we obtain a list of L possible pairs of the form (α_i, a_i). These set of (at most NL pairs) are the fed into the polynomial time Reed-Solomon list decoding algorithm (guaranteed by Proposition A.3), which returns a list of $\mathrm{poly}(1/\varepsilon)$ 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 $2^{O(S)}$ and decoding the Reed-Solomon code as well as REC takes time polynomial in N. So the overall run time is polynomial in N and 2^S.

The main theorem about decoding is the following.

Theorem 7.3. *Let W_S be an arbitrary randomized online-space-S channel on N input bits that is pN-bounded with probability $1 - \beta$. Consider the code construction described in Section 7.3 using component codes REC, a Reed-Solomon code of small enough rate R^{RS}, and the code LSC that is $(p + \varepsilon, L)$-list decodable and $(2S, 2^{-2S})$-pseudorandom (which happens with $1 - 2^{-\Omega(S)}$ probability).*

Then for every message m, with high probability over the choice of control information s_π, s_T, s_Γ and the errors introduced by W_S, the list output by the above-mentioned list decoding algorithm includes the message m with probability at least

$$1 - 2\beta - N2^{-\Omega(\varepsilon^3 S)} - 2^{-\Omega(\varepsilon^3 N/S)} .$$

The running time of the decoding algorithm is polynomial in N and 2^S (and thus polynomially bounded in the block length if the space bound is logarithmic).

Since the construction of LSC in Proposition 7.2 guarantees the required pseudorandomness property with probability $1 - 2^{-\Omega(S)}$, setting $S = \Theta(\log N)$ in the above theorem implies our main result (Theorem 3.4) on capacity-achieving codes for list decoding on online log-space channels.

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 7.3.

7.6 Analyzing Decoding: Main Steps

Our analysis requires two main claims.

Lemma 7.4 (Few Control Candidates)**.** *The decoder recovers a list of $L' \leqslant \text{poly}(1/\varepsilon)$ candidate values of the control information. With high probability (specifically probability at least $1 - \beta - \exp(-\Omega(\varepsilon^3 N/S))$), the list includes the correct value $\omega = (s_\pi, s_T, s_\Gamma)$ of the control information used at the encoder.*

Lemma 7.5 (Payload decoding succeeds)**.** *Given the* correct *control information (π, T, Γ), the decoder succeeds with high probability. Specifically, the probability of successful decoding is at least $1 - \beta - N\exp(-\Omega(\varepsilon^3 S))$).*

Combining these two lemmas, which we prove in the next two sections, we get that except with probability at most $2\beta + \exp(-\Omega(\varepsilon^3 N/S)) + N\exp(-\Omega(\varepsilon^3 S))$, the decoder recovers a list of at most $L' \leqslant \text{poly}(1/\varepsilon)$ potential messages, one of which is the correct original message. This establishes Theorem 7.3.

7.7 Control Candidates Analysis

In this section we show that the decoder can recover a small list of candidate control strings, one of which is correct with high probability (Lemma 7.4).

Our analysis follows the case of additive errors, but the relaxed goal of list-decoding simplifies the analysis of this part considerably. Recall that the sampled set T is "good" for a particular error pattern (Definition 5) if at least a fraction ε of the n_{ctrl} control blocks have an error rate (fraction of flipped bits) bounded above by $p + \varepsilon$.

There were four main lemmas in the analysis of additive errors. We can reuse the first (Lemma 6.2) verbatim, with the only change in the calculation being that the number of control blocks n_{ctrl} is now $\Theta(\varepsilon N/S)$ instead of $\Theta(\varepsilon N/\log N)$.

Lemma 7.6 (Good Samplers: Lemma 6.2 Restated)**.** *For every error vector e of weight at most pN, with probability at least $1 - \exp(-\Omega(\varepsilon^3 N/S))$ over the choice of sampler seed s_T, the set T is good for e.*

Lemma 7.7 (Correct Control Blocks — list decoding version)**.** *For any e, T such that T is good for e, the decoding algorithm for the inner codes LSC outputs a list of L symbols containing the correct symbol a_i for at least $\frac{\varepsilon n_{\mathsf{ctrl}}}{2} = \Theta(\frac{\varepsilon^2 N}{S})$ control blocks.*

Proof. The list-decoding radius of the LSC 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 6.4), which previously stated that very few payload blocks are mistaken for control blocks, is the piece of the analysis that requires the most significant change. It is possible for the space-bounded channel to inject fake control blocks into the codeword (by changing a block to some pre-determined codeword of LSC). Therefore we can only say that the total number of candidate control blocks is small.

Lemma 7.8 (Bounding mistaken control blocks)**.** *For every m, e, ω, the total number of candidate control symbols is at most $\frac{NL}{2\Lambda_0 S}$.*

Proof. Since each block has $b_{\mathsf{ctrl}} = 2\Lambda_0 S$ bits, there are $\frac{N}{2\Lambda \log N}$ blocks considered by the decoder. The list decoding of each such block yields at most L candidate control symbols. □

Given Lemmas 7.7 and 7.8, 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 that list decoding is possible according to Proposition A.3 as long as (1) the number of data pairs n is at most $\frac{NL}{2\Lambda_0 S}$ and the number of agreements t is at least $\Theta(\frac{\varepsilon^2 N}{S})$. The claimed list decoding is possible with $R^{\mathsf{RS}} = O(\varepsilon^4/L)$, and the list decoder will return at most $O(L/\varepsilon^2)$ candidates for the control information. Since the list decoding radius δ of LSC was chosen to be $\delta = p + \varepsilon < 1/2 - \varepsilon$, we have $L \leqslant 1/\varepsilon^{O(1)}$ by the guarantee of Proposition 7.2, so the output list size is bounded by a polynomial in $1/\varepsilon$. This proves Lemma 7.4; the claimed failure probability is obtained by adding the probability β that the channel flips more than pN bits, and the failure probability of the sampler from Lemma 7.6.

7.8 Payload Decoding Analysis

We now turn to the analysis of the decoding of the payload codeword and the task of proving Lemma 7.8. We first develop a key tool called the "Hiding Lemma" which will be crucial to our analysis.

7.8.1 The Hiding Lemma

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

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

denote the output of the encoding algorithm when the r_i's are used as the random bits for the LSC encoding. Let $\mathsf{Enc}(m; \pi, T, \cdot)$ be a random encoding of the message m using a given π, T and selecting all other inputs at random.

Definition 11 (Conditional Indistinguishability)**.** *For random variables X, Y, Z with X, Y defined on $\{0,1\}^N$, and $\eta \geqslant 0$, we say that X and Y are* online-space-S indistinguishable given Z with advantage η *if for all values z of Z, and for all randomized online branching programs $\mathcal{A}_z$ (that could depend nonuniformly on z) with N inputs and width 2^S, we have $X \overset{\eta}{\approx}_{\mathcal{A}_z} Y$, where X and Y are conditioned on $Z = z$.* □

The following crucial lemma lets us limit the damage that an online space-bounded channel can cause to our codewords.

Lemma 7.9 (Hiding Lemma)**.** *For all messages m, sampler sets T and permutations π, the random variables $\mathsf{Enc}(m; \pi, T, \cdot)$ and U_N (the uniform distribution on $\{0,1\}^N$) are online-space-$2S$ indistinguishable given (m, π, T) with advantage η, where η is at most $N2^{-2S} + \eta_{\mathsf{Nis}} \leqslant (N+1) \cdot 2^{-2S}$.*

We defer the proof of the above lemma to Section 7.8.2. First, we develop a useful corollary on *error distributions*.

Definition 12 (Error distributions)**.** *Given a randomized channel $\mathcal{A}$ on N bits and a random variable D on $\{0,1\}^N$, let $\mathcal{E}_{\mathcal{A}}(D)$ denote the error distribution of $\mathcal{A}$ on D, that is $D \oplus \mathcal{A}(D)$.*

An important consequence of the Hiding Lemma is that even with the knowledge of π and T, 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 space-bounded tests.

Corollary 7.10 (Errors are Near-Oblivious)**.** *Let W_S be a randomized online-space-S channel. For every m, π, T, the error distributions $\mathcal{E}_{\mathsf{W}_S}(U_N)$ and $\mathcal{E}_{\mathsf{W}_S}(\mathsf{Enc}(m; \pi, T, \cdot))$ are online-space-S indistinguishable given (m, π, T), with advantage at most $(N+1)2^{-2S}$.*

Proof. One can compose a distinguisher for the two error random variables with the channel W_S to get a distinguisher for the original distributions of the Hiding Lemma. This composition can be achieved while maintaining the online restriction and the space usage is the sum of the space of W_S and the distinguisher (that is, at most $2S$). □

7.8.2 Proof of the Hiding Lemma 7.9

Proof of Lemma 7.9. The proof proceeds by hybrid argument. Fix m, π, T, and recall that $|T| = n_{\mathsf{ctrl}}$ is the number of control blocks. Let D_0 be the random variable $\mathsf{Enc}(m; \pi, T, \cdot)$, and D_2 be the uniform distribution over $\{0,1\}^N$. We will define an intermediate random variable D_1, in which the control blocks of D_0 are replaced by fresh uniformly random strings. We show that D_0 and D_2 are both indistinguishable from D_1, and hence from each other.

For notational convenience, suppose that π is the identity permutation, and that the set T, which dictates the locations of the control blocks, occupies the last $\ell = n_{\mathsf{ctrl}}$ locations so that the control information is sent at the end of the codeword (the proof works identically for any other fixed pair T, π). Let the encoding LSC decompose as $[C^*, \mathsf{BPPRG}]$. We can then write D_0 as

$$D_0 = (\text{payload} \ \oplus \Gamma \ \|c_1 \oplus \mathsf{BPPRG}(r_1)\| \cdots \|c_\ell \oplus \mathsf{BPPRG}(r_\ell)) \,,$$

where r_i is the randomness used by the stochastic encoder LSC and $c_1, ..., c_\ell$ are codewords of C^*. Similarly, we can write $D_1 = (\text{payload} \ \oplus \Gamma \ \|U^1\| \cdots \|U^\ell)$.

If ν is the concatenated string $\mathsf{BPPRG}(r_1)\| \cdots \|\mathsf{BPPRG}(r_\ell)$, then conditioned on m, π, T and Γ, any branching program that distinguishes D_1 from D_0 can be used to construct a branching program of the same complexity that distinguishes ν from uniform, by hard-wiring in the string $(\text{payload} \oplus \Gamma \ \|c_1\| \cdots \|c_\ell)$. By the $(2S, 2^{-2S})$-pseudorandom property of LSC and a standard hybrid argument, no online-space-$2S$ branching program can distinguish ν from uniform with advantage better than $\ell \cdot 2^{-2S} \leqslant N2^{-2S}$. Hence D_0 and D_1 are indistinguishable with advantage greater than $N2^{-2S}$.

We now show that D_1 and D_2 are indistinguishable with advantage greater than η_{Nis}. Note that in both distributions, the last ℓ blocks are uniform and independent of the long payload block. A randomized online-space-$2S$ branching program $\mathcal{A}$ that distinguishes D_1 from D_2 with advantage η can be turned into a distinguisher $\mathcal{B}_{\text{payload}}$, defined by $\mathcal{B}_{\text{payload}}(z) = \mathcal{A}(z \oplus \text{payload} \| U^1 \| \cdots U^\ell)$,

that distinguishes Γ from uniform with advantage η. We can represent $\mathcal{B}_{\text{payload}}$ as a randomized online branching program with the same width as $\mathcal{A}$, that is, at most $2S$. By the property of the pseudorandom generator Nis, we conclude that $\mathcal{A}$'s advantage in distinguishing D_1 from D_2 is at most $\eta_{\mathsf{Nis}} \leqslant 2^{-2S}$. □

7.8.3 Proof of Lemma 7.5

Armed with the Hiding Lemma, we return to the task of proving Lemma 7.5 on the claim that the payload decoding succeeds. Recall that for this part, we can assume that the decoder is given the *correct control information* $\omega = (\pi, T, \Gamma)$. We are thus in the *shared randomness* setting.

We will use the Hiding Lemma (actually, Corollary 7.10) 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 against the online-space-S channel W_S. However, the high probability guarantee we will be able to prove is weaker, being $1 - N^{-\Omega(1)}$ when $S = O(\log N)$.

In the following, we fix the message m and the choice T of the control block locations.

Definition 13 (Friendly errors)**.** *For an error vector $e \in \{0,1\}^N$ with Hamming weight at most pN, define e to be* friendly *for (π, T) if the permuted error vector $\pi(e_{|\bar{T}})$ is an error pattern on which the decoder for the code REC succeeds.*

The decoder for REC is the standard "hard decoder" for concatenated codes: it decodes each inner block by brute force to the message in $\{0,1\}^a$ whose encoding by REC^{in} is closest to it, and runs a unique decoder to correct up to a fraction $\kappa = \kappa(\varepsilon)$ of errors for the outer code $\mathsf{REC}^{\text{out}}$. By the linearity of REC, the success of the decoding depends only on the error pattern, and not on the message. *In particular, e is friendly for (π, T) if and only if decoding $\pi(e_{|\bar{T}})$ leads to the all zeroes message.*

The following key lemma says that the error caused by any randomized online-space-S bounded channel is likely to be friendly and thus lead to successful decoding of REC. This lemma immediately implies Lemma 7.5.

Lemma 7.11 (Errors are likely to be friendly)**.** *Let W_S be any randomized online-space-S channel on N bits that causes at most pN errors with probability $1 - \beta$. For all subsets T of control block locations, with probability at least*

$$1 - \beta - N \cdot 2^{-\Omega(\varepsilon^3 S)}$$

(taken over π and the choice of e according to $\mathcal{E}_{\mathsf{W}_S}(\mathsf{Enc}(m; T, \cdot))$), e is friendly for (π, T).

The proof of the above lemma (which appears in Section 7.8.4 below) is one of key difficulties in the analysis compared to the additive errors case. Another principal difference was that we could not argue that the number of payload blocks mistaken for control blocks was small, but allowing for list decoding enabled getting around this difficulty relatively easily.

7.8.4 Proof of Lemma 7.11

Let us denote $\mathsf{W} = \mathsf{W}_S$ for ease of notation. Our plan to prove Lemma 7.11 is the following: By Corollary 7.10 of the Hiding Lemma, we know that $\mathcal{E}_{\mathsf{W}}(\mathsf{Enc}(m; T, \cdot))$ is online-space-S indistinguish-

able (with advantage $(N+1)2^{-2S}$) from $\mathcal{E}_{\mathsf{W}}(U_N)$. The latter error distribution is oblivious to the actual codeword, and so by the analysis of the additive errors case (proof of Theorem 6.1), we know that an error vector distributed according to $\mathcal{E}_{\mathsf{W}}(U_N)$ is friendly for (π, T) with high probability (specifically at least $1 - \beta - \exp(-\Omega_\varepsilon(N/\log N))$, where the β term accounts for the chance that W causes more than pN errors).

If $\mathcal{E}_{\mathsf{W}}(\mathsf{Enc}(m; T, \cdot))$ were *statistically close* to $\mathcal{E}_{\mathsf{W}}(U_N)$, we could conclude that errors distributed according to $\mathcal{E}_{\mathsf{W}}(\mathsf{Enc}(m; T, \cdot))$ are also friendly for (π, T) and we would be done. However, these error distributions are only online-space-S indistinguishable. So in order to apply this style of reasoning, we would need to argue that checking whether an input error vector $e \in \{0,1\}^N$ is friendly for (π, T) can be done by an online-space-S machine (that can depend non-uniformly on π).

This task amounts to checking that $\pi(e_{|\bar{T}})$ decodes to the all zeroes message. Since the decoder for REC corrects a fraction κ of worst-case errors for the outer code $\mathsf{REC}^{\mathrm{out}}$, this condition is met if at most a κ fraction of inner blocks (each with b_{data} bits, corresponding to a codeword of $\mathsf{REC}^{\mathrm{in}}$) are decoded to a non-zero element of $\{0,1\}^a$. This check could potentially be made in low space if we had access to e permuted according to π. Unfortunately, we are only guaranteed indistinguishability against tests with *online* access to e. We do not know of a method for checking friendliness of e for (π, T) in online-space-S.

We therefore resort to a more complicated and indirect argument. We follow ideas from our earlier work [29] on the correction of t-wise independent errors. Specifically, we will show that any particular set of S/b_{data} blocks of the concatenated code behave as they would for binary symmetric errors. We can then use concentration bounds for S/b_{data}-wise independent random variables to argue that decoding succeeds with high probability.

Lemma 7.12. *For every subset P of at most S/b_{data} positions of $\mathsf{REC}^{\mathrm{out}}$, the probability (taken over π and the choice of e according to $\mathcal{E}_{\mathsf{W}}(m; T, \cdot)$) that the inner decodings for $\mathsf{REC}^{\mathrm{in}}$ on $\pi(e_{|\bar{T}})$ return a non-zero element of $\{0,1\}^a$ for every position in P is at most $(\kappa/5)^{|P|} + (N+1)2^{-S}$.*

Proof. Since $|P| b_{\mathsf{data}} \leqslant S = o(N/\log N)$, by the $\Omega(N/\log N)$-wise independence of the permutation π, for each fixed error vector e, and therefore also for e chosen according to $\mathcal{E}_{\mathsf{W}}(U_N)$, the probability that all positions in P are decoded incorrectly is at most $(\kappa/10)^{|P|} + 2^{-|P| b_{\mathsf{data}}} \leqslant (\kappa/5)^{|P|}$. The condition that all inner blocks of $\mathsf{REC}^{\mathrm{in}}$ corresponding to positions in P are incorrectly decoded can be checked by an online branching program of width 2^S, by simply keeping the at most $(S/b_{\mathsf{data}}) \cdot b_{\mathsf{data}} = S$ bits in the blocks corresponding to P in memory. Since $\mathcal{E}_{\mathsf{W}}(m; T, \cdot)$ and $\mathcal{E}_{\mathsf{W}}(U_N)$ are online-space-S indistinguishable with advantage $(N+1)2^{-2S}$, we get the conclusion of the lemma. □

To complete the argument and prove that at most a κ fraction of positions of $\mathsf{REC}^{\mathrm{out}}$ are decoded incorrectly, we need the following probabilistic fact. We include a proof, which is based on ideas used to prove similar statements in [28], for completeness.

Claim 7.1. *Let $\alpha \in (0, 1/3)$ and $X_1, X_2, \ldots, X_n$ be 0-1 valued random variables with $\Pr[X_i = 1] \leqslant \alpha$ for $i = 1, 2, \ldots, n$. Further assume that for every subset $P \subseteq \{1, 2, \ldots, n\}$ of size ℓ,*

$$\Pr\Big[\prod_{i \in P} X_i = 1\Big] \leqslant \alpha_\ell \,. \tag{1}$$

for some $\alpha_\ell \geqslant 0$. Then

$$\Pr\Big[\sum_{i=1}^{n} X_i \geqslant 3\alpha n\Big] \leqslant \alpha_\ell \left(\frac{n-\ell}{3\alpha n-\ell}\right)^{\ell} . \tag{2}$$

Proof. Define $Z = \sum_{i=1}^{n} X_i$ and $S_\ell = \sum_{1\leqslant i_1<i_2<\cdots<i_\ell\leqslant n} X_{i_1}X_{i_2}\cdots X_{i_\ell}$ to be the ℓ'th symmetric function. Since the X_i are 0-1 valued, when $Z = a$, $S_\ell = \binom{a}{\ell}$. We have

$$\Pr[Z \geqslant 3\alpha n] \leqslant \Pr\left[S_\ell \geqslant \binom{3\alpha n}{\ell}\right] \leqslant \frac{\mathbb{E}[S_\ell]}{\binom{3\alpha n}{\ell}}$$

using Markov's inequality. By linearity of expectation and the hypothesis (1), we have $\mathbb{E}[S_\ell] \leqslant \binom{n}{\ell}\alpha_\ell$. Since $3\alpha n \leqslant n$, $\binom{n}{\ell}/\binom{3\alpha n}{\ell} \leqslant \left(\frac{n-\ell}{3\alpha n-\ell}\right)^{\ell}$ and the claim follows. □

Let us now combine Lemma 7.12 and Claim 7.1 applied with the choice $\ell = S/b_{\mathsf{data}}$, $\alpha = \kappa/5 + (N+1)2^{-2S}$, $\alpha_\ell = (\kappa/5)^\ell + (N+1)2^{-2S}$, and $n = n_{\mathsf{data}}$ which is the number of data blocks of the payload codeword (which is also the block length of $\mathsf{REC}^{\mathrm{out}}$).

As $n_{\mathsf{data}} = \Omega(N)$ and $\ell \leqslant S \leqslant o(N/\log N)$, we have $\frac{n_{\mathsf{data}}-\ell}{3\alpha n_{\mathsf{data}}-\ell} \leqslant \frac{1}{2\alpha}$ for large enough N. Also $\alpha_\ell = (\kappa/5)^\ell + (N+1)2^{-2S} \leqslant 2N(\kappa/5)^\ell$ since $\ell \leqslant \varepsilon^2 S$ and $\kappa \geqslant \varepsilon^{O(1)}$. Since $3\alpha = 3\kappa/4 + o_N(1) \leqslant \kappa$, the tail bound (2) implies that the the probability (taken over π and the choice of e according to $\mathcal{E}_{\mathsf{W}}(m;T,\cdot)$) that more than a fraction κ of the inner blocks corresponding to $\mathsf{REC}^{\mathrm{in}}$ are incorrectly decoded is at most

$$\frac{\alpha_\ell}{(2\alpha)^\ell} \leqslant \frac{2N(\kappa/5)^\ell}{2^\ell(\kappa/5+(N+1)2^{-2S})^\ell} \leqslant \frac{2N}{2^\ell} = \frac{2N}{2^{S/b_{\mathsf{data}}}} \leqslant N\cdot 2^{-\Omega(\varepsilon^3 S)} .$$

This finishes the proof of Lemma 7.11, which in turn implies Lemma 7.5 and completes the proof of our main Theorem 7.3 on space-bounded channels.

8 Time-Bounded Channels

The ideas behind our code construction for online space-bounded channels are quite general and can be extended to construct codes against more powerful channels provided we have the necessary explicit pseudorandom generators that can play the role of Nisan's generator for branching programs. In this section, we focus on channels which can be described by polynomial sized circuits. Specifically, we say a channel has circuit size T on inputs of length N if the effect of the channel can be described by a randomized circuit of at most T gates.

Construction. Suppose we desire a code of block length N and rate $1 - H(p) - \varepsilon$ that can be list-decoded errors caused by a channel of circuit size N^c that flips at most pN bits with high probability. We can use a similar construction scheme to the online space-bounded case, with the size of the control blocks $b_{\mathsf{ctrl}} = c' \log N$ for a suitable c' (chosen large enough compared to c), and with the components LSC and the generator Nis changed (as described below) to accommodate the more powerful channel.

The inner code LSC used for the control information encoding will be replaced by a (δ, L)-list-decodable whose codewords are indistinguishable from $U_{b_{\mathsf{ctrl}}}$ with advantage N^{-c} by (randomized) circuits of size $N^{c+\Omega(1)}$. A Monte Carlo construction similar to the one described in Proposition 7.2 can construct such a code with probability $1 - N^{-\Omega(1)}$ in $\mathrm{poly}(N)$ time.

The generator Nis will be replaced by an efficiently computable pseudorandom generator $\mathsf{PolyPRG} : \{0,1\}^{\zeta(\varepsilon)N} \to \{0,1\}^N$ of *constant stretch* such that the output of $\mathsf{PolyPRG}$ fools all circuits C of size $N^{c+\Omega(1)}$; formally

$$\mathsf{PolyPRG}(U_{\zeta N}) \overset{N^{-c}}{\approx}_C U_N \ .$$

Such a pseudorandom generator which is computable in $\mathrm{poly}(N)$ time exists under computational assumptions. For instance, the existence of one-way functions suffices [33, 17], as does the worst-case complexity assumption that $\mathsf{E} \not\subseteq \mathsf{SIZE}(2^{\varepsilon_0 n})$ for some absolute constant $\varepsilon_0 > 0$ [18] where $\mathsf{E} = \mathrm{DTIME}(2^{O(n)})$ and $\mathsf{SIZE}(2^{\varepsilon_0 n})$ denotes the class of languages that have size $O(2^{\varepsilon_0 n})$ circuits.

Decoding algorithm and its analysis. The decoding algorithm is identical to the algorithm described in Section 7.5 for the case of online space-bounded channels.

Turning to the analysis, the part about recovering the control information applies verbatim, and implies that the list decoding of the control information succeeds in finding the correct control information with high probability. An analog of the Hiding Lemma 7.9 (and its Corollary 7.10) where indistinguishability is with respect to size $N^{c+\Omega(1)}$ circuits follows with an identical argument. The analog of Lemma 7.11, which was at the heart of the proof that the payload decoding also succeeds w.h.p., is in fact easier to prove for polynomial-sized circuits and implies that the error vector caused by the size N^c channel is friendly for π with high probability. (The proof is easier since a circuit of some fixed polynomial size can perform the check that an error vector e is friendly for π; recall that this was the difficulty in the online space-bounded case and we required a more complex argument). We can thus prove the following formal statement for coding against channels of polynomial size.

Theorem 8.1. *Assume either* $\mathsf{E} \not\subseteq \mathsf{SIZE}(2^{\varepsilon_0 n})$ *for some* $\varepsilon_0 > 0$ *or the existence of one-way functions. For all constants* $\varepsilon > 0$, $p \in (0, 1/2)$, *and* $c > 1$, *and for infinitely many integers* N, *there exists a Monte Carlo construction (succeeding with probability* $1 - N^{-\Omega(1)}$*) of a stochastic encoder/decoder pair* $(\mathsf{Enc}, \mathsf{Dec})$ *with the following properties:*

- Enc *encodes a message of length* $RN \geqslant (1 - H(p) - \varepsilon)N$ *bits into* N *bits.*
- $(\mathsf{Enc}, \mathsf{Dec})$ *runs in time* $N^{O(c)}$.
- *For every received word* $r \in \{0,1\}^N$, *the decoder* $\mathsf{Dec}(r)$ *outputs a list of at most* $\mathrm{poly}(1/\varepsilon)$ *candidate messages.*
- *For all messages* $m \in \{0,1\}^{RN}$, *and for all randomized channels* W *with circuit size* N^c *(which could depend non-uniformly on* m*) that cause at most* pN *errors with probability* $1 - N^{-\Omega(1)}$, *the list output by the decoder contains* m *with probability at least* $1 - N^{-\Omega(1)}$ *(taken over the stochastic encoding and the channel noise).*

References

[1] R. Ahlswede. Elimination of correlation in random codes for arbitrarily varying channels. *Z. Wahrscheinlichkeitstheorie Verw. Gebiete*, 44:159–175, 1978. 3, 40

[2] N. Alon, L. Babai, and A. Itai. A fast and simple randomized parallel algorithm for the maximal independent set problem. *J. Algorithms*, 7(4):567–583, 1986. 37

[3] M. Bellare and J. Rompel. Randomness-efficient oblivious sampling. In *Proceedings of the 35th Annual IEEE Symposium on Foundations of Computer Science*, pages 276–287, 1994. 18

[4] R. Cramer, Y. Dodis, S. Fehr, C. Padró, and D. Wichs. Detection of algebraic manipulation with applications to robust secret sharing and fuzzy extractors. In *Advances in Cryptology - EUROCRYPT, 27th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 471–488, 2008. 6, 10, 11

[5] I. Csiszár and P. Narayan. Arbitrarily varying channels with constrained inputs and states. *IEEE Transactions on Information Theory*, 34(1):27–34, 1988. 2

[6] I. Csiszár and P. Narayan. The capacity of the arbitrarily varying channel revisited: Positivity, constraints. *IEEE Transactions on Information Theory*, 34(2):181–193, 1988. 5, 17

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

[8] B. K. Dey, S. Jaggi, and M. Langberg. Codes against online adversaries. *CoRR*, abs/0811.2850, 2008. 7

[9] P. Elias. List decoding for noisy channels. *Technical Report 335, Research Laboratory of Electronics, MIT*, 1957. 4

[10] P. Elias. Error-correcting codes for list decoding. *IEEE Transactions on Information Theory*, 37:5–12, 1991. 2, 4

[11] G. D. Forney. *Concatenated Codes*. MIT Press, Cambridge, MA, 1966. 1, 4, 8, 37

[12] Z. Galil, R. J. Lipton, X. Yu, and M. Yung. Computational error-correcting codes achieve shannon's bound explicitly. *Manuscript*, 1995. 3, 5, 6, 7, 20

[13] V. Guruswami. List decoding with side information. In *Proceedings of the 18th IEEE Conference on Computational Complexity (CCC)*, pages 300–309, July 2003. 2, 4

[14] V. Guruswami. *Algorithmic Results in List Decoding*, volume 2 of *Foundations and Trends in Theoretical Computer Science (FnT-TCS)*. NOW publishers, January 2007. 4

[15] V. Guruswami, J. Hastad, M. Sudan, and D. Zuckerman. Combinatorial bounds for list decoding. *IEEE Transactions on Information Theory*, 48(5):1021–1035, 2002. 4, 24

[16] V. Guruswami and M. Sudan. List decoding algorithms for certain concatenated codes. In *Proceedings of the 32nd Annual ACM Symposium on Theory of Computing (STOC)*, pages 181–190, 2000. 24

[17] J. Håstad, R. Impagliazzo, L. A. Levin, and M. Luby. A pseudorandom generator from any one-way function. *SIAM J. Comput.*, 28(4):1364–1396, 1999. 3, 32

[18] R. Impagliazzo and A. Wigderson. P = BPP if e requires exponential circuits: Derandomizing the xor lemma. In *Proceedings of the 29th Annual ACM Symposium on the Theory of Computing*, pages 220–229, 1997. 3, 32

[19] E. Kaplan, M. Naor, and O. Reingold. Derandomized constructions of k-wise (almost) independent permutations. *Electronic Colloquium on Computational Complexity TR06-002*, 2006. 15, 37

[20] M. Langberg. Private codes or succinct random codes that are (almost) perfect. In *Proceedings of the 45th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 325–334, 2004. 4

[21] M. Langberg. Oblivious communication channels and their capacity. *IEEE Transactions on Information Theory*, 54(1):424–429, 2008. 2, 5

[22] A. Lapidoth and P. Narayan. Reliable communication under channel uncertainty. *IEEE Transactions on Information Theory*, 44(6):2148–2177, 1998. 4, 38

[23] R. J. Lipton. A new approach to information theory. In *Proceedings of the 11th Annual Symposium on Theoretical Aspects of Computer Science*, pages 699–708, 1994. 3, 4, 5, 8

[24] S. Micali, C. Peikert, M. Sudan, and D. A. Wilson. Optimal error correction against computationally bounded noise. In *Proceedings of the 2nd Theory of Cryptography Conference*, pages 1–16, 2005. 2, 3, 4, 5

[25] N. Nisan. Pseudorandom generators for space-bounded computation. *Combinatorica*, 12(4):449–461, 1992. 5, 21

[26] N. Nisan and A. Wigderson. Hardness vs randomness. *J. Comput. Syst. Sci.*, 49(2):149–167, 1994. 3

[27] W. W. Peterson. Encoding and error-correction procedures for Bose-Chaudhuri codes. *IEEE Transactions on Information Theory*, 6:459–470, 1960. 36

[28] J. P. Schmidt, A. Siegel, and A. Srinivasan. Chernoff-hoeffding bounds for applications with limited independence. *SIAM J. Discrete Math.*, 8(2):223–250, 1995. 30

[29] A. Smith. Scrambling adversarial errors using few random bits, optimal information reconciliation, and better private codes. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 395–404, 2007. 4, 10, 16, 19, 30, 37

[30] M. Sudan. Decoding of Reed-Solomon codes beyond the error-correction bound. *Journal of Complexity*, 13(1):180–193, 1997. 36

[31] S. P. Vadhan. Constructing locally computable extractors and cryptosystems in the bounded-storage model. *J. Cryptology*, 17(1):43–77, 2004. 15, 36

[32] J. M. Wozencraft. List Decoding. *Quarterly Progress Report, Research Laboratory of Electronics, MIT*, 48:90–95, 1958. 4

[33] A. C.-C. Yao. Theory and applications of trapdoor functions. In *Proceedings of the 23rd IEEE Symposium on Foundations of Computer Science*, pages 80–91, 1982. 3, 32

[34] V. V. Zyablov and M. S. Pinsker. List cascade decoding. *Problems of Information Transmission*, 17(4):29–34, 1981 (in Russian); pp. 236-240 (in English), 1982. 2, 4

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 6.1.

A.1 Constant rate codes for average error

By plugging in an appropriate explicit construction of list-decodable codes (with sub-optimal rate) into Theorem 5.2, 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 5.2 (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 5.2.

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 strongly α-decodable with probability $1 - c_1 2^{-b}$.*

Moreover, for every message and every error pattern of more than a fraction α of errors, the decoder for $\mathsf{SC}_{k,\alpha}$ returns $\perp$ and reports a decoding failure with probability $1 - c_1 2^{-b}$.

Further, 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 5.2. The claim about decoding a uniformly random input follows since the number of strings y which differ from some valid output of the encoder E is at most a fraction α of positions is at most $2^{2b} 2^{H(\alpha) c_0 b}$. By standard entropy arguments, we have $(1 - H(\alpha)) c_0 b + \log(c_1(\alpha)/2) \geqslant 3b$ (since the code encodes $3b$ bits, the capacity is $1 - H(\alpha)$, and at most $\log(c_1(\alpha)/2)$ additional bits of side information are necessary to disambiguate the true message from the list). We conclude that the probability that a random string gets accepted by the decoder is at most $2^{-b} \cdot 2^{\log(c_1(\alpha)/2)} \leqslant c_1 2^{-b}$. □

A.2 Reed-Solomon codes

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

$$\mathsf{RS}_{\mathbb{F},S,n,k}(m) = (f(\alpha_1), f(\alpha_2), \cdots, f(\alpha_n)) \ . \tag{3}$$

where $f(X) = m_0 + m_1 X + ... + m_{k-1} X^{k-1}$. The following is a classic result on unique decoding Reed-Solomon codes [27], 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 $\big|\{i : f(\alpha_i) = a_i\}\big| - \big|\{i : f(\alpha_i) \neq a_i)\}\big| > k$.*

We also state a list-decoding generalization (the version due to Sudan [30] suffices), which will be used in our result for space-bounded channels.

Proposition A.3 (List decoding of RS codes [30])**.** *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 $T \subseteq [N]$ of size ℓ, the density of $B \cap T$ is within $\pm\theta$ of μ with overwhelming probability (at least $1 - \exp(-c_\theta \ell)$). But picking a random subset of size ℓ requires $\approx \ell \log(N/\ell)$ random bits. The following shows that a similar effect can be achieved by a sampling procedure that uses fewer random bits. The idea is the well known one of using random walks of length ℓ in a low-degree expander on N vertices. This could lead to repeated samples while we would like ℓ distinct samples. This can be achieved by picking slightly more than ℓ samples and discarding the repeated ones. The result below appears in this form as Lemma 8.2 in [31].

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 14. *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.5 ([19])**.** *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 15. *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 [2], but we can work with the weaker $O(t \log n)$ seed length.

Proposition A.6. *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 [11] constructed binary linear concatenated codes that achieve the capacity of the binary symmetric channel BSC_p. Smith [29] 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.7. *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 [22] 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 16 (Codes for average error)**.** *A code C with encoding function $\mathcal{E} : \mathcal{M} \to \Sigma^n$ is said to be (efficiently) p-*decodable with average error δ *if there is a (polynomial time computable) decoding function $D : \Sigma^n \to \mathcal{M} \cup \{\perp\}$ such that for* every *error vector $e \in \Sigma^n$, the following holds for at least a fraction $(1-\delta)$ of messages $m \in \mathcal{M}$: $D(\mathcal{E}(m) + e) = m$.* □

B.1 Codes for average error from stochastic codes for additive errors

A slightly more general notion of the p-decodable stochastic codes from Definition 3 implies codes for average error.

Definition 17 (Strongly decodable stochastic codes)**.** *For a code as in Definition 3, if the decoding function correctly computes in addition to the message m also the randomness ω used at the encoder with probability at least $1-\delta$, the we say that the stochastic code is strongly p-decodable with probability $1-\delta$.* □

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

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

B.2 Explicit capacity-achieving codes for average error

We would now like to apply Observation B.1 to the stochastic codes constructed in Section 6 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 6.5) 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 12. 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 6.4, 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 6.5). Also r is the unique string which 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 16).

Lemma B.2 (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 $(r_1, ..., r_\ell)$ 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 6.3 (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 $r_1, ..., r_\ell$. We can thus prove the following:

Theorem B.3 (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 show 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 $p > 1/4$.

Theorem C.1 (Impossibility for $p > \frac{1}{4}$, detailed version)**.** *For every pair of randomized encoding/decoding algorithms $\mathsf{Enc}, \mathsf{Dec}$ that make n uses of the channel and use a message space whose size tends to infinity with n, if a uniformly random message is sent over the channel, then*

1. *there is a distribution over memoryless channels that alters at most $n/4$ bits* in expectation *and causes a decoding error 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 with probability $\Omega(\nu)$.*

Our proof adapts the impossibility results of Ahlswede [1] 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 [1].

The Swapping Channel. We begin by considering a simple *swapping* channel, whose behavior is specified by a *state* vector $s = (s_1, ..., s_n) \in \{0,1\}^n$. On input a transmitted word $c = (c_1, ..., c_n) \in \{0,1\}^n$, 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,

$$\mathsf{W}_s(c) \text{ and } \mathsf{W}_c(s) \text{are identically distributed} \tag{4}$$

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 $\text{dist}(c, s)$; specifically, the number of errors is distributed as $\text{Binomial}(\text{dist}(c, s), \frac{1}{2})$. As long as $\text{dist}(c, s)$ 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.2. *Under the conditions of Theorem C.1, 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:

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

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.1 (Plotkin)**.** *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 C.1, 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 $\mathrm{Binomial}(n(\frac{1}{2}+\nu), \frac{1}{2})$. 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 C.1.

ECCC ISSN 1433-8092

http://eccc.hpi-web.de