Electronic Colloquium on Computational Complexity, Revision 2 of Report No. 47 (2020)

Explicit Uniquely Decodable Codes for Space Bounded Channels That Achieve List-Decoding Capacity

Ronen Shaltiel* Jad Silbak†

November 16, 2020

Abstract

We consider codes for space bounded channels. This is a model for communication under noise that was introduced by Guruswami and Smith (J. ACM 2016) and lies between the Shannon (random) and Hamming (adversarial) models. In this model, a channel is a space bounded procedure that reads the codeword in one pass, and modifies at most a p fraction of the bits of the codeword.

- *Explicit uniquely decodable codes for space bounded channels:* Our main result is that for every $0 \leq p < \frac{1}{4}$, there exists a constant $\delta > 0$ and a *uniquely decodable* code with rate $1 - H(p)$ for channels with space n^{δ}. This code is *explicit* (meaning that encoding and decoding are in poly-time). This improves upon previous explicit codes by Guruswami and Smith, and Kopparty, Shaltiel and Silbak (FOCS 2019). Specifically, we obtain the same space and rate as earlier works, even though prior work gave only *list-decodable* codes (rather than uniquely decodable codes).
- *Complete characterization of the capacity of space bounded channels:* Together with a result by Guruswami and Smith showing the impossibility of unique decoding for $p \geq \frac{1}{4}$, our techniques also give a complete characterization of the capacity $R(p)$ of space $n^{1-o(1)}$ channels, specifically:

$$R(p) = \begin{cases} 1 - H(p) & 0 \leq p < 1/4 \\ 0 & p \geq 1/4. \end{cases}$$

  This capacity is strictly larger than the capacity of Hamming channels for every $0 < p < \frac{1}{4}$, and matches the capacity of list decoding, and binary symmetric channels in this range. Curiously, this shows that $R(\cdot)$ is not continuous at $p = 1/4$.

Our results are incomparable to recent work on casual channels (these are stronger channels that read the codeword in one pass, but there is no space restriction). The best known codes for casual channels, due to Chen, Jaggi and Langberg (STOC 2015), are shown to exist by the probabilistic method, and no explicit codes are known.

A key new ingredient in our construction is a new notion of "*evasiveness*" of codes, which is concerned with whether a decoding algorithm rejects a word that is obtained when a channel induces few errors to a *uniformly chosen* (or *pseudorandom*) string. We use evasiveness (as well as several additional new ideas related to coding theory and pseudorandomness) to identify the "correct" message in the list obtained by previous list-decoding algorithms.

*Department of Computer Science. University of Haifa, Email: ronen@cs.haifa.ac.il. Supported by ISF grant 1628/17.

†School of Computer Science, Tel Aviv University. Email: jadsilbak@mail.tau.ac.il. Supported by ERC starting grant 638121 and ISF grant 1628/17

ISSN 1433-8092

Contents

1 Introduction

1.1 Codes and channels

Coding theory studies transmission of messages using noisy channels. In this paper we are interested in binary codes, and prefer to focus on decoding properties of a code, rather than combinatorial properties like minimal distance. More specifically, given a family $\mathcal{C}$ of (possibly randomized) functions $C : \{0,1\}^n \to \{0,1\}^n$ (which we call "channels") the goal is to design a code (namely, a pair $(\mathrm{Enc}, \mathrm{Dec})$ of an encoding map $\mathrm{Enc} : \{0,1\}^k \to \{0,1\}^n$ and a decoding map $\mathrm{Dec} : \{0,1\}^n \to \{0,1\}^k$) such that for every message $m \in \{0,1\}^k$ and every channel $C \in \mathcal{C}$, decoding is successful, namely:

$$\mathrm{Dec}(\mathrm{Enc}(m) \oplus C(\mathrm{Enc}(m))) = m.$$

The rate of a code is $R = \frac{k}{n}$. For a family $\mathcal{C}$ of channels, we use $R(\mathcal{C})$ to denote the capacity of the family, which is the best possible rate of a code for this family.[1] For a family $\mathcal{C}$ of channels, there are two main goals:

1. Determine the capacity $R(\mathcal{C})$.
2. Construct explicit codes (namely codes with poly-time encoding and decoding algorithms).

Let us review some coding scenarios and channel families. In all examples below $0 \leq p < \frac{1}{2}$ is a parameter.

Binary symmetric channels. A binary symmetric channel (denoted by BSC_p) is the randomized function that ignores its input and produces n i.i.d. random bits, where each of them is one with probability p. This is a special case of an extensively studied class of randomized channels (often referred to as "Shannon channels"). A celebrated theorem of Shannon shows that $R(\mathrm{BSC}_p) = 1 - H(p)$.[2] Later work on code concatenation (due to Forney [For65]) produced codes with explicit and even linear time algorithms [GI05].

Hamming channels. The class of Hamming channels (denoted by Ham_p) is the class of all functions such that for every input x, the relative Hamming weight of $C(x)$ is at most p.[3] This is probably the most studied class of channels, and yet, its capacity $R(\mathrm{Ham}_p)$ is not precisely understood. It is known that $R(\mathrm{Ham}_p) = 0$ for $p \geq \frac{1}{4}$, and that for $0 < p < \frac{1}{4}$, $R(\mathrm{Ham}_p) < 1 - H(p)$.[4] The Gilbert-Varshamov bound shows that $R(\mathrm{Ham}_p) \geq 1 - H(2p)$, but explicit codes with this rate are unknown.

List-decoding. In the relaxed goal of list-decoding, the decoding map is allowed to output a list of $L = O(1)$ messages, and decoding is considered successful if $\mathrm{Dec}(\mathrm{Enc}(m) \oplus C(\mathrm{Enc}(m))) \ni m$. Unlike the case of unique decoding, the list decoding capacity of Hamming channels (denoted by $R^{\mathrm{List}}(\mathrm{Ham}_p)$) is known to be $R^{\mathrm{List}}(\mathrm{Ham}_p) = 1 - H(p)$,[5] which allows positive rate even for $\frac{1}{4} \leq p < \frac{1}{2}$ (in contrast to unique decoding). Explicit constructions of such codes are unknown.

[1] More formally, $R(\mathcal{C})$ is the largest number R such that for every $\epsilon > 0$, there exist infinitely many n, for which there exists a code $\mathrm{Enc} : \{0,1\}^k \to \{0,1\}^n$ for $\mathcal{C}$, with rate at least $R - \epsilon$. We mostly use the term "rate" for a specific (family of) codes, and "capacity" for a class of channels, but these terms are interchangeable in this paper.

[2] Here $H(p) = p \cdot \log(1/p) + (1-p) \cdot \log(1/(1-p))$ is Shannon's entropy function.

[3] The relative Hamming weight of a string $z \in \{0,1\}^n$ is $wt(z) = \frac{|\{i \in [n] : z_i \neq 0\}|}{n}$.

[4] This follows because by the Elias-Bassalygo bound, which states that $R(\mathrm{Ham}_p) < R_{\text{Elias-Bassalygo}}(p)$ where the latter is strictly smaller than $1 - H(p)$. We remark that the Elias-Bassalygo bound gives a stronger result, and that later work by McEliece, Rodemich, Rumsey and Welch [MRRW77] improves this bound in some ranges. We state the bound $R < 1 - H(p)$ to stress that $R(\mathrm{Ham}_p) < R(\mathrm{BSC}_p) = 1 - H(p)$.

[5] This formally means that for every $\epsilon > 0$, there exists a constant L_ϵ such that there are infinitely many n, for which there exists a code with rate $R = 1 - H(p) - \epsilon$ and a list-decoding map with list size L_ϵ.

Intermediate classes of channels. It is natural to consider intermediate classes of channels that lie between binary symmetric channels and Hamming channels. One such example was studied by Guruswami and Smith [GS16] that considered the class of additive channels. This class (denoted by Add_p) contains all *constant* functions $C \in \mathrm{Ham}_p$. This means that an additive channel $C : \{0,1\}^n \to \{0,1\}^n$ has a predetermined noise vector $e \in \{0,1\}^n$ of Hamming weight at most p, and the channel C uses this noise vector *regardless* of its input. In particular, the channel does not choose the noise vector as a function of the transmitted codeword.

It turns out that with the standard definition of codes, every code for additive channels is also a code for Hamming channels.[6] In order to take advantage of restricted families of channels, one needs to consider a different coding scenario. Several such scenarios were considered in the literature, see Section 1.4 on related work. In this paper, we follow the approach of Guruswami and Smith [GS16] and consider *stochastic codes*.

Stochastic codes. These are codes where the encoding algorithm is randomized, and decoding only needs to succeed with high probability. More precisely, an encoding map of a stochastic code, is a function $\mathrm{Enc} : \{0,1\}^k \times \{0,1\}^d \to \{0,1\}^n$ and it is required that for every $m \in \{0,1\}^k$, and every channel C in the considered class:

$$\Pr_{S \leftarrow U_d}[\mathrm{Dec}(\mathrm{Enc}(m,S) \oplus C(\mathrm{Enc}(m,S)) = m] \geq 1 - \nu,$$

where ν is an error parameter. (A precise formal definition is given in Definition 2.17). Note that the decoding algorithm *does not* need to receive S, and so, these codes can be used in the standard coding communication scenario. The rate of a stochastic code is $R = \frac{k}{n}$. Stochastic codes do not give an improvement in capacity in the case of Hamming channels (as it is easy to show that a stochastic code for Hamming channels yields a standard code with the same rate) but they do allow improved capacities for other classes.

Additive channels. Recall that an additive channel $C \in \mathrm{Add}_p$ is a channel that ignores the transmitted codeword, and always uses a predetermined noise vector. Guruswami and Smith [GS16] showed that the *stochastic capacity*, $R^{\mathrm{Stoc}}(\mathrm{Add}_p) = 1 - H(p)$, while also providing explicit encoding and decoding algorithms for the stochastic code. Jumping ahead, we mention that prior to our work, the (weak) class of additive channels is the strongest class for which explicit stochastic codes with rate $1 - H(p)$ are known. All the channel classes that we consider below are stronger (and can simulate additive channels). Therefore, an upper bound of $R \leq 1 - H(p)$ on the stochastic capacity holds for all such classes.

Space bounded channels. Guruswami and Smith [GS16], and later work [SS16, KSS19] also considered space bounded channels (sometimes called "online channels"). The class of space s channels (denoted by Spc_p^s) is the class of all functions $C : \{0,1\}^n \to \{0,1\}^n$ such that $C \in \mathrm{Ham}_p$, and furthermore, C can be implemented by a space s procedure that reads its input in one pass. More precisely, C is a procedure that reads its input in one pass, and maintains a state of at most s bits. Whenever the channel reads an input bit, it updates its internal state, and produces an output bit. Both actions are done as a function of the previous internal state and the read input bit. (A precise definition is given in Definition 2.3.)

Note that unlike additive channels or binary symmetric channels, this allows channels to choose the error pattern as a function of the transmitted codeword. Additive channels are a subclass of space zero channels.

Guruswami and Smith [GS16] showed that $R^{\mathrm{Stoc}}(\mathrm{Spc}_p^{\log n}) = 0$, for $p > \frac{1}{4}$. This means that (similar to the case of standard codes for Hamming channels) unique decoding is impossible for $p > \frac{1}{4}$.

Previous works on explicit stochastic codes for space bounded channels [GS16, SS16, KSS19] did not achieve *unique decoding* and settled for *list-decoding*. Kopparty, Shaltiel and Silbak [KSS19] (building on [GS16, SS16]) show that for every $0 \leq p < \frac{1}{2}$, there exists a constant $\delta > 0$ such that $R^{\mathrm{List,Stoc}}(\mathrm{Spc}_p^{n^\delta}) = 1 - H(p)$. Furthermore, this is achieved by explicit encoding and list-decoding algorithms.

[6] This follows as if there is a message $m \in \{0,1\}^k$ and a channel $C \in \mathrm{Ham}_p$ such that $\mathrm{Dec}(\mathrm{Enc}(m) \oplus C(\mathrm{Enc}(m))) \neq m$, then the channel $C'(x) = C(\mathrm{Enc}(m))$ is a channel in Add_p on which decoding is not successful.

Casual channels. The class of casual channels (denoted by Cas_p) is Spc_p^n, namely the class of all channels that read the codeword in one pass, but have no space restriction. No nontrivial explicit codes (even for the case of list-decoding) are known for casual channels. Chen, Jaggi and Langberg [CJL15] used the probabilistic method to show existence of codes that match earlier upper bounds by Dey, Jaggi, Langberg and Sarwate [DJLS13]. Together, this gives that $R^{\mathrm{Stoc}}(\mathrm{Cas}_p) = f(p)$ for:

$$f(p) = \begin{cases} \min_{\overline{p}\in[0,p]}[(1-4(p-\overline{p}))(1-H(\frac{\overline{p}}{(1-4(p-\overline{p}))}))] & 0 \le p < 1/4 \\ 0 & p \ge 1/4. \end{cases}$$

It is known that there exists a number $p_0 \approx 0.0804$ such that $f(p) = 1 - H(p)$ for $p < p_0$, and $f(p) < 1 - H(p)$ for $p > p_0$. This means that for $p > p_0$, $R^{\mathrm{Stoc}}(\mathrm{Cas}_p) < 1 - H(p)$.

We summarize all theses surveyed results (as well as our new results) in Table 1. In the case of results that produce codes, we list whether or not the codes are explicit.

Table 1: Summary of surveyed known results. The results of this paper appear in bold text

Channel	Decoding	Stochastic?	Range	Rate	Explicit?	Reference
BSC_p	Unique	No	$0 \le p < \frac{1}{2}$	$R = 1 - H(p)$	Yes	[For65]
Ham_p	Unique	No	$0 \le p < \frac{1}{4}$	$R < 1 - H(p)$	N/A	[MRRW77]
Ham_p	Unique	No	$\frac{1}{4} \le p < \frac{1}{2}$	$R = 0$	N/A	Easy
Ham_p	List	No	$0 \le p < \frac{1}{2}$	$R = 1 - H(p)$	No	Easy
Add_p	Unique	Yes	$0 \le p < \frac{1}{2}$	$R = 1 - H(p)$	Yes	[GS16]
$\mathrm{Spc}_p^{\log n}$	Unique	Yes	$\frac{1}{4} < p < \frac{1}{2}$	$R = 0$	N/A	[GS16]
$\mathrm{Spc}_p^{n^{\Theta(1)}}$	List	Yes	$0 \le p < \frac{1}{2}$	$R = 1 - H(p)$	Yes	[KSS19]
$\mathrm{Cas}_p = \mathrm{Spc}_p^n$	Unique	Yes	$0 \le p < 0.0804$	$R = 1 - H(p)$	No	[CJL15]
$\mathrm{Cas}_p = \mathrm{Spc}_p^n$	Unique	Yes	$0.0804 < p < \frac{1}{4}$	$R < 1 - H(p)$	No	[DJLS13]
$\mathbf{Spc_p^{n^{\Theta(1)}}}$	**Unique**	**Yes**	$\mathbf{0 \le p < \frac{1}{4}}$	$\mathbf{R = 1 - H(p)}$	**Yes**	**Here**
$\mathbf{Spc_p^{n^{1-o(1)}}}$	**Unique**	**Yes**	$\mathbf{0 \le p < \frac{1}{4}}$	$\mathbf{R = 1 - H(p)}$	**No**	**Here**

1.2 Our results

In this paper we study *uniquely decodable* stochastic codes for space bounded channels. Previous work on this class considered *list-decoding*, and the best known lower bound on the capacity of uniquely decodable codes is given by non-explicit codes for casual channels. Natural questions are:

1. What is the *unique decoding* capacity of space bounded channels? Is it larger than that of casual channels?
2. Can we explicitly construct *uniquely decodable* stochastic codes matching this capacity with poly-time encoding and decoding?

In this paper we answer both questions affirmatively, and show that:

1. The unique decoding capacity for space bounded channels (with space as high as $n^{1-o(1)}$) is $1 - H(p)$, matching the capacity of binary symmetric channels (and list-decoding) for $0 \le p < \frac{1}{4}$. This capacity is strictly larger than the capacity of casual channels for $p > p_0 \approx 0.0804$. Curiously, this shows that:
    - The capacity of space bounded channels is not continuous at $p = \frac{1}{4}$.

- Unlike casual channels which do not achieve rate $1 - H(p)$ for $p > p_0$, space bounded channels are "better behaved" and achieve rate $1 - H(p)$ all the way up to $\frac{1}{4}$.

2. More importantly, we can achieve this capacity with *explicit* stochastic codes for slightly smaller space of $s = n^{\Theta(1)}$ (which is exactly the same space bound considered in previous explicit constructions of *list-decodable* codes for space bounded channels).

Perspective: Prior to this work, the best explicit *uniquely decodable* stochastic code with rate $1 - H(p)$ could handle *additive channels* [GS16] (which do not get to look at the transmission). This work gives an explicit, uniquely decodable code for the much stronger class of space bounded channels.

It was pointed out by Guruswami and Smith [GS16] that many of the "stochastic channels" studied in Shannon's framework are captured by space bounded channels (in fact even with space $O(\log n)$). Our results gives explicit codes in all these cases.[7]

On a more philosophical level, one may postulate that the behavior of most conceivable channels that are not "fully adversarial" is captured by space bounded channels, which can now be explicitly uniquely decoded with rate $1 - H(p)$ that matches that of binary symmetric channels for $p < \frac{1}{4}$. This rate beats the best possible rate for uniquely decodable codes for Hamming channels or casual channels (even if one is satisfied with non-explicit codes).

Prior to this work, explicit stochastic codes [GS16, SS16, KSS19] for space bounded channels only achieved *list-decoding* rather than *unique decoding*. We stress that there is no difference in list-decoding capacities between Hamming channels and space bounded channels. In contrast, in the more natural and standard case of *uniquely-decodable codes* there is a difference between the capacities of Hamming channels and space bounded channels, and our codes achieve rate of $1 - H(p)$, which is impossible to achieve in the case of uniquely decodable codes for Hamming channels.
Our results are listed in detail below, together with a comparison to relevant previous work.

1.2.1 Explicit uniquely decodable codes for space bounded channels with rate $1 - H(p)$

The main result of this paper is that for every $0 \leq p < \frac{1}{4}$, there exists $\delta > 0$, and an *explicit uniquely decodable* stochastic code for $\mathrm{Spc}_p^{n^\delta}$ with rate $1 - H(p)$.

Theorem 1.1 (Explicit uniquely decodable codes with optimal rate for $\mathrm{Spc}_p^{n^\delta}$)**.** *For every constant $0 \leq p < \frac{1}{4}$ there exists a constant $\delta > 0$ such that for every constant $\epsilon > 0$, for infinitely many n, there is a stochastic code $(\mathrm{Enc}, \mathrm{Dec})$ for $\mathrm{Spc}_p^{n^\delta}$, with rate $R = 1 - H(p) - \epsilon$, and success probability $1 - \nu$ for $\nu = 2^{-polylog(n)}$. Furthermore, Enc and Dec run in time $\mathrm{poly}(n)$.*

Theorem 1.1 is stated in a more general way in Theorem 5.1. Theorem 1.1 achieves the same rate and same space bound as the results of [KSS19] with the significant advantage of achieving *unique decoding* rather than *list-decoding*. In both cases, the constant $\delta = c \cdot ((\frac{1}{4} - p)^2)$ for some universal constant $c > 0$. Our result does not completely subsume the list-decoding of [KSS19] because the latter has three additional advantages: It works for $\frac{1}{4} \leq p < \frac{1}{2}$ which is impossible for unique decoding. It works against channels that can select the order in which they read the bits of the codeword, whereas our result does not. The running time of encoding and decoding is quasilinear (namely $T(n) = n \cdot \mathrm{polylog}(n)$). Using optimized components in our construction, it is possible to achieve time $T(n) = n^{1+o(1)}$ in Theorem 1.1, but we are not sure whether some of the components can be made to run in quasilinear time (and we plan to look into that in future versions). See remark 2.13 and Section 5.6.

[7] Our results clearly extend to any channel that is a convex combination of space $s = n^{\Theta(1)}$ channels. Furthermore, with an additional $\log n$ space, a channel can count the number of error that it induces, and avoid inducing more than pn errors. This means that our theorems handle any distribution over space s channels, in which the probability of inducing significantly more than pn errors is small.

1.2.2 The capacity of space bounded channels

Before our work, it was known that the capacity $R(p)$ of space bounded channels is zero for $p \geq \frac{1}{4}$ [GS16], and at most $1 - H(p)$ for $0 \leq p < \frac{1}{4}$. Theorem 1.1 implies that for space $s = n^{\theta(1)}$ the capacity is:

$$R(p) = \begin{cases} 1 - H(p) & 0 \leq p < 1/4 \\ 0 & p \geq 1/4. \end{cases}$$

This capacity is larger than that of $\mathrm{Cas}_p = \mathrm{Spc}_p^n$ for $p > p_0 \approx 0.0804$. A natural question is whether this capacity applies also for channels with space s that approaches n. The next theorem shows that the capacity is $1 - H(p)$ even for space $s = n^{1-o(1)}$.

Theorem 1.2 (The capacity of space $n^{1-o(1)}$ channels)**.** *There exists a function $s = n^{1-o(1)}$ such that for every constant $0 \leq p < \frac{1}{4}$ and every constant $\epsilon > 0$, for infinitely many n, there is a stochastic code $(\mathrm{Enc}, \mathrm{Dec})$ for Spc_p^s, with rate $R = 1 - H(p) - \epsilon$, and success probability $1 - \nu$ for $\nu = 2^{-\mathit{polylog}(n)}$.*

In many cases, existence of good codes follows by a simple analysis of a random code. In the case of space bounded channels, it is not immediately clear how to analyze a random stochastic code. Theorem 1.2 follows by using the construction used for Theorem 1.1, and replacing a certain explicit code with one that is shown to exist by the probabilistic method. An advantage of this approach is that it gives a strategy to constructing explicit codes for larger space.[8] We remark that the space bound s in Theorem 1.2 can be pushed to $s = n/\mathrm{polylog}(n)$ if we allow more components in our construction to be nonexplicit. See Section 5.5.

Summing up we have that for every $0 \leq p < \frac{1}{4}$:

$$R^{\mathrm{Stoc}}(\mathrm{Spc}_p^{n^{1-o(1)}}) = 1 - H(p) = R(\mathrm{BSC}_p) = R^{\mathrm{List}}(\mathrm{Ham}_p).$$

This shows a separation between casual channels and space bounded channels with space $n^{1-o(1)}$.

Other tradeoffs in explicit codes. We have chosen the space and success probability in Theorem 1.1 to match the parameters of [KSS19]. However, other tradeoffs can also be achieved. Specifically:

- For every $p < \frac{1}{8}$ the space in Theorem 1.1 can be increased to $s = n^{\frac{1}{2}-o(1)}$.
- In [KSS19], the error parameter ν was chosen to be $2^{-\mathrm{polylog}(n)}$. This is because the construction has steps that inherently run in time $n \cdot \log(1/\nu)$, and it was desired to get time $n \cdot \mathrm{polylog}(n)$. However, it is also possible to obtain much smaller ν. For example, it is possible to obtain $\nu = 2^{-n^{\Omega(1)}}$.
- We can get explicit codes for channels with larger space (specifically, space $n^{1-o(1)}$), for $0 < p \leq p_1$ for some constant $p_1 > 0$.

1.3 Overview of the technique

Our construction relies on the recent construction of *list-decodable* stochastic code for space bounded channels by Kopparty, Shaltiel and Silbak [KSS19] (which in turn builds on the approach of Guruswami and Smith [GS16]). The high level idea is that in order to obtain a *unique* decoding algorithm, we will first apply the list

[8] More specifically, the component that is needed is a linear code $\mathrm{Enc} : \{0,1\}^k \to \{0,1\}^n$ that has distance approaching $\frac{1}{2}$, its dual has distance r, and Enc has explicit list decoding from slightly less than $\frac{1}{2}$ relative errors. Loosely speaking, our construction can take such a code and transform it into a stochastic code for space $s = r^{1-o(1)}$ channels. The current best explicit construction of Kopparty, Shaltiel and Silbak [KSS19] shows that for every $p < \frac{1}{4}$, there exists $\delta > 0$ and a code with $r = n^{\delta}$, and this translates into the final parameters of Theorem 1.1. If one gives up explicit list-decoding, then by standard calculations, it is easy to show existence of such codes that give $s = n^{1-o(1)}$ and imply Theorem 1.2. See discussion in Section 7.

decoding algorithm of [KSS19] (which produces a constant size list of messages) and then apply a pruning procedure which will identify the correct message. (Note that while this approach is very natural, it cannot work for Hamming channels).

In order to distinguish the correct message in the list from incorrect ones, we introduce a new concept of "evasiveness" of codes. Specifically, in the case of standard (that is non-stochastic) codes for binary symmetric channels, we will be interested in whether a space bounded channel that gets access to a *uniform* string $Z \leftarrow U_n$, can induce pn errors in a way that will lead the decoding algorithm to correctly decode. We will say that the channel "wins" if the decoding algorithm correctly decodes.

Our first step is constructing explicit evasive code with rate $1 - H(p)$ (and our final stochastic codes will inherit this rate). This argument is outlined in Sections 1.3.1, 1.3.2 and the formal treatment is in Section 3.

We then want to argue that a space bounded channel cannot win even in an experiment where it receives a string Z that was generated by a pseudorandom generator that fools small space algorithms (rather than a uniform string $Z \leftarrow U_n$). This does not follow immediately from the pseudorandomness of the generator, because whether or not the channel wins is also determined by the actions of the decoding algorithm that does not run in small space. Nevertheless, by using recent constructions of locally-correctable and locally-testable codes by Guo, Kopparty and Sudan [GKS13] and Kopparty, Meir, Ron-Zewi and Saraf [KMRS17] we are able to construct codes that are also evasive on pseudorandom strings. This argument is outlined in Section 1.3.3 and the formal treatment is in Section 3.

Our next step is to modify the construction of Kopparty, Shaltiel and Silbak [KSS19] hoping to argue that when list-decoding, all incorrect messages in the list were obtained in a scenario that corresponds to the evasiveness experiment on pseudorandom strings, and are therefore rejected. This argument is outlined in Sections 1.3.4, 1.3.5, and the formal treatment is in Section 5 and 6.

In the remainder of this section we give a more detailed high level overview of the ideas and techniques that we use. We allow ourselves to be informal and imprecise (in order to highlight the main ideas). The reader can safely skip this section, as complete definitions, theorem statements and proofs, appear in later sections (and these do not rely on the informal description given in this section).

1.3.1 Evasive codes for space bounded channels

A key new ingredient in our construction is a notion of "evasive codes" that we now explain. Let $\mathrm{Enc} : \{0,1\}^k \to \{0,1\}^n$ and $\mathrm{Dec} : \{0,1\}^n \to \{0,1\}^k$ be a code where Dec is a decoding algorithm for some communication scenario. To make things less abstract, let us start by focusing on the case that Dec decodes from BSC_p, and rejects words that have relative distance slightly more than p from any codeword. The evasiveness experiment is concerned with corrupting and decoding a *uniform string* (rather than a codeword):

The evasiveness experiment: Given a channel C that induces at most p relative errors:

- Choose uniform $Z \leftarrow U_n$.
- Treat Z as a "codeword". That is, let $V = Z \oplus C(Z)$ be corrupted by the channel C.
- The channel C "wins" if $\mathrm{Dec}(V)$ does not reject.

We say that a code is *evasive* for C if the probability that C wins is small.[9] In this paper we design codes for BSC_p that have rate $1 - H(p)$, and are evasive against space $s = o(n)$ channels. This is stated informally below (and the formal statement and treatment is in Section 3).

[9] A straightforward observation is that any code with rate R slightly smaller than $1 - H(2p)$ (for example, a code with $R = 1 - H(2p) - \epsilon$ that match the Gilbert-Varshamov bound) is evasive against the class of *all* channels $C \in \mathrm{Ham}_p$. This is because w.h.p., a uniform $Z \leftarrow U_n$ has distance larger than $2p$ from any codeword. Consequently, no matter how the unbounded channel C uses its ability to inject p relative errors, it cannot make $V = Z \oplus C(Z)$ be within distance p to a codeword.

Informal Theorem 1.3 (Evasive codes for BSC_p)**.** *For every $p < \frac{1}{2}$, there are explicit codes for BSC_p with rate $1 - H(p)$ that are evasive against Spc_p^s for $s = o(n)$.*

These codes are an important ingredient in our construction of stochastic codes for space bounded channels. Our final code inherits the rate of these codes.

1.3.2 Arguing that codes are evasive

We now outline the proof of informal theorem 1.3. A standard way to construct codes for BSC_p is to concatenate a rate $1 - \epsilon$ outer code over constant size alphabet that corrects from a constant $\lambda = \lambda(\epsilon) > 0$ fraction of errors, with a constant length inner code with rate $1 - H(p) - \epsilon$ that decodes from BSC_p. The concatenated code has rate approaching $1 - H(p)$ and decodes from errors induced by BSC_p

We are interested in showing that this concatenated code is evasive against space bounded channels. It turns out that for this, it is sufficient to focus on the outer code, and show that it is evasive. Specifically, it will be sufficient to prove:

Informal Theorem 1.4 (Evasive codes with high rate)**.** *For every $p < \frac{1}{2}$ and every $\epsilon > 0$, there exists $\lambda > 0$ and explicit codes with constant size alphabet, rate $1 - \epsilon$ that decode from λ relative errors, and are evasive for Spc_p^s for $s = o(n)$.*

We will now focus on this task. We consider a code $\mathrm{Enc} : \Sigma^k \to \Sigma^n$, where we take some slack, and assume that there is a decoding algorithm $\mathrm{Dec}_{2\lambda}$ that decodes from 2λ relative errors (rather than just λ relative errors). This makes no difference in our setup where λ is small. We consider a modified version Dec of the decoding algorithm $\mathrm{Dec}_{2\lambda}$, that rejects an input v, if the relative Hamming distance between $\mathrm{Enc}(\mathrm{Dec}_{2\lambda}(v))$ and v is larger than λ. We can safely reject such strings, as we only want Dec to decode from λ relative errors.

We will show that Enc is evasive (using the decoding algorithm Dec) if $p < \frac{1}{2} - 2\lambda$ (which we can arrange). We will divide the n symbols of a string in Σ^n into $u = O(1)$ blocks of length $\ell = n/u$, we will choose the parameters so that $s \ll \ell \leq \lambda \cdot n$. We will denote the i'th block of a string $z \in \Sigma^n$ by $z[i]$.

For every channel $C \in \mathrm{Spc}_p^s$, the evasiveness experiment considers $Z \leftarrow U_n$, $V = Z \oplus C(Z)$, and $Y = \mathrm{Enc}(\mathrm{Dec}_{2\lambda}(V))$. Our goal is to show that w.h.p. the relative Hamming distance between Y and V is at least λ. For this purpose we consider the following mental experiment: For every $i \in [u]$, we consider the random variable $V^{(i)}$ which is obtained by taking V and replacing the i'th block $V[i]$ with a constant string (say a sequence of zeros). Let $Y^{(i)} = \mathrm{Enc}(\mathrm{Dec}_{2\lambda}(V^{(i)}))$.

A weakness of a space s channel is that whenever it leaves the i'th block of an input z, it remembers only s bits of infromation about the $\ell \gg s$ symbols it read in the block.

An oversimplifying assumption. Let us make an oversimplifying assumption and pretend that we are dealing with a channel C that "wipes its memory" whenever it leaves a block. This means that the errors injected on each block are determined by the content of that block (and does not depend on previous blocks). In this case, we claim that:

- For every $i \in [u]$, $Y^{(i)}$ is independent of $Z[i]$.
- Whenever $\mathrm{Dec}(V)$ does not reject, we have that for every $i \in [u]$, $Y^{(i)} = Y$.

For the first item we observe that having erased the i'th block, $V^{(i)}$ is determined by $V[1], \ldots, V[i-1], V[i+1], \ldots, V[u]$ which in turn is independent of $Z[i]$. (The latter follows by the simplifying assumption that the error induced by C on blocks $1, \ldots, i-1, i+1, \ldots, u$ does not depend on $Z[i]$). As $Y^{(i)}$ is determined by $V^{(i)}$ it is independent of $Z[i]$.

For the second item, we note that the absolute Hamming distance between V and $V^{(i)}$ is at most ℓ (as they agree on all blocks except the i'th block) and we have chosen ℓ so that an absolute distance of ℓ is a

relative distance smaller than λ. Therefore, if Dec did not reject V, then V is within relative distance λ to a codeword, and so, for every $i \in [u]$, $V^{(i)}$ is within relative distance 2λ to the same codeword. This gives that $\mathrm{Dec}_{2\lambda}(V^{(i)}) = \mathrm{Dec}_{2\lambda}(V)$ and $Y^{(i)} = Y$. This completes the proof of the second item.

However, by the first item, the expected relative distance between the $Y^{(i)}[i]$ and an independent random string $Z[i]$ is half. By a union bound and a Chernoff bound, w.h.p. for every $i \in [u]$, this distance is very close to half. This cannot hold together with the second item, because then, for every i, the relative Hamming distance between $Y[i]$ and $Z[i]$ is very close to half, which means that the relative distance between Y and Z is very close to half. This is impossible, because Y was obtained from Z by injecting p relative errors, and then correcting to a codeword that is within distance 2λ. This is impossible as $p + 2\lambda < \frac{1}{2}$.

Removing the simplifying assumption. We have made a simplifying assumption that C "wipes its memory" whenever it leaves a block. In order to justify this assumption, we will perform the previous analysis for every fixing of the states of C at the end of all u blocks. For every such fixing, the assumption holds (as the states of C at the end of blocks are fixed). Furthermore, the blocks $Z[1], \ldots, Z[u]$ remain independent under such a fixing. For a typical fixing, the (min)-entropy of a block $Z[i]$ is at least $\ell \cdot \log|\Sigma| - s = (1 - o(1)) \cdot \ell \cdot \log|\Sigma|$, and the previous analysis can be repeated using the fact that every event that happens with low probability over the uniform distribution, also happens with slightly larger, but still low probability, in a high (min)-entropy distribution. This allows us to remove the oversimplifying assumption.

1.3.3 Evasiveness on pseudorandom strings

We will now consider a version of the evasiveness experiment, in which rather than choosing a uniform $Z \leftarrow U_n$, we will choose $Z = G(U_d)$ where $G : \{0,1\}^d \to \{0,1\}^n$ is a pseudorandom generator against small space ROBPs with seed length $d \ll n$.[10] We would like to argue that if a code is evasive, then it is also evasive on pseudorandom strings. At first glance, it may seem that this follows because G fools small space ROBPs and C is a space bounded channel. However, this is not the case, because in the evasiveness experiment, in order to determine whether C wins, one needs to apply the decoding algorithm Dec (which is not implementable by a small space ROBP) and is not fooled by G.

One way to think about this, is that the "adversary" that we need G to fool is an adversary $A(Z)$ that is more powerful than C, and in addition to applying C, $A(Z)$ runs an additional computation that is not a small space ROBP.

This problem is a recurring theme in this paper, and will also come up later on. Fortunately, at this point, we can use *locally correctable* and *locally testable* codes in order to show that a pseudorandom generator G for small space ROBPs suffices and prove that:

Informal Theorem 1.5. *The codes of informal theorems 1.4 and 1.3 remain evasive when $Z = G(U_d)$ where G is a pseudorandom generator for small space ROBPs.*

For this, we use the recent remarkable results of Guo, Kopparty and Sudan [GKS13] and Kopparty, Meir, Ron-Zewi and Saraf [KMRS17]. These results allow to take an explicit high rate code Enc in informal theorem 1.4 that is locally correctable and locally testable with $Q = n^{o(1)}$ queries.[11] A space $O(Q)$ ROBP can simulate both local correcting and local testing algorithms on an input v by reading v in one pass, and storing all the Q symbols required by the (non-adaptive) local tester or local corrector. Using these local correcting and local testing capabilities, a space $n^{o(1)}$ ROBP can w.h.p. *approximate* the distance between

[10] A read once branching program (ROBP) $A(x)$ is a procedure that reads the input x in one pass using bounded space. A precise definition is given in Section 2.2. We use "channel" for the case where the ROBP outputs one bit for every input bit, and "ROBP" for the case of one bit output. A formal definition of pseudorandom generators is given in Section 2.2.2.

[11] A precise definition of locally correctable codes and locally testable codes appears in Section 2.5.2.

a given string v and the closest codeword, and determine whether that distance is smaller than λ, which determines whether C wins (without having to actually run decoding).

This means that the "combined adversary" $A(Z)$ (that on input Z applies the channel to obtain $V = Z \oplus C(Z)$ and determines whether V is within distance λ from a codeword) can be compute by a small space ROBP. This means that $A(Z)$ is fooled by the pseudorandom generator G, and we obtain that the codes of informal theorems 1.4 and 1.3 are evasive on pseudorandom strings.

1.3.4 A high level overview of the list decodable stochastic codes of [GS16, KSS19]

We are planning to use the evasive codes explained in the earlier section as a component in our construction of *uniquely decodable* stochastic codes for space bounded channels. This construction will build on the earlier *list decodable* stochastic codes for space bounded channels by [GS16, KSS19]. We will modify this construction in several key ways. We start by giving a high level description of the list-decodable codes of [GS16, KSS19] (focusing on issues that are important to explain our results).

Codes with shared private randomness. The first ingredient in this construction is a *uniquely decodable code with shared private randomness* for space bounded channels. This is a pair of functions $\mathrm{Enc}_{\mathrm{data}} : \{0,1\}^k \times \{0,1\}^d \to \{0,1\}^{N_{\mathrm{data}}}$ and $\mathrm{Dec}_{\mathrm{data}} : \{0,1\}^{N_{\mathrm{data}}} \times \{0,1\}^d \to \{0,1\}^k$ such that for every message $m \in \{0,1\}^k$ and for every $C \in \mathrm{Spc}_p^s$, let $Y = \mathrm{Enc}_{\mathrm{data}}(m, S)$ be the codeword, and $\bar{Y} = Y \oplus C(Y)$ be the "received word". It is guaranteed that:

$$\Pr_{S \leftarrow U_d}[\mathrm{Dec}_{\mathrm{data}}(\bar{Y}, S) = m] \geq 1 - \nu.$$

Note that unlike the setup of stochastic codes, in this setup of shared private randomness, the decoding algorithm Dec, *does receive* the randomness S chosen by the encoding procedure. This gives the decoding a huge advantage over the case of stochastic codes (as the channel does not receive S). Indeed, it is much easier to construct such codes than stochastic codes (in which the decoding algorithm does not receive S). Explicit codes with this property, and rate $1 - H(p)$ were constructed in [GS16, KSS19] (based on earlier ideas due to Lipton [Lip94], see also [Smi07]). These constructions are based on codes for BSC_p. Using the codes of informal theorem 1.3, and informal theorem 1.5, we can get that this code is evasive on pseudorandom strings.

From codes for shared randomness to list decodable stochastic codes. In order to convert a code with shared private randomness into a stochastic code, the encoding algorithm needs to send the seed S to the decoding algorithm in a way that the channel will not be able to learn S, or to prevent the decoding algorithm from recovering S.

The idea in [GS16, KSS19] is to combine Y and S together into one word $Z = \mathrm{Combine}(Y, S)$ of length $N = (1 + o(1))N_{\mathrm{data}}$. It is then shown that after a small space channel $C \in \mathrm{Spc}_p^s$ corrupts the codeword Z. The decoding algorithm that receives $\bar{Z} = Z \oplus C(Z)$, can still recover a small list of candidate that contains the original seed S.

More formally, the list-decoding algorithms of [GS16, KSS19] work by designing such a (randomized) procedure Combine together with a *list-decoding* procedure Recover, that given $\bar{Z}$ produces a list $(\bar{Y}^1, \bar{S}^1), \ldots, (\bar{Y}^L, \bar{S}^L)$ such that there exists $i \in [L]$ for which $\bar{S}^i = S$ and $\bar{Y}^i$ is a string $\bar{Y}$ that is obtained by applying the channel C on Y. Consequently, a list-decodable stochastic code $(\mathrm{Enc}, \mathrm{Dec})$ with rate $1 - H(p)$ can be constructed by:

Encoding: On a message m and randomness S, $\mathrm{Enc}(m, S)$ does the following:

- Compute $Y = \mathrm{Enc}_{\mathrm{data}}(m, S)$.
- Compute $Z = \mathrm{Combine}(Y, S)$, and output Z.

List-Decoding: On a received word $\bar{Z} = Z \oplus C(Z)$, $\mathrm{Dec}(\bar{Z})$ does the following:

- Apply $\mathrm{Recover}$ on $\bar{Z}$ to obtain a list of pairs $(\bar{S}^1, \bar{Y}^1), \ldots, (\bar{S}^L, \bar{Y}^L)$.
- For every $i \in [L]$, output the candidate message $\mathrm{Dec}_{\mathrm{data}}(\bar{Y}^i, \bar{S}^i)$.

It is guaranteed that w.h.p. there exists an $i^* \in [L]$ for which $\bar{S}^{i^*} = S$, and then, $\mathrm{Dec}_{\mathrm{data}}(\bar{Y}^{i^*}, \bar{S}^{i^*}) = \mathrm{Dec}_{\mathrm{data}}(\bar{Y}, S) = m$. This means that w.h.p. the original message m indeed appears in the list.

1.3.5 Using evasiveness to reject incorrect candidates

We would like to use evasiveness to design a pruning procedure that rejects all incorrect messages in the list. The key will be to identify a property that will allow us to distinguish $(\bar{S}^i, \bar{Y}^i)$ for $i \neq i^*$, from $(\bar{S}^{i^*}, \bar{Y}^{i^*})$.

We will modify the constructions of the functions $\mathrm{Combine}$ and $\mathrm{Recover}$ so that we can argue that for every $i \neq i^*$, $\bar{S}^i$ is independent of $S = \bar{S}^{i^*}$. (We remark that that here we are oversimplifying and we are not able to achieve this goal as stated).[12] Quite a bit of the technical work in the paper is devoted to this step. Nevertheless, in this high level overview, we will not elaborate on these ideas, as this require delving into the precise implementation of $\mathrm{Combine}$ and $\mathrm{Recover}$. We will now show how to use evasiveness to identify i^* assuming that for every $i \neq i^*$, $\bar{S}^i$ is independent of $S = \bar{S}^{i^*}$.

We will append an additional short random seed (which we denote by S_{PRG}) to S. This means that S is composed of two short seeds $S = (S_{\mathrm{data}}, S_{\mathrm{PRG}})$, where S_{data} is the seed used for the data code. We will use a pseudorandom generator G that upon receiving a short seed S_{PRG} produces a "pseudorandom string" $W = G(S_{\mathrm{PRG}})$ of length N_{data}. (We will discuss the pseudorandomness properties that we require from G shortly). We will then "mask" the data encoding by xoring it with W (prior to combining it with S). This leads to the following stochastic encoding/list-decoding scheme.

Encoding: On a message m and randomness $S = (S_{\mathrm{PRG}}, S_{\mathrm{data}})$, $\mathrm{Enc}(m, S)$ does the following:

- Compute $X = \mathrm{Enc}_{\mathrm{data}}(m, S_{\mathrm{data}})$.
- Compute $W = G(S_{\mathrm{PRG}})$.
- Compute $Y = X \oplus W$.
- Compute $Z = \mathrm{Combine}(Y, S)$, and output Z.

List-Decoding: On a received word $\bar{Z} = Z \oplus C(Z)$, $\mathrm{Dec}(\bar{Z})$ does the following:

- Apply $\mathrm{Recover}$ on $\bar{Z}$ to obtain a list of pairs $(\bar{S}^1, \bar{Y}^1), \ldots, (\bar{S}^L, \bar{Y}^L)$.
- For every $i \in [L]$,
  - Compute $\bar{W}^i = G(\bar{S}^i_{\mathrm{PRG}})$.
  - Compute $\bar{X}^i = \bar{Y}^i \oplus \bar{W}^i$.
  - Output the candidate message $\mathrm{Dec}_{\mathrm{data}}(\bar{X}^i, \bar{S}^i_{\mathrm{data}})$.

This scheme also achieves list decoding, because on the correct i^*, $\bar{S}^{i^*} = S$, meaning that $\bar{W}^{i^*} = G(\bar{S}^{i^*}) = G(S) = W$, and the two strings $\bar{W}^{i^*}$ and W cancel each other when $i = i^*$. Consequently, the correctness of list-decoding follows from the correctness of the previously described list decoding scheme.

[12] It turns out that due to the structure of previous constructions, the channel C has quite a bit of control on the L candidates $(\bar{S}^1, \bar{Y}^1), \ldots, (\bar{S}^L, \bar{Y}^L)$. More specifically, the channel can control many of these candidates and set them to any fixed value of his choice. Furthermore, while the channel does not know the correct seed S, it can pick in advance a function f, and guarantee that some of the candidates in the list will be set to $f(S)$. By modifying the construction of $\mathrm{Combine}$ and $\mathrm{Recover}$ we will be able to argue that the number of i for which $\bar{S}^i$ and $S = \bar{S}^{i^*}$ are correlated is small. This means that we still need to handle the few i which are correlated, and we ignore this problem in this high level overview.

Connecting this scheme to the evasiveness of the data code. We now observe that in this scheme, there is indeed a big difference between i^* and $i \neq i^*$, specifically:

- For $i = i^*$, we have that $\bar{X}^{i^*} = X \oplus E$, where X is the data codeword, and E has relative Hamming weight at most p, and was generated by a space s channel.

  This follows because on the correct i^*, $\bar{S}^{i^*} = S$ and so the $W = \bar{W}^{i^*}$ and the two strings cancel each other when list-decoding.

- For $i \neq i^*$, we have that $\bar{X}^i = R \oplus E^i$ where R is a *pseudorandom* string, and E^i has relative Hamming weight at most p, and was generated by a space s channel.

  This loosely follows because on $i \neq i^*$, $\bar{S}^i$ and S are independent. Therefore, the two strings $\bar{W}^i$ and W that are being xored in the list-decoding algorithm, are independent. Furthermore, we have that $W = G(S_{\mathrm{PRG}})$ is a pseudorandom string, and so it remains pseudorandom when xored with independent strings. This gives that the string $\bar{X}^i$ is a string that was obtained by corrupting a pseudorandom string with a space s channel.[13]

We would like to use the "evasiveness on pseudorandom strings" of the data code $(\mathrm{Enc}_{\mathrm{data}}, \mathrm{Dec}_{\mathrm{data}})$ to argue that in the last step of the decoding, for $i \neq i^*$, $\mathrm{Dec}_{\mathrm{data}}$ is applied on a pseudorandom string that was corrupted by a small space channel and is therefore rejected.

Unfortunately, we once again run into the problem that the "adversary" $A(W)$ that we are dealing with is not a small space ROBP. This is because the computation performed in this experiment on the string $W = G(S_{\mathrm{PRG}})$ (which we denote by $A(W)$) includes steps that cannot be computed by a small space ROBP. Specifically, this computation applies the pseudorandom generator G (as well as other steps hidden in the procedure $\mathrm{Recover}$) and cannot be computable by a small space ROBP.[14]

Using evasiveness for strings that are pseudorandomness for ROBPs. We now explain how we overcome these difficulties. By making some additional modifications to the functions $\mathrm{Combine}$ and $\mathrm{Recover}$ (which we will not explain here) we will be able to argue that for every channel C, there exists a not too large set H_C of *fixed* candidates $\bar{s}$, such that every candidate $\bar{S}^i$ for $i \neq i^*$, is with high probability in H_C. This means that if we want to simulate $A(W)$ by a small ROBP (that depends on C), we can guess in advance which candidates will come up during the computation of A, and for every $\bar{s} \in H_C$, we can consider a "fixed candidate adversary" $A_{\bar{s}}(W)$ that is hardwired with (the constants) $\bar{s}$ and $G(\bar{s}_{\mathrm{PRG}})$. This "fixed candidate adversary" $A_{\bar{s}}(W)$ can now be simulated by a small space ROBP (as it does not need to run G of $\mathrm{Recover}$).

It follows using the evasiveness of $\mathrm{Enc}_{\mathrm{data}}$ that the probability that $A_{\bar{s}}(W)$ wins is small. We can do a union bound over all the choices of $\bar{s} \in H_C$ to get that the probability that one of these adversaries win is small, and this gives a bound on the probability that $A(W)$ wins. Overall, this shows that the probability that an incorrect $i \neq i^*$ is not rejected is small, meaning that we can identify the correct candidate i^*.

Finally, we stress once again that this high level explanation is oversimplified, and ignores many issues that come up in the construction and analysis. The reader is referred to the technical section for precise details. In particular, Section 5.3 has a high level intuition of the construction.

Organization of this paper

In Section 1.4 we survey some additional related work that is not directly comparable to our work (and was not surveyed before). In Section 2 we give preliminaries and formal definitions. We also state past work on

[13] We remark that here we are oversimplifying, and this clean statement doesn't follow precisely as stated. However, a very similar statement does follow using some additional properties of $\mathrm{Combine}$ and $\mathrm{Recover}$.

[14] We remark that a pseudorandom generator G that fools adversaries that can compute the pseudorandom generator, immediately implies one way functions, if it fools strong adversaries. Therefore, such pseudorandom generators are not expected to exist unconditionally, and require cryptographic assumptions in our current state of knowledge.

the tools and ingredients that are used in our construction. In Section 3 we construct evasive codes for BSC_p with rate $1-H(p)$ and list additional properties of these codes that will be used in our construction. In Section 4 we discuss stochastic control codes. These are an important ingredient in our construction, and in order to achieve unique decoding, we need to introduce a property of "repetition decoding" that was not considered in earlier work [GS16, SS16, KSS19], and construct codes with this property. Our main construction of stochastic codes for space bounded channels is presented in Section 5. In that section, we also explain the difference between our construction and the previous list-decodable codes of [GS16, KSS19]. In Section 6 we prove the correctness of our construction.

1.4 More related work on codes for bounded channels

In this section we survey some additional related work that is not directly comparable to our work (and was not surveyed before).

1.4.1 Stochastic codes for other classes of channels

poly-size circuits. Guruswami and Smith [GS16] also gave constructions of stochastic codes with rate approaching $1 - H(p)$ that are list-decodable for channels that are circuits of size n^c, and induce pn errors. They achieved success probability n^{-c}. A significant drawback of these results is that the running time of the encoding algorithm was polynomial in n^c, for a large and unspecified polynomial (meaning that efficiency quickly deteriorates even for conservative estimates on channel complexity). The construction of [GS16] is "Monte-Carlo". Meaning that it requires a preprocessing stage, in which a random string of length $\mathrm{poly}(n^c)$ is shared between the encoding and decoding algorithm. The correctness of encoding and decoding algorithms is guaranteed w.h.p. over the choice of this string. (This string need not be kept secret from the channel).

Shaltiel and Silbak [SS16] removed the need for a preprocessing stage by slightly modifying the construction of Guruswami and Smith, and providing explicit constructions for the modified components. They give results for size n^c circuits (here a complexity assumption that there are functions in $\mathit{DTIME}(2^{O(n)})$ that are hard for small circuits is used, and is necessary).

Non malleable codes. Non-malleable codes (introduced by Dziembowski, Pietrzak, and Wichs [DPW18]) consider channels that are not restricted in the number of errors that they induce. Instead, channels are assumed to come from some limited class of functions (or complexity class). Codes are stochastic (meaning that the encoding procedure is randomized) and it is required that following the corruption by the channel, the decoder either reproduces the encoded message, or an "unrelated" message. The definition of "unrelated" is given using the simulation paradigm from cryptography. Several classes have been considered, and some of the constructions rely on cryptographic assumptions. The reader is referred to [DPW18] and the references therein for precise definition and a survey of results in non-malleable codes.

1.4.2 Other coding scenarios with randomized encoding or bounded channels

The notion of computationally bounded channels was also studied in other setups. We mention some of these works below.

Shared private randomness. We start with the notion of codes with "shared private randomness". While this setup was considered before the notion of stochastic codes, in this paper, it is natural to view it as a version of stochastic codes in which the decoding algorithm *does* receive the randomness S chosen by the encoding algorithm. This corresponds to a standard symmetric cryptography setup in which honest parties (the encoder and decoder) share a uniform private key S, and the bad party (the channel) does not get the key. Lipton [Lip94] and following work (see [Smi07] for more details) gave explicit constructions of uniquely

decodable codes against computationally bounded channels, in this setup, with rate approaching $1 - H(p)$, under cryptographic assumptions.

Note that the setup of stochastic codes is lighter. The encoder and decoder do not need to share a private random key. Moreover, a fresh key can be chosen on the spot every time the encoder encodes a message.

A related notion of "private codes" was studied by Langberg [Lan04]. This is also in the setup of shared private randomness. Here channels are computationally unbounded, codes are existential rather than explicit, and have rate approaching $1 - H(p)$. The focus is on minimizing the length of the shared key. Langberg provides asymptotically matching upper and lower bounds of $\Theta(\log n + \log(1/\nu))$, on the amount of randomness that needs to be shared for unique decoding in this setup, where ν is the error parameter.

Public key setup. Micali et al. [MPSW10] considered computationally bounded channels, and a cryptographic public key setup. Their focus is to use this setup to convert a given (standard) explicit list-decodable code into an explicit uniquely decodable codes (in this specific public key setup).

2 Preliminaries, and ingredients used in the construction

In this section we give formal definitions of the notions and ingredients used in the construction. We also cite previous results from coding theory and pseudorandomness that are used in the construction.

General notation. We use $H(p)$ to denote the ***Shannon binary entropy*** function: $H(p) = p \cdot \log(1/p) + (1-p) \cdot \log(1/(1-p))$. We use U_n to define the uniform distribution on n bits. The ***statistical distance*** between two distributions P, Q over Ω is $\max_{A \subseteq \Omega} |P(A) - Q(A)|$. The ***min-entropy*** of P is $H_\infty(P) = min_{s \in \Omega}(\log \frac{1}{p(s)}$.

We sometimes use the notation $O_\lambda(\cdot)$ to emphasize that the constant hidden in the $O(\cdot)$ notation may depend on λ.

Definition 2.1 (Hamming distance and weight)**.** *The* **Hamming weight** *of $x \in [q]^n$ is $WT(x) = |\{i : x_i \neq 0\}|$. The* **relative Hamming weight** *of x is $wt(x) = \frac{WT(x)}{n}$. The* **Hamming distance** *between $x, y \in [q]^n$ is $\Delta(x, y) = |\{i : x_i \neq y_i\}|$. The* **relative Hamming distance** *between $x, y \in [q]^n$ is $\delta(x, y) = \frac{\Delta(x,y)}{n}$.*

2.1 Permuting strings

We will use a permutation $\pi : [n] \to [n]$ to "reorder" the bits of a string $x \in \{0,1\}^n$: The i'th bit in the rearranged string will be $\pi(i)$'th bit in x. This is captured in the definition below.

Definition 2.2 (Permuting strings)**.** *Given a string $x \in \{0,1\}^n$ and a permutation $\pi : [n] \to [n]$. Let $\pi(x)$ denote the string $x' \in \{0,1\}^n$ with $x'_i = x_{\pi(i)}$.*

2.2 Read once branching programs

2.2.1 Formal definition of ROBPs and bounded space channels

We give a formal definition of bounded space computation and channels. The model that we consider is that of oblivious read once branching programs (ROBP). In the definition below, we will consider several variants depending on whether the ROBP outputs a single bit, or one bit per any input bit (which is the case for channels that are function $C : \{0,1\}^n \to \{0,1\}^n$).

Definition 2.3 (Read Once Branching Programs (ROBP) and channels)**.** *A space s* **ROBP** *C which receives input in $\{0,1\}^n$ is defined by picking n transition functions $\delta_1, \ldots, \delta_n$ where for each i, $\delta_i : \{0,1\}^s \times \{0,1\} \to \{0,1\}^s$. On input $x \in \{0,1\}^n$, the computation path of C is the sequence $r_0, \ldots, r_n$ of states defined by $r_0 = 0^s$ and for $i \geq 1$, $r_i = \delta_i(r_{i-1}, x_i)$. We distinguish between two types of ROBPs:*

- *If $C : \{0,1\}^n \to \{0,1\}$ is an ROBP that outputs a single bit, then C also has an output function $o : \{0,1\}^s \to \{0,1\}$ and $C(x)$ is defined to be $o(r_n)$.*
- *If $C : \{0,1\}^n \to \{0,1\}^n$ is an ROBP that outputs n bits, then C also has n output functions $o_1, \ldots, o_n$ where for each i, $o_i : \{0,1\}^s \to \{0,1\}$ and $C(x)$ is defined by the n bit string $o_1(r_1), \ldots, o_n(r_n)$.*

The class Spc_p^s is the class of all space s ROBPs $C : \{0,1\}^n \to \{0,1\}^n$ such that for every input x, $C(x)$ has Hamming weight at most pn. We will say that such channels "induce pn errors".

We are stating this definition in the terminology of "transition functions" and "output functions" which is more convenient when discussing ROBPs that output more than one bit. However, we stress that this definition is equivalent to the more common definition of width $w = 2^s$ ROBPs in terms of a layered graph with $n+1$ layers of "width" w, where the i'th transition function specifies the edges from the $(i-1)$'th level to the i'th level.

2.2.2 PRGs for ROBPs

We need the following standard definition of pseudorandom distributions and generators.

Definition 2.4 (Pseudorandom generators)**.** *A distribution X on n bits is ϵ-***pseudorandom** *for a class $\mathcal{C}$ of functions from n bits to one bit, if for every $C \in \mathcal{C}$, $|\Pr[C(X) = 1] - \Pr[C(U_n)] = 1]| \leq \epsilon$. A function $G : \{0,1\}^d \to \{0,1\}^n$ is an ϵ-***PRG** *for $\mathcal{C}$ if $G(U_d)$ is ϵ-pseudorandom for $\mathcal{C}$.*

A long line of work is concerned with pseudorandom generators for small space ROBPs. Some of the considerations that we make in this paper are reminiscent of [Nis92, NZ96, INW94, RR99]. We will use the following PRG by Forbes and Kelly [FK18]. The furthermore part is proven in [KSS19]. (We remark that earlier constructions also have these properties, and we use [FK18] because the furthermore part was verified in [KSS19]).

Theorem 2.5. *For every $\log n \leq s \leq n$, there exists an ϵ-PRG $G : \{0,1\}^d \to \{0,1\}^n$ for space s ROBPs that output one bit, with $d = O((s + \log \frac{1}{\epsilon}) \cdot \log^2 n)$. Furthermore, G can be computed in time $O(n \cdot \mathrm{polylog}(n))$.*

2.3 Averaging Samplers

The reader is referred to Goldreich's survey [Gol97] on averaging samplers.

Definition 2.6 (Averaging Samplers)**.** *A function $\mathrm{Samp} : \{0,1\}^n \to (\{0,1\}^m)^t$ is an (ϵ, δ)-***Sampler** *if for every $f : \{0,1\}^m \to [0,1]$,*

$$\Pr_{(z_1,\ldots,z_t) \leftarrow \mathrm{Samp}(U_n)}[|\frac{1}{t}\sum_{i \in [t]} f(z_i) - \frac{1}{2^m}\sum_{x \in \{0,1\}^m} f(x)| > \epsilon] \leq \delta.$$

A sampler has **distinct samples** *if for every $x \in \{0,1\}^n$, the t elements in $\mathrm{Samp}(x)$ are distinct.*

The next theorem follows from the "expander sampler". This particular form can be found (for example) in [Vad04].

Theorem 2.7. *For every sufficiently large m and every $\epsilon \geq \delta > 0$ such that $m \leq \log(1/\delta)$ there is an (ϵ, δ)-sampler with distinct samples, $\mathrm{Samp} : \{0,1\}^{O(\log(1/\delta) \cdot \mathrm{poly}(1/\epsilon))} \to (\{0,1\}^m)^t$ for any $t \leq 2^m$ such that $t \geq \mathrm{poly}(1/\epsilon) \cdot \log(1/\delta)$. Furthermore, Samp is computable in time $t \cdot \mathrm{poly}(1/\epsilon, \log(1/\delta))$ and has distinct samples.*

2.4 Almost t-wise independent permutations

We also need the following notion of almost t-wise permutations.

Definition 2.8 (Almost t-wise independent permutations)**.** *A function $\pi : \{0,1\}^d \times [n] \to [n]$ is an (ϵ, t)-wise independent permutation if:*

- *For every $s \in \{0,1\}^d$ the function $\pi_s(i) = \pi(s,i)$ is a permutation over $[n]$.*
- *For every distinct $i_1, \ldots, i_t \in [n]$, the random variable $R = (R_1, \ldots, R_t)$ defined by $R_j = \pi(s, i_j) \colon s \leftarrow U_d$, is ϵ-close to t uniform samples without repetition from $[n]$.*

Theorem 2.9. *[KNR09] For every t and every sufficiently large n, there exists an (ϵ, t)-wise independent permutation with $d = O(t \cdot \log n + \log(1/\epsilon))$. Furthermore, computing $\pi(s,i)$ on inputs $s \in \{0,1\}^d$ and $i \in [n]$ can be done in time $\mathrm{poly}(d, \log n)$.*[15]

2.5 Error-Correcting Codes

In this section we give formal definitions of some of the various notions of error correcting codes used in this paper. We will also introduce less standard definitions in the next sections.

A code is pair $(\mathrm{Enc}, \mathrm{Dec})$ of encoding and decoding algorithms, and different notions are obtained by considering the requirements on the decoding algorithm.

2.5.1 Standard notions of error correcting codes

We start by giving definitions of error correcting codes that covers the standard cases of Hamming channels and binary symmetric channels. In the definition below we consider codes that are not necessarily binary (as we will use non-binary codes as components in our construction).

Definition 2.10 (Codes for Hamming channels)**.** *Let k, n, q be parameters and let $\mathrm{Enc} : \{0,1\}^k \to (\{0,1\}^{\log q})^n$ be a function. We say that Enc is an encoding function for a code that is:*

- **decodable from t errors***, if there exists a function $\mathrm{Dec} : (\{0,1\}^{\log q})^n \to \{0,1\}^k$ such that for every $m \in \{0,1\}^k$ and every $v \in (\{0,1\}^{\log q})^n$ with $\Delta(\mathrm{Enc}(m), v) \leq t$, $\mathrm{Dec}(v) = m$.*
- **L-list-decodable from t errors***, if the function Dec is allowed to output a list of size at most L, and for every $m \in \{0,1\}^k$ and every $v \in (\{0,1\}^{\log q})^n$ with $\Delta(\mathrm{Enc}(m), v) \leq t$, $\mathrm{Dec}(v) \ni m$.*
- **decodable from** *P, with success probability $1 - \nu$, if P is a distribution over $(\{0,1\}^{\log q})^n$, $0 \leq \nu \leq 1$, and there exists a function $\mathrm{Dec} : (\{0,1\}^{\log q})^n \to \{0,1\}^k$ such that for every $m \in \{0,1\}^k$, $\mathrm{Pr}_{e \leftarrow P}[\mathrm{Dec}(\mathrm{Enc}(m) \oplus e) = m] \geq 1 - \nu$.*

In the definitions above, we use "δ **relative errors***" to mean "$t = \delta n$ errors".*

A code has **encoding time** *[resp.* **decoding time***] $T(\cdot)$, if Enc [resp. Dec] can be computed in time $T(n \log q)$. The code is* **explicit** *if both encoding and decoding run in polynomial time. (Naturally, this makes sense only for a family of encoding and decoding functions with varying block length n, message length $k(n)$, and alphabet size $q(n)$).*

The rate of the code is the ratio of the message length and output length of Enc, where both lengths are measured in bits. That is the rate $R = \frac{k}{n \cdot \log q}$.

[15] We will be interested in the time it takes to compute the permutation on all $i \in [n]$ (namely given s, we want to compute $(\pi(s,i))_{i \in [n]}$) and will use $n \cdot \mathrm{poly}(d)$ as a bound on the time for this task. Note that this also gives that computing $(\pi^{-1}(s,i))_{i \in [n]}$ can be done within the same time bound.

The notion of "decoding from errors" corresponds to *Hamming channels*, where the decoding algorithm needs to decode (or list-decode) from a certain distance. We remark that it is standard that a code is decodable from t errors if and only if the Hamming distance between every two codewords is at least $2t+1$.

The notion of decoding from P covers the case of *BSC channels*, by choosing $q = 2$ and P to be the distribution BSC_p of n i.i.d. bits where each bit has probability p to be one.

2.5.2 Locally-correctable and locally-testable codes

We will rely on constructions of locally-corerctable and locally-testable codes with high rate by Kopparty, Meir, Ron-Zewi and Saraf [KMRS17]. For our application it is crucial that these constructions have a *non-adaptive* local corrector. This can be verified by inspecting the construction.[16] In the definition below, we restrict ourselves to locally correctable codes and locally testable codes with non-adaptive local correctors.

Definition 2.11 (Locally-correctable and locally testable codes with non-adaptive queries)**.** *Let k, n, q be parameters and let $\mathrm{Enc} : \{0,1\}^k \to (\{0,1\}^{\log q})^n$ be a function.*

- *We say that* Enc *is an encoding function for a code that is* **non-adaptively locally-correctable** *from t errors using Q queries, if there exists a randomized local-correcting procedure $\mathrm{Dec}^{(\cdot)}(\cdot)$ such that for every $m \in \{0,1\}^k$, every $i \in [n]$ and every $v \in (\{0,1\}^{\log q})^n$ with $\Delta(\mathrm{Enc}(m), v) \leq t$, we have that when* Dec *receives input i, it makes at most Q nonadaptive oracle calls to v, and:*

$$\Pr[\mathrm{Dec}^v(i) = \mathrm{Enc}(m)_i] \geq \frac{2}{3}.$$

- *We say that* Enc *is an encoding function for a code that is* **non-adaptively locally-testable** *from t errors using Q queries, if $t \in [n]$, and there exists a randomized local-testing procedure $\mathrm{Dec}^{(\cdot)}(\cdot)$ such that for every $v \in (\{0,1\}^{\log q})^n$ the following holds:*
  - *If there exists $m \in \{0,1\}^k$ such that $v = \mathrm{Enc}(m)$ then $\Pr[\mathrm{Dec}^v = 1] \geq \frac{2}{3}$.*
  - *If for every $m \in \{0,1\}^k$, $\Delta(v, \mathrm{Enc}(m)) > t$ then $\Pr[\mathrm{Dec}^v = 1] \leq \frac{1}{3}$.*

  In the definitions above, we use "δ **relative errors***" to mean "$t = \delta n$ errors".*

Theorem 2.12 (High rate locally-correctable and locally-testable codes [KMRS17])**.** *For every constants $0 < r < 1$ and $\epsilon > 0$, there exists a constant $q = 2^{\mathrm{poly}(1/\epsilon)}$ such that for infinitely many n, there is a function $\mathrm{Enc} : \{0,1\}^{r \cdot n \cdot \log q} \to (\{0,1\}^{\log q})^n$ (that is,* Enc *has rate r) such that:*

- Enc *is computable in polynomial time.*
- Enc *is decodable from $\frac{1-r-\epsilon}{2}$ relative errors, by a decoding algorithm that runs in polynomial time.*
- Enc *is non-adaptively locally correctable and locally-testable from $\frac{1-r-\epsilon}{2}$ relative errors, using $Q = n^{o(1)}$ queries, and a local-correcting and local testing algorithms that run in time $\mathrm{poly}(Q)$.*

Remark 2.13 (The parameters needed in our construction)**.** *The statement of Theorem 2.12 asks for codes that are simultaneously locally correctable and locally testable. Such codes are stated in Section 1.3 in [KMRS17]. These codes are exceptional as they match the Singleton bound and achieve $Q = n^{o(1)}$ queries.*

For our purposes, this is not necessary and (using a slightly more careful analysis) our results follow from codes with weaker parameters. More specifically:

[16] We remark that by the work of Ben-Sasson, Harsha and Raskhodnikova [BHR05], it follows that every linear code that is locally testable has a non-adaptive tester. We are not sure whether this also holds for locally-correctable codes.

- *We do not require codes that match the singleton bound. It is sufficient that for every $\epsilon > 0$ there exists a constant $\epsilon' > 0$ and a code with rate $1 - \epsilon$ that is locally-correctable and locally testable from $\epsilon' > 0$ relative errors. We can also allow the alphabet to be any large constant that only depends on ϵ.*
- *It is not important that the number of queries is $Q = n^{o(1)}$, and our results hold even if $Q = n^{\alpha}$ for a constant $\alpha > 0$, as long as we can choose this α to be sufficiently small. We can also allows the constant ϵ' and the alphabet size in the previous item to depend on α.*

The codes of Guo, Kopparty and Sudan [GKS13] already achieve this weaker property, and are in fact a component in the stronger result of [KMRS17] (which amplifies their parameters).

The method of [KMRS17] is known to produce codes with nearly linear time encoding (See Remark 2.6 in [HRW19] by Hemenway, Ron-Zewi and Wootters). Therefore, it seems likely that a better bound of $n^{1+o(1)}$, also applies for the encoding time in Theorem 2.12. For this to follow directly, one needs to make a careful inspection of the encoding time of the codes of Guo, Kopparty and Sudan [GKS13] when set up as locally-correctable codes. It seems very likely that these codes can be encoded in near linear time, but we have not carefully verified this.

Plugging codes with encoding time $n^{1+o(1)}$ into our construction gives a stochastic code with (randomized) encoding and decoding time $n^{1+o(1)}$. As we have not examined the codes of [GKS13] carefully, we now explain an alternative, indirect approach that achieves time encoding time $n^{1+o(1)}$.

A less direct approach that gives a code that achieves the parameters mentioned in the second item is given by "tensored codes". More specifically, by tensoring a code with itself t times, one can start with a linear locally-correctable code with Q queries, that has rate larger than $R = 1 - \beta/t$, and obtain a code with rate roughly $1 - \beta$, while still having local correction with roughly Q^t queries. The advantage of such a transformation is that if the initial code is explicit, then the obtained code has time $n^{1+O(1/t)}$ encoding and decoding. An additional advantage is that Viderman [Vid15] showed that tensored linear codes are also locally correctable with $N^{O(1/t)}$ queries (where N is the block length of the target code).

This procedure can be applied on the multiplicity codes of Kopparty, Saraf and Yekhanin [KSY14], or the aforementioned codes of Guo, Kopparty and Sudan [GKS13]. The rate of these codes can be taken to be larger than say $1 - O(\frac{1}{t^2})$, in order to allow the transformation described above for any large constant t. It is also possible to push t further, by taking versions of these codes that have sub-constant distance, to get better initial rate. Then, after applying the tensoring transformation, one can increase the distance (without harming the other parameters by too much) using the methods of Kopparty, Meir, Ron-Zewi and Saraf [KMRS17]. We defer the exact details to the final version.

2.5.3 Concatenated codes and outer distance

We give the following standard definition of concatenated codes.

Definition 2.14 (Concatenated code)**.** *Given functions:*

- $\mathrm{Enc}_{\mathrm{out}} : \{0,1\}^{k_{\mathrm{out}}} \to (\{0,1\}^{\log q_{\mathrm{out}}})^{n_{\mathrm{out}}}$, *and*
- $\mathrm{Enc}_{\mathrm{in}} : \{0,1\}^{k_{\mathrm{in}}} \to (\{0,1\}^{\log q_{\mathrm{in}}})^{n_{\mathrm{in}}}$,

such that $\log q_{\mathrm{out}} = k_{\mathrm{in}}$ *we define the* **concatenated encoding function** $\mathrm{Enc} : \{0,1\}^{k_{\mathrm{out}}} \to (\{0,1\}^{\log q_{\mathrm{in}}})^{n_{\mathrm{out}} \cdot n_{\mathrm{in}}}$ *denoted by* $\mathrm{Enc}_{\mathrm{out}} \circ \mathrm{Enc}_{\mathrm{in}}$ *as follows: For $i_{\mathrm{out}} \in [n_{\mathrm{out}}]$, $i_{\mathrm{in}} \in [n_{\mathrm{in}}]$, and $i = (i_{\mathrm{out}} - 1) \cdot n_{\mathrm{in}} + i_{\mathrm{in}}$ we define* $\mathrm{Enc}(m)_i = \mathrm{Enc}_{\mathrm{in}}(\mathrm{Enc}_{\mathrm{out}}(m)_{i_{\mathrm{out}}})_{i_{\mathrm{in}}}$.

Concatenated codes can be decoded by "concatenated decoding" which is the "natural decoding algorithm" that decodes each block using a decoding algorithm for the inner code, and then applies a decoding algorithm of the outer code. This algorithm is defined below. (We remark that in this definition, we don't care about

the decoding properties of the outer and inner codes, and the concatenated decoding is defined, without mentioning the decoding properties of the code.

We also define a notion of *relative outer distance*. This measure will play an important role in later sections. The relative outer distance of a "received word" $z \in \{0,1\}^n$ is the relative Hamming distance between $\mathrm{Enc}_{\mathrm{out}}(\mathrm{Dec}(z))$ and the string that came up in the concatenated decoding algorithm after decoding the inner code.

Definition 2.15 (Concatenated decoding and outer distance)**.** *Let* $\mathrm{Enc} = \mathrm{Enc}_{\mathrm{out}} \circ \mathrm{Enc}_{\mathrm{in}}$ *be a concatenated code, and let* $\mathrm{Dec}_{\mathrm{out}} : (\{0,1\}^{\log q_{\mathrm{out}}})^{n_{\mathrm{out}}} \to \{0,1\}^{k_{\mathrm{out}}}$, $\mathrm{Dec}_{\mathrm{in}} : (\{0,1\}^{\log q_{\mathrm{in}}})^{n_{\mathrm{in}}} \to \{0,1\}^{k_{\mathrm{in}}}$ *be functions. For* $i \in [n_{\mathrm{out}}]$ *we define* $\mathrm{Dec}^i_{\mathrm{in}} : (\{0,1\}^{\log q_{\mathrm{in}}})^{n_{\mathrm{out}} \cdot n_{\mathrm{in}}} \to \{0,1\}^{k_{\mathrm{in}}}$ *by:*

$$\mathrm{Dec}^i_{\mathrm{in}}(z) = \mathrm{Dec}_{\mathrm{in}}(z_{(i-1)\cdot n_{\mathrm{in}}+1}, \ldots, z_{i \cdot n_{\mathrm{in}}}).$$

The **concatenated decoding** *function* $\mathrm{Dec} : (\{0,1\}^{\log q_{\mathrm{in}}})^{n_{\mathrm{out}} \cdot n_{\mathrm{in}}} \to \{0,1\}^{k_{\mathrm{out}}}$ *is defined by:*

$$\mathrm{Dec}(z) = \mathrm{Dec}_{\mathrm{out}}(\mathrm{Dec}^1_{\mathrm{in}}(z), \ldots, \mathrm{Dec}^{n_{\mathrm{out}}}_{\mathrm{in}}(z)).$$

Under the same conditions, given $z \in \{0,1\}^n$*, the* **relative outer distance** *of* z *(for a specific choice of* $\mathrm{Enc}_{\mathrm{in}}, \mathrm{Enc}_{\mathrm{out}}, \mathrm{Dec}_{\mathrm{in}}$ *and* $\mathrm{Dec}_{\mathrm{out}}$*) is defined by:*

$$\mathrm{outdist}(z) = \frac{|\left\{i \in [n_{\mathrm{out}}] : \mathrm{Dec}^i_{\mathrm{in}}(z) \neq \mathrm{Enc}_{\mathrm{out}}(\mathrm{Dec}(z))_i\right\}|}{n_{\mathrm{out}}}.$$

Meaning and usefulness of outer distance. We will use a concatenated code $\mathrm{Enc} = \mathrm{Enc}_{\mathrm{out}} \circ \mathrm{Enc}_{\mathrm{in}}$ such that $\mathrm{Enc}_{\mathrm{out}}$ is decodable from λ relative errors, and $\mathrm{Enc}_{\mathrm{in}}$ will have a decoding algorithm $\mathrm{Dec}_{\mathrm{in}}$ with properties that we don't discuss at this point. We will use the concatenated decoding algorithm Dec as a decoding algorithm for Enc.

The relative outer distance $\mathrm{outdist}(z)$ measures the fraction of errors that $\mathrm{Dec}_{\mathrm{out}}$ corrected, when the concatenated decoding algorithm Dec was applies on z. We will be using an inner code $\mathrm{Enc}_{\mathrm{in}}$ in which $q_{\mathrm{in}} = 2$ and n_{in} is small. We will now argue that in such a setup, even though we're not sure what $\mathrm{Dec}_{\mathrm{in}}$ is doing, we can still obtain some properties of Dec with respect to relative outer distance that are analogous to properties of $\mathrm{Enc}_{\mathrm{out}}$ with respect to relative distance.

Specifically, In the case that $\mathrm{Enc}_{\mathrm{out}}$ decodes from λ relative errors, it is standard that if while running $\mathrm{Dec}_{\mathrm{out}}(z)$, the fraction of errors that were corrected is $\lambda' \leq \lambda$ and if $\delta(z,z') \leq \lambda - \lambda'$ then $\mathrm{Dec}_{\mathrm{out}}(z') = \mathrm{Dec}_{\mathrm{out}}(z)$. The following lemma stated that this holds for the concatenated code Enc, with respect to relative outer distance, while paying a cost that depends on n_{in}.

Lemma 2.16. *Under the setup of Definition 2.15, assume further that* $\mathrm{Dec}_{\mathrm{out}}$ *is a decoding algorithm showing that* $\mathrm{Enc}_{\mathrm{out}}$ *is decodable from* λ *relative errors. For every* z, z' *such that:*

- $\mathrm{outdist}(z) \leq \lambda' \leq \lambda$*, and*
- $\delta(z,z') \leq \frac{\lambda - \lambda'}{n_{\mathrm{in}}}$*,*

we have that $\mathrm{Dec}(z') = \mathrm{Dec}(z)$*, and* $\mathrm{outdist}(z') \leq \mathrm{outdist}(z) + \delta(z,z') \cdot n_{\mathrm{in}}$.

Proof. Let $\delta' = \delta(z,z')$. The strings z and z' differ in at most $\delta' \cdot n$ bits, and so when thinking of them as n_{out} blocks of length n_{in}, they differ in at most $\delta' n = (\delta' \cdot n_{\mathrm{in}}) \cdot n_{\mathrm{out}}$ blocks. This means that when the concatenated decoding algorithm is applied on z and z' the word obtained after inner decoding of z differs from the word obtained by inner decoding of z' in a $\delta' \cdot n_{\mathrm{in}}$ fraction of the n_{out} symbols. We have that when

concatenated decoding was applied on z, the outer code corrected λ' relative errors, and so the fraction of errors that is added is $\delta' \cdot n_{\text{in}}$. Together, the fraction of errors is at most

$$\lambda' + \delta' \cdot n_{\text{in}} \leq \lambda' + \frac{\lambda - \lambda'}{n_{\text{in}}} \cdot n_{\text{in}} \leq \lambda,$$

and this means that the concatenated decoding algorithm decodes z' to the same word that z was decoded to, and the outer relative distance is bounded by $\lambda' + \delta' \cdot n_{\text{in}}$. □

2.5.4 Stochastic Codes for a class of channels

In this section we give a precise formal definition of the notion of stochastic codes for a class of channels (that was already explained in the introduction).

Definition 2.17 (Stochastic codes for channels)**.** *Let k, n, d be parameters and let $\text{Enc} : \{0,1\}^k \times \{0,1\}^d \to \{0,1\}^n$ be a function. Let $\mathcal{C}$ be a class of functions from n bits to n bits. We say that* Enc *is an encoding function for a stochastic code that is:*

- **decodable** *for "channel class" $\mathcal{C}$, with success probability $1-\nu$, if there exists a (possibly randomized) procedure $\text{Dec} : \{0,1\}^n \to \{0,1\}^k$ such that for every $m \in \{0,1\}^k$ and every $C \in \mathcal{C}$, setting $X = \text{Enc}(m, U_d)$, we have that $\Pr[\text{Dec}(X \oplus C(X)) = m] \geq 1 - \nu$, where the probability is over coin tosses of the encoding and decoding procedures.*
- **L-list-decodable** *for "channel class" $\mathcal{C}$, with success probability $1-\nu$, if the procedure* Dec *is allowed to output a list of size at most L, and $\Pr[\text{Dec}(X \oplus C(X)) \ni m] \geq 1 - \nu$, where the probability is over coin tosses of the encoding and decoding procedures.*

A code has **encoding time** *[resp.* **decoding time***] $T(\cdot)$, if* Enc *[resp.* Dec*] can be computed in time $T(k + n + d)$. The code is* **explicit** *if both encoding and decoding run in polynomial time. (Naturally, this makes sense only for a family of encoding and decoding functions with varying block length n, message length $k(n)$ and seed length $d(n)$).*

The rate of the code is the ratio of the message length and output length of Enc*, where both lengths are measured in bits. That is the rate $R = \frac{k}{n}$.*

Remark 2.18 (Stochastic codes with randomized decoding)**.** *Definition 2.17 assumes that the decoding algorithm is deterministic. However, as we are allowing the encoding algorithm to be randomized, and allowing success probability smaller than one, we may as well allow the decoding algorithm to be randomized.*

This approach was used in [KSS19] to speed up the time of the decoding algorithm. Allowing randomized decoding also allows us to speed up the decoding time from polynomial to almost linear. We will elaborate on this in Section 5.6.

3 Evasive codes for BSC channels and related variants

In this section, we construct explicit codes with the following properties:

- Explicit decoding from BSC_p. (In fact, we will need a related, but stronger property of decoding from "almost t-wise independent errors" that we will explain later).
- Rate $1 - H(p)$.
- "Evasiveness" against Spc_p^s.

We start by giving an informal definition of the evasiveness property (we will state things more precisely below). This description is similar to the one given in Section 1.3.2.

The evasiveness experiment: Given $\mathrm{Enc} : \{0,1\}^k \to \{0,1\}^n$ and $\mathrm{Dec} : \{0,1\}^n \to \{0,1\}^k \cup \{\perp\}$, and a channel C: rather than giving the channel C a codeword of Enc to corrupts, we will be interested in the behavior of the channel and decoding algorithm on a ***uniformly chosen*** string. Specifically, we consider the following experiment:

- A uniform $Z \leftarrow U_n$ is chosen.
- The "received word" $V = Z \oplus C(Z)$ is obtained when the channel C "corrupts" Z.
- We apply $\mathrm{Dec}(V)$ and will say that the code $(\mathrm{Enc}, \mathrm{Dec})$ is *evasive* if the probability that $\mathrm{Dec}(V) \neq \perp$ is small.

Naturally, evasiveness is only interesting when coupled with some additional decoding properties of the code, like in our case decoding from BSC_p. Previous work on codes for space bounded channels (starting with Guruswami and Smith [GS16]) uses codes for BSC_p with rate approaching $1 - H(p)$, and the additional evasiveness notion that we introduce will be key in converting the earlier list decoding algorithms to unique decoding.

Remark 3.1 (Evasiveness and the Gilbert-Varshamov bound)**.** *Loosely speaking,* evasiveness *is one of the keys that allows us to achieve unique decoding while beating the Gilbert-Varshamov bound. More specifically, it is easy to see that any code with rate $R < 1 - H(2p)$ (and in particular, a code that achieves the rate $R = 1 - H(2p) - \epsilon$ for every $\epsilon > 0$, namely a code that achieves the Gilbert Varshamov bound) is evasive against the class of all channels. This is because with high probability a random word Z has relative distance larger than $2p$ from any codeword. This means that if (an unbounded channel) examines Z and induces p relative errors, the corrupted word is still not within distance p to a codeword, and therefore will be rejected by a decoding algorithm that decodes from p relative errors.*

Indeed, one of the keys to our unique decoding algorithm is a code with rate approaching $1 - H(p)$ (rather than $1 - H(2p)$) while still achieving evasiveness, and decoding from BSC_p.

3.1 Road map for this section

Following [GS16, SS16, KSS19] we will require a code with many additional properties (in addition to having rate $1 - H(p)$ from BSC_p). In this paper, we introduce the notion of evasiveness, and will also require that the code is evasive. Following [GS16] our starting point are concatenated codes for BSC_p channels, that for every $\epsilon > 0$ achieve rate $R \geq 1 - H(p) - \epsilon$. Such codes Enc are constructed by concatenating:

- An outer code $\mathrm{Enc}_{\mathrm{out}}$ with rate $1 - \epsilon/10$ that is decodable from $\lambda = \Omega(\epsilon)$ errors. The alphabet of this outer code (which determines the message length of the inner code) is a constant that depends on ϵ.
- A binary inner code $\mathrm{Enc}_{\mathrm{in}}$ with constant message length that decodes from BSC_p and has rate $1 - H(p) - \epsilon/10$. The existence of such a code follows by the probabilistic method, and the decoding algorithm is maximum likelihood decoding.

This gives an explicit code from BSC_p, where the decoding algorithm for the concatenated code is the concatenated decoding algorithm from Definition 2.15. We will specify this more formally below in Theorem 3.10, and in fact, some additional properties of this code is required by [KSS19] and also here. We start by focusing on the notion of evasiveness and how to achieve it. We will list the additional required properties of the code as we go along.

Evasiveness of (concatenated) codes. In Section 3.2 we will show that this decoding algorithm is evasive. More precisely, during the computation of concatenated decoding on a received word v, the concatenated decoding algorithm Dec also computes $\mathrm{outdist}(v)$ (which is the relative outer distance from Definition 2.15).

The relative outer distance of v is the fraction of errors that were corrected by the decoding algorithm $\mathrm{Dec}_{\mathrm{out}}$ of the outer code. In our setup, when decoding from BSC_p, the outer distance will be smaller than $\frac{1}{2} \cdot \lambda$ with high probability.

We will show that the probability that a small space channel C can make the decoding algorithm have outer distance at most $\leq 0.99\lambda$ is small. This means that the code Enc can be made evasive (by rejecting whenever the outer distance during decoding is larger than $\frac{1}{2} \cdot \lambda$).

Concatenated codes for BSC and related variants. In Section 3.3 we review the constructions of concatenated codes for BSC_p. We observe that we can take the outer code to be the high rate locally correctable and testable code of Kopparty, Meir, Ron-Zewi and Saraf [KMRS17]. Furthermore, following [Smi07, GS16, KSS19] we observe that the construction is also decodable from t-wise independent errors (that is defined formally in that section). This part is identical to [KSS19] (except for the different choice of outer code).

Approximating the outer distance by a small space ROBP. We will later be concerned with a "pseudorandom version" of the evasiveness experiment (which was explained in Section 1.3) in which rather than choosing a uniform $Z \leftarrow U_n$, the string Z is chosen by $Z \leftarrow G(U_d)$ where G is a PRG for small space ROBPs.

This version of the experiment will be crucial in our actual construction, as we will want to argue that except for the correct message, many other messages that are considered in the list decoding algorithm are obtained by decoding pseudorandom codewords. We will want to argue that evasiveness (on uniform strings) implies evasiveness on pseudorandom strings. This will enable us to argue that many incorrect messages are rejected by the decoding algorithm (and will be a key step in achieving unique decoding).

At first glance, it may appear that because C is a space bounded channel, and G fools small space ROBPs, then evasiveness implies evasiveness on pseudorandom strings. This isn't the case, because in the evasiveness experiment, one needs to run the decoding algorithm in order to decide whether or not the channel succeeds (and decoding algorithms are inherently not implemented by small space ROBPs).

Fortunately, the way we presented our notion of evasiveness *does not* require the full power of decoding. Instead, it is sufficient to compute the outer distance $\mathrm{outdist}$ by a small space ROBP.

In Section 3.4 we show that if the outer code $\mathrm{Enc}_{\mathrm{out}}$ is *locally correctable* and *locally testable* then the outer distance $\mathrm{outdist}$ can be approximated by a small space ROBP (where the space depends on the number of queries in the local correcting and locale testing algorithm). This will be sufficient to argue later on that evasiveness implies evasiveness on pseudorandom strings.

3.2 Evasiveness of concatenated codes

In Theorem 3.3 below, we state conditions under which a concatenated code is evasive. For this statement, we need the following notion of a "decoding reach".

Definition 3.2 (Decoding reach of a code)**.** *For a pair of functions* $\mathrm{Enc} : \{0,1\}^k \to \{0,1\}^n$ *and* $\mathrm{Dec} : \{0,1\}^n \to \{0,1\}^k$*, the* **relative decoding reach** *of* Enc *and* Dec *is*

$$\max_{v \in \{0,1\}^n} \delta(v, \mathrm{Enc}(\mathrm{Dec}(v))).$$

Note that a code that decodes from λ-relative errors, can w.l.o.g. have relative decoding reach at most λ (by re-encoding the decoded message and computing the distance). Note also that if $\mathrm{Enc} = \mathrm{Enc}_{\mathrm{out}} \circ \mathrm{Enc}_{\mathrm{in}}$ is a concatenated code, and Dec is the concatenated decoding algorithm, then the relative decoding reach of Enc and Dec is bounded by the sum of the relative decoding reaches of the outer and inner codes.

3.2.1 Statement of evasiveness theorem

In this section, we state and prove a theorem showing that any concatenated code with certain parameters is evasive. In our application, we will need a stronger evasiveness property (than the one stated before) that holds even if the channel C is allowed to choose a permutation $\sigma : [n] \to [n]$ that is applied to $V = Z \oplus C(Z)$ before it is decoded.

The theorem below shows that under certain conditions, a small space channel that induces few errors cannot make the concatenated decoding algorithm decode with small relative outer distance. This means that we can obtain evasiveness of Dec by rejecting strings $v \in \{0,1\}^n$ for which $\mathrm{outdist}(v)$ is large. (Jumping ahead we comment that when decoding from a corrupted codeword (rather than a corrupted uniform string) we will have the property that the relative outer distance is small, and this will allow us to distinguish between the case that the channel was applied on a uniform string, from the case that the channel was applied on a codeword).

Theorem 3.3 (Evasiveness of concatenated codes)**.** *For every constant $\gamma_1 > 0$, there exists a constant $\gamma_2 > 0$ such that for every sufficiently large n_{out} and a concatenated code $\mathrm{Enc} = \mathrm{Enc}_{\mathrm{out}} \circ \mathrm{Enc}_{\mathrm{in}} : \{0,1\}^n \to \{0,1\}$ with $n = n_{\mathrm{out}} \cdot n_{\mathrm{in}}$ such that:*

- $\mathrm{Enc}_{\mathrm{out}} : \{0,1\}^{k_{\mathrm{out}}} \to (\{0,1\}^{\log q_{\mathrm{out}}})^{n_{\mathrm{out}}}$ *that is decodable from λ relative errors by a decoding algorithm* $\mathrm{Dec}_{\mathrm{out}} : (\{0,1\}^{\log q_{\mathrm{out}}})^{n_{\mathrm{out}}} \to \{0,1\}^{k_{\mathrm{out}}}$.
- $\mathrm{Enc}_{\mathrm{in}} : \{0,1\}^{k_{\mathrm{in}}=\log q_{\mathrm{out}}} \to \{0,1\}^{n_{\mathrm{in}}}$ *and* $\mathrm{Dec}_{\mathrm{in}} : \{0,1\}^{n_{\mathrm{in}}} \to \{0,1\}^{k_{\mathrm{in}}}$ *is a function such that* $\mathrm{Enc}_{\mathrm{in}}, \mathrm{Dec}_{\mathrm{in}}$ *have relative decoding reach β.*

Let Dec *and* $\mathrm{outdist}$ *be the concatenated decoding algorithm and relative outer distance from Definition 2.15. If $\lambda + \beta + p \leq \frac{1}{2} - \gamma_1$ and $s \leq \gamma_2 \cdot \lambda \cdot n_{\mathrm{out}}$ then for every channel $C : \{0,1\}^n \to \{0,1\}^n$ such that $C \in \mathrm{Spc}_p^s$ and every permutation $\sigma : [n] \to [n]$, if we choose $Z \leftarrow U_n$ and set $X = \sigma(Z \oplus C(Z))$, we have that:*

$$\Pr[\mathrm{outdist}(X) \leq 0.99 \cdot \lambda] \leq 2^{-\Omega_{\gamma_1}(\lambda \cdot n_{\mathrm{out}})}$$

In the remainder of this section we prove Theorem 3.3. A high level explanation of this argument was given in Section 1.3.2.

3.2.2 Concatenated codes are evasive: proof of Theorem 3.3

We will divide $Z \in \{0,1\}^n$ into blocks. Specifically, let $u = \frac{100 n_{\mathrm{in}}}{\lambda}$ and $v = n/u = \frac{\lambda \cdot n_{\mathrm{out}}}{100}$. For a string $y \in \{0,1\}^n$ and $i \in [u]$, we use $y[i]$ to denote the v bit long, i'th block of y, namely $y_i = y_{(i-1)\cdot v+1}, \ldots, y_{(i-1)\cdot v+v}$. We now define the probability space that will be used in the proof. We refer to this experiment as expr.

- We choose: $Z \leftarrow U_n$, and set:
- $E = C(Z)$ to be the error induced by C.
- Let q_0 be the starting state of C, and for $i \in [u]$, let $Q_i \in \{0,1\}^s$ be the state that C arrives at after reading $Z[1], \ldots, Z[i]$. Let $Q \in (\{0,1\}^s)^u$ be defined by $Q = (Q_1, \ldots, Q_u)$.
- $Y = Z \oplus E$. For every $i \in [u]$, let $Y(i)$ be the string obtained from Y by replacing the i'th block with zeros. Namely, $Y(i)[j] = Y[j]$ for $j \neq i$ and $Y(i)[i] = 0^v$.
- $X = \sigma(Y)$, and for every $i \in [u]$, $X(i) = \sigma(X(i))$.
- $M = \mathrm{Dec}(X)$, and for every $i \in [u]$, $M(i) = \mathrm{Dec}(X(i))$.
- $W = \sigma^{-1}(\mathrm{Enc}(M))$, and for every $i \in [u]$, $W(i) = \sigma^{-1}(\mathrm{Dec}(M(i)))$.

In addition to the probability space of the experiment expr, for every $q \in (\{0,1\}^s)^u$ such that $\Pr[Q = q] > 0$, we consider the probability space $\mathrm{expr}^q = (\mathrm{expr}|Q = q)$. The random variables in this experiment will be denote by adding the superscript q. Thus, for example, $Z^q = (Z|Q = q)$, and $E^q = (E|Q = q) = C(Z^q)$.

We start by proving several claims about these experiments.

Claim 3.4. *The following holds:*

- *For every q such that $\Pr[Q = q] > 0$, $Z^q[1], \ldots, Z^q[u]$ are independent.*
- *For every q such that $\Pr[Q = q] > 0$, there exist functions $C^q_1, \ldots, C^q_u$ such that for every $i \in [u]$, $E[i] = C^q_i(Z^q[i])$.*
- *Let $\rho = \gamma_2 \cdot \lambda \cdot n_{\mathrm{out}}$. With probability at least $1 - u\rho$ over the choice of $q \leftarrow Q$, we have that q is "good", meaning that for every $i \in [u]$, $H_\infty(Z^q[i]) \geq v - s - \log(1/\rho) \geq \Omega(\lambda \cdot n_{\mathrm{out}})$.*

Proof. By the definition of ROBPs, for every $1 < i \leq u$, Q_i is determined by Q_{i-1} and $Z[i-1]$. For $i \in [u]$ and $q \in (\{0,1\}^s)^u$, let $D^q_i : \{0,1\}^v \to \{0,1\}$ be the space s ROBP that on input $x \in \{0,1\}^v$, interprets x as the i'th block of an n bit input string to C, and applies C on $x \in \{0,1\}^v$ starting as if C begins at the i'th block (that is, at layer $(i-1) \cdot v + 1$) from the state q_{i-1}. The ROBP D^q_i accepts x if this simulation concludes in state q_i. It follows that:

$$Z^q = (Z[1]|D^q_1(Z[1]) = 1), \ldots, (Z[u]|D^q_u(Z[u]) = 1)$$

Showing that the blocks of Z^q are independent, proving the first item.

Note that by the definition of ROBPs, $E[i]$ is determined by Q_{i-1} and $Z[i]$. This means that $E^q[i]$ is determined by $Z^q[i]$ (as Q^q_i is fixed to q_i) proving the second item.

Note that the definition of the ROBP D^q_i depends only on $q_1, \ldots, q_i$ and not on $q_{i+1}, \ldots, q_u$. Thus, we will allow ourselves to write $D^{q_1,\ldots,q_i}_i$, and this is well defined.

We say that $q \in (\{0,1\}^s)^j$ is good at position $i \leq j$ if

$$\Pr[D^q_i(U_v) = 1] \geq \rho \cdot 2^{-s}.$$

We say that $q \in (\{0,1\}^s)^u$ is good, if it is good at every position $i \in [u]$. We claim that:

$$\Pr[Q \text{ is good}] \geq 1 - u \cdot \rho.$$

For this, it is sufficient to prove that for every $i \in [u]$ and every $q_1, \ldots, q_{i-1} \in \{0,1\}^s$ such that $(q_1, \ldots, q_{i-1})$ is good at positions $1, \ldots, i-1$, we have that:

$$\Pr[(Q_1, \ldots, Q_i) \text{ is not good at position } i | Q_1 = q_1, \ldots, Q_{i-1} = q_{i-1}] \leq \rho.$$

In order to show this, for fixed $q_1, \ldots, q_{i-1}$, we define:

$$B_{q_1,\ldots,q_{i-1}} = \left\{ q_i : \Pr[D^{(q_1,\ldots,q_{i-1},q_i)}_i(U_v) = 1] < \rho \cdot 2^{-s} \right\},$$

and by a union bound over at most 2^s choices of $q_i \in B_{q_1,\ldots,q_{i-1}}$ we have that:

$$\Pr[Q_i \in B_{q_1,\ldots,q_{i-1}} | Q_1 = q_1, \ldots, Q_{i-1} = q_{i-1}] \leq 2^s \cdot (\rho \cdot 2^{-s}) = \rho.$$

Finally, for every good $q \in (\{0,1\}^s)^u$, we have that:

$$H_\infty(Z^q[i]) = H_\infty(Z[i]|D^q_i(Z[i]) = 1) \geq v - \log(\frac{1}{\rho \cdot 2^{-s}}) = v - s - \log(1/\rho).$$

□

Claim 3.5. *For every good $q \in (\{0,1\}^s)^u$, and every $i \in [u]$, $M^q(i)$ is independent of $Z^q[i]$.*

Proof. The message $M^q(i)$ is a function of $Y^q(i)$, that was obtained after the i'th block of Y was erased, and is therefore determined by:

$$Y^q[1], \ldots, Y^q[i-1], Y^q[i+1], \ldots, Y^q[u].$$

We have that $Y^q = Z^q \oplus E^q$, meaning that $Y^q[i] = Z^q[i] \oplus E^q[i]$. By the second item of Claim 3.4, for every $i \in [u]$, $E^q[i]$ is determined by $Z^q[i]$. This means that $Y^q[i]$ is determined by $Z^q[i]$. This means that $M^q(i)$ is determined by:

$$Z^q[1], \ldots, Z^q[i-1], Z^q[i+1], \ldots, Z^q[u],$$

and the latter is independent of $Z^q[i]$ by the first item of Claim 3.4. □

Claim 3.6. *For every $\gamma_1 > 0$, every good $q \in (\{0,1\}^s)^u$, and every $i \in [u]$,*

$$\Pr[\delta(W^q(i)[i], Z^q[i]) \leq \frac{1}{2} - \gamma_1] \leq 2^{-\Omega_{\gamma_1}(\lambda \cdot n_{\text{out}})}.$$

Proof. Let us imagine that the probability space expr^q contains an additional random variable U, which is uniform on $\{0,1\}^v$, and independent of all random variables. As $W^q(i)[i]$ is independent of U, we have that the expected relative distance between $W^q(i)[i]$ and U is half. By a Chernoff bound:

$$\Pr[\delta(W^q(i)[i], U) \leq \frac{1}{2} - \gamma_1] \leq 2^{-\Omega(\gamma_1^2 \cdot v)} \leq 2^{-\Omega_{\gamma_1}(\lambda \cdot n_{\text{out}})}.$$

The claim will follow if we argue that:

$$\Pr[\delta(W^q(i)[i], Z^q[i]) \leq \frac{1}{2} - \gamma_1] \leq \Pr[\delta(W^q(i)[i], U) \leq \frac{1}{2} - \gamma_1] \cdot 2^{2 \cdot \gamma_2 \cdot \lambda \cdot n_{\text{out}}}. \tag{1}$$

We first note that by Claim 3.5, just like U, $Z^q[i]$ is independent of $W^q(i)[i]$. Furthermore, by Claim 3.4 we have that $H_\infty(Z^q[i]) = v - \log(\frac{2^s}{\rho})$. It follows that for every $x \in \{0,1\}^v$,

$$\Pr[Z^q[i] = v] \leq \Pr[U = v] \cdot \frac{2^s}{\rho} \leq \Pr[U = v] \cdot 2^{2 \cdot \gamma_2 \cdot \lambda \cdot n_{\text{out}}},$$

and this implies (1) and the claim. □

Claim 3.7. *For every good $q \in \{0,1\}^u$, if $\text{outdist}(X^q) \leq 0.99 \cdot \lambda$ then for every $i \in [u]$, $M^q(i) = M^q$.*

Proof. We have chosen $u = \frac{100 n_{\text{in}}}{\lambda}$, so that $0.99 \cdot \lambda \leq \lambda - \frac{n_{\text{in}}}{u}$. By definition, for every $i \in [u]$, $\Delta(Y^q(i), Y^q) \leq v$, meaning that $\delta(X^q(i), X^q) = \delta(Y^q(i), Y^q) \leq \frac{v}{n} = \frac{1}{u}$. By Lemma 2.16 we get that if $\text{outdist}(X^q) \leq \lambda - \frac{n_{\text{in}}}{u}$ and $\delta(X^q, X^q(i)) \leq \frac{1}{u}$, then $\text{Dec}(X^q) = \text{Dec}(X^q(i))$, meaning that $M^q = M^q(i)$, for every $i \in [u]$. □

Claim 3.8. *For every $\gamma_1 > 0$, and every good $q \in \{0,1\}^u$,*

$$\Pr[\delta(W^q, Z^q) > \frac{1}{2} - \gamma_1] > \Pr[\text{outdist}(X^q) \leq 0.99\lambda] - 2^{-\Omega_{\gamma_1}(\lambda \cdot n_{\text{out}})}.$$

Proof. By Claim 3.6 and a union bound over all $i \in [u]$ we have that:

$$\Pr[\exists i \in [u] : \ \delta(W^q(i)[i], Z^q[i]) \leq \frac{1}{2} - \gamma_1] \leq u \cdot 2^{-\Omega_{\gamma_1}(\lambda \cdot n_{\text{out}})} = 2^{-\Omega_{\gamma_1}(\lambda \cdot n_{\text{out}})}.$$

By Claim 3.7, whenever the event $\{\text{outdist}(X^q) \leq 0.99 \cdot \lambda\}$ occurs, we have that for every $i \in [u]$, $M^q(i) = M^q$, which implies that $W^q(i) = W^q$, and in particular $W^q(i)[i] = W^q[i]$. When this event occurs, we have that for every $i \in [u]$, $\delta(W^q[i], Z^q[i]) = \delta(W^q(i)[i], Z^q[i])$, meaning that

$$\delta(W^q, Z^q) = \frac{1}{u} \cdot \sum_{i \in [u]} \delta(W^q(i)[i], Z^q[i]),$$

and the claim follows. □

We are finally ready to prove Theorem 3.3.

Proof. (of Theorem 3.3) By Claim 3.4, $\Pr[Q \text{ is not good}] < u \cdot \rho \leq 2^{-\Omega_{\gamma_1}(\lambda \cdot n_{\text{out}})}$. Therefore,

$$\Pr[\text{outdist}(X) \leq 0.99 \cdot \lambda \cap Q \text{ is good}] > \Pr[\text{outdist}(X) \leq 0.99 \cdot \lambda] - 2^{-\Omega_{\gamma_1}(\lambda \cdot n_{\text{out}})}.$$

By averaging, there exists a good q such that:

$$\Pr[\text{outdist}(X^q) \leq 0.99 \cdot \lambda] > \Pr[\text{outdist}(X) \leq 0.99 \cdot \lambda] - 2^{-\Omega_{\gamma_1}(\lambda \cdot n_{\text{out}})}.$$

For this q, by Claim 3.8 we have that:

$$\Pr[\delta(W^q, Z^q) > \frac{1}{2} - \gamma_1] > \Pr[\text{outdist}(X) \leq 0.99 \cdot \lambda] - 2^{-\Omega_{\gamma_1}(\lambda \cdot n_{\text{out}})}.$$

We claim that:

$$\Pr[\delta(W^q, Z^q) > \frac{1}{2} - \gamma_1] = 0,$$

which implies $\Pr[\text{outdist}(X) \leq 0.99 \cdot \lambda] \leq 2^{-\Omega_{\gamma_1}(\lambda \cdot n_{\text{out}})}$ and proves the theorem. The claim follows because by the triangle inequality:

$$\delta(W^q, Z^q) \leq \delta(W^q, Y^q) + \delta(Y^q, Z^q) \leq \lambda + \beta + p \leq \frac{1}{2} - \gamma_1.$$

The second inequality holds (with probability one) because $\delta(Y^q, Z^q) \leq p$, and $\delta(W^q, Z^q)$ is bounded by the relative decoding reach of Enc, Dec, which in turn is bounded by the sum of the relative decoding reaches of Dec_{out} (which is λ) and Dec_{in} (which is β). □

3.3 Concatenated codes for binary-symmetric channels and related variants

We will make use of known constructions of codes for binary symmetric channels.

Definition 3.9 (Binary symmetric channel)**.** *Let BSC_p^n denote the distribution over n bit strings in which individual bits are i.i.d. and each is one with probability p.*

There are constructions of explicit (and even linear time) codes with rate approaching $1 - H(p)$ that are decodable from BSC_p^n with very high success probability [For65, GI05].

Following [Smi07, GS16, KSS19] we are interested in codes for a somewhat similar scenario of "t-wise independent errors" in which the error distribution is obtained by:

- Taking an arbitrary string $e \in \{0,1\}^n$ with $wt(e) \leq p$.
- Taking an (ϵ, t)-wise independent permutation $\pi : \{0,1\}^d \times [n] \to [n]$.
- The error distribution is $e' = \pi_{U_d}(e)$.

In this distribution, each noise bit has probability p to be one, and the bits have "negative correlation". This negative correlation can be used to argue that current constructions for BSC_p (which are based on concatenating a high rate outer code that decodes few relative errors, with a random inner code with rate approaching $1 - H(p)$ on constant block length) also work in this scenario.

We will require a code that (in addition to being decodable from t-wise independent errors in the sense explained above) also has many additional properties. Theorem 3.10 below was proven in [KSS19] for a different choice of outer code, and very similar arguments were previously made by Smith [Smi07] and in an early version of [GS16]. The theorem revisits the code construction of [For65, GI05] for BSC_p^n, and observes that the constructed concatenated code has some properties that we will use later on. The theorem is also stated so that Theorem 3.3 applies, and that in addition to all the properties listed in Theorem 3.10, the code is also *evasive*.

The statement of Theorem 3.10 below is essentially identical to that in [KSS19]. A difference is that we will use an outer code that is also locally correctable and locally testable. Such codes were recently constructed by Kopparty, Meir, Ron-Zewi and Saraf [KMRS17] and decode from the same fraction of errors than the codes of Guruswami and Indyk [GI05] which was proved in [KSS19] (and so the proof of [KSS19] also proves this theorem).

Theorem 3.10 (Similar to [KSS19])**.** *For every constant $0 < p < 1/2$, and every sufficiently small constant $\epsilon > 0$, there exist integer constants $k_{\mathrm{in}}, n_{\mathrm{in}}, q_{\mathrm{out}}$ and real constants $\lambda_1, \lambda_2, \lambda_3 > 0$ such that $k_{\mathrm{in}} = \log q_{\mathrm{out}} \leq \mathrm{poly}(\frac{1}{\epsilon})$, and for infinitely many choices of n_{out} there exist functions:*

- $\mathrm{Enc}_{\mathrm{out}} : \{0,1\}^{k_{\mathrm{out}}} \to (\{0,1\}^{\log q_{\mathrm{out}}})^{n_{\mathrm{out}}}$,
- $\mathrm{Enc}_{\mathrm{in}} : \{0,1\}^{k_{\mathrm{in}}} \to \{0,1\}^{n_{\mathrm{in}}}$,

such that:

- $R_{\mathrm{out}} = \frac{k_{\mathrm{out}}}{n_{\mathrm{out}} \cdot \log q_{\mathrm{out}}} \geq 1 - \frac{\epsilon}{10}$, *and* $\mathrm{Enc}_{\mathrm{out}}$ *is the code from Theorem 2.12, that is decodable and non-adaptively locally correctable from λ_1 relative errors, the local corrector and local tester make $Q = (n_{\mathrm{out}})^{o(1)}$ queries, and runs in time $\mathrm{poly}(Q)$.*
- $R_{\mathrm{in}} = \frac{k_{\mathrm{in}}}{n_{\mathrm{in}}} \geq 1 - H(p) - \epsilon/10$, *and* $\mathrm{Enc}_{\mathrm{in}}$ *is decodable from* $\mathrm{BSC}_p^{n_{\mathrm{in}}}$ *with probability $1 - 2^{-\lambda_2 \cdot n_{\mathrm{in}}}$. This decoding is achieved by a function* $\mathrm{Dec}_{\mathrm{in}}$ *that implements "maximum likelihood decoding" from relative distance $p + \epsilon''$, where $\epsilon'' > 0$ is a constant that depends on ϵ.*
- *Consequently, setting $n = n_{\mathrm{out}} \cdot n_{\mathrm{in}}$, and $q_{\mathrm{in}} = 2$, the concatenated code* $\mathrm{Enc} = \mathrm{Enc}_{\mathrm{out}} \circ \mathrm{Enc}_{\mathrm{in}} : \{0,1\}^{k_{\mathrm{out}}} \to \{0,1\}^n$ *is well defined, has rate $R = \frac{k_{\mathrm{out}}}{n} \geq 1 - H(p) - \epsilon$, and is encodable and in time $O(T(n))$ for a universal polynomial T, and the constant c hidden in the $O(\cdot)$ depends on ϵ, specifically $c(\epsilon) = 2^{\mathrm{poly}(1/\epsilon)}$. The concatenated decoding algorithm also runs in time $O(T(n))$.*
- *Let $t \leq n^{0.1}$, and let $\pi : \{0,1\}^d \times [n] \to [n]$ be a $(2^{-10 \cdot t}, t)$-wise independent permutation. Let $m \in \{0,1\}^{k_{\mathrm{out}}}$, and let $A_m^{\lambda} : \{0,1\}^n \to \{0,1\}$ be the function that on input $e' \in \{0,1\}^n$, outputs one iff*

$$\left| \left\{ i \in [n_{\mathrm{out}}] : \mathrm{Dec}_{\mathrm{in}}^i(\mathrm{Enc}(m) \oplus e') \neq \mathrm{Enc}_{\mathrm{out}}(m)_i \right\} \right| \leq \lambda \cdot n_{\mathrm{out}}.$$

  Note that for $\lambda \leq \lambda_1$, if $A_m^{\lambda}(e') = 1$ then:

  - *Applying concatenated decoding on $v = \mathrm{Enc}(m) \oplus e'$ indeed recovers m.*
  - $\mathrm{outdist}(v) \leq \lambda$.

  For every $e \in \{0,1\}^n$ of Hamming weight at most pn,

$$\Pr[A_m^{\lambda_1/10}(\pi_{U_d}(e)) = 1] \geq 1 - 2^{-\lambda_3 \cdot t}.$$

- *Consequently, for every $e \in \{0,1\}^n$ of Hamming weight at most pn, the code* Enc *is decodable from $\pi_{U_d}(e)$ with probability $1 - 2^{-\lambda_3 \cdot t}$.*

The final item in Theorem 3.10 follows from the penultimate item. However, as in [KSS19], for our purposes, the final item will not be sufficiently strong, and we will need to use the penultimate item (as well as the previous items).

Preparing ahead, we list the following corollary of Theorem 3.10.

Claim 3.11. *Using the notation of Theorem 3.10, for every $\lambda \leq \lambda_1$, every message $m \in \{0,1\}^{k_{\text{out}}}$ and every error vector $e' \in \{0,1\}^n$, if $A^{\lambda}_m(e') = 1$ then* $\text{outdist}(\text{Enc}(m) \oplus e') \leq \lambda$.

Proof. We have that $A^{\rho}_m(e') = 1$ and $\lambda \leq \lambda_1$. This implies that for $z = \text{Enc}(m) \oplus e'$ it holds that $\text{Dec}(z) = m$. This gives that $\text{outdist}(z) \leq \lambda$. □

3.4 Approximating the outer distance by a small space ROBP.

In this section we show that the relative outer distance of the concatenated code Enc from the previous section can be approximated by a small space (randomized) ROBP. Thinking ahead, we will show that this holds even if the ROBP reads the bits in a different order, after a permutation σ was applied on the input v. In the statement below we approximate $\min(\text{outdist}, \lambda_1)$, as when using this approximation we will only care whether the real outer distance is smaller than say $\frac{3}{4} \cdot \lambda_1$.

Lemma 3.12. *Let* Enc *be the code constructed in Theorem 3.10 using all the choices in the statement of Theorem 3.10. There exists a function $a(n) = n^{o(1)}$ such that for every $s(n) \geq a(n)$, there exists a distribution $\bar{D}$ over ROBPs of space $s(n)$ such that for every $v \in \{0,1\}^n$, and every permutation $\sigma : [n] \to [n]$:*

$$\Pr_{D \leftarrow \bar{D}}[|D(\sigma(v))) - min(\text{outdist}(v), \lambda_1)| > \frac{\lambda_1}{100}] \leq 2^{-\frac{s(n)}{a(n)}}.$$

In the remainder of this section we prove Lemma 3.12. Loosely speaking, the lemma follows because a small space ROBP can decode the inner code (which has constant size alphabet) and simulate a nonadaptive local testing and correcting algorithms of the outer code, allowing it to compute the outer distance. Details follow.

For every $v \in \{0,1\}^n$, let

$$w(v) = \text{Dec}^{(1)}_{\text{in}}(v), \ldots, \text{Dec}^{(n_{\text{out}})}_{\text{in}}(v)$$

denote the first step of the concatenated decoding algorithm. We first observe that by investing n_{in} queries, one can simulate oracle access to $w(v)$ when given oracle access to v.

Claim 3.13. *There is an algorithm A that given input $i \in [n_{\text{out}}]$ and oracle access to $\sigma(v) \in \{0,1\}^n$, computes $w(v)_i = \text{Dec}^{(i)}_{\text{in}}(v)$ using n_{in} non-adaptive queries.*

Proof. The algorithm A (that knows σ) simply queries $\sigma(v)$ at the bits corresponding to $v_{(i-1)\cdot n_{\text{in}}+1}, \ldots, v_{i \cdot n_{\text{in}}}$ and computes:

$$\text{Dec}^{i}_{\text{in}}(v) = \text{Dec}_{\text{in}}(v_{(i-1)\cdot n_{\text{in}}+1}, \ldots, v_{i \cdot n_{\text{in}}}),$$

□

We now observe that using local-correcting and local-testing with relative distance λ_1, it is possible to check whether a word $w \in (\{0,1\}^{\log q_{\text{out}}})^{n_{\text{out}}}$ is within distance λ_1 to a codeword of Enc_{out}.

Claim 3.14. *There is an algorithm B that given oracle access to $w \in (\{0,1\}^{\log q_{\text{out}}})^{n_{\text{out}}}$ makes $O(Q^2 \cdot \log Q) = n^{o(1)}$ non-adaptive queries, and:*

- *If there exists* $m \in \{0,1\}^{k_{\text{out}}}$ *such that* $\delta' = \delta(\text{Enc}_{\text{out}}(m), w) \leq \lambda_1$ *then* $\Pr[|B^w - \delta'| \leq \frac{\lambda_1}{1000}] \geq \frac{2}{3}$.
- *Otherwise,* $\Pr[B^w = \lambda_1] \geq \frac{2}{3}$.

Proof. The algorithm B acts as follows:

- It simulates the local tester (reducing the failure probability from $1/3$ to $1/10$). This requires $O(Q)$ queries of the tester.
- Whenever the local tester makes a query i, the algorithm calls the local correcting algorithm with input i and oracle w, and returns the answer to the local tester. Each application of the local correcting algorithm will be amplified so that the error is less than $\frac{1}{Q^4}$, and requires $O(Q \log Q)$ queries.
- If the local tester rejects, then B answers λ_1.
- Choose $t = O(1/\lambda_1^2) = O(1)$ uniform indices $i_1, \ldots, i_t \in [n_{\text{out}}]$. For each $j \in [t]$, B queries w at i_j, and applies the local correcting algorithm with input i_j to obtain a string a_j.
- Compute the fraction $\bar{\delta}$ of $j \in [t]$ for which w_{i_j} and the a_j differ.
- Output $\min(\bar{\delta}, \lambda_1)$.

The total number of queries of algorithm B is $O(Q^2 \log Q) = n^{o(1)}$ as required.

The first item follows because if there exists $m \in \{0,1\}^{k_{\text{out}}}$ such that $\delta' = \delta(\text{Enc}_{\text{out}}(m), w) \leq \lambda_1$ then by a union bound, with probability at least $1 - 1/Q$, each of the queries of local correcting algorithm returns the correct symbol in $\text{Enc}_{\text{out}}(m)$. When this occurs, the local testing algorithm accepts with probability at least $9/10$, and by a Chernoff bound, B gives a suitable approximation $\bar{\delta}$ of δ'.

The second item follows because we can imagine that there is a pre-processing step in which the local correcting algorithm is applied not only on a subset of the n_{out} indices, but in fact, for every $i \in [n_{\text{out}}]$ (with independent choices of random coins that is chosen in advance and fixed). For every fixed random coins (for all these applications) this process induces a word w' which at index i is the output of the local correcting algorithm. Whenever, the local correcting algorithm makes a real query i, we can imagine that it is answered by w'_i. This means that for every fixed choice of the random coins of all applications of the local correcting algorithm in algorithm B, the local testing algorithm is receiving oracle access to w'. For every such word w' we have that the local tester rejects w' with probability $9/10$ if it is not within relative distance λ_1 to a codeword of Enc_{out}. This means that if w' is not rejected and is a codeword, then w' is not within relative distance λ_1 to w (as w isn't within distance λ_1 to any codeword). This means that if the tester does not reject, then the approximated distance $\bar{\delta}$ will be at least $\frac{99}{100} \cdot \lambda_1$. Overall, with probability at least $2/3$, the algorithm B outputs $\bar{\delta}$ which is very close to λ_1. □

We are ready to prove Lemma 3.12.

Proof. (of Lemma 3.12) We first consider an amplified version of the algorithm B of Claim 3.4, where the error is amplified to $2^{-s(n)/a(n)}$. This takes $\frac{n^{o(1)} \cdot s(n)}{a(n)}$ non-adaptive queries.

We then consider a version of B which receive oracle access to v and uses Claim 3.13 to simulate access to $w(v)$. This multiplies the number of queries by n_{in} and takes $n_{\text{in}} \cdot \frac{n^{o(1)} \cdot s(n)}{a(n)} \leq s(n)$ non-adaptive queries for some choice of $w(n) = n^{o(1)}$. The obtained algorithm computes the approximation that is required in the lemma.

Finally, a randomized ROBP with space $s(n)$ (which is a distribution over space $s(n)$ ROBPs) that reads $\sigma(v)$ (and knows σ) can simulate an algorithm with non-adaptive oracle access to v by tossing the coins in advance, and capturing the required bits while reading $\sigma(v)$. This gives an ROBP with space $s(n)$ that computes the desired approximation. □

4 Stochastic control codes

An important ingredient in the previous constructions of [GS16, SS16, KSS19] is a special type of a stochastic code that is called "control code". This is a stochastic code $\mathrm{Enc} : \{0,1\}^k \times \{0,1\}^d \to \{0,1\}^n$ which (in addition to certain decoding properties) also has some pseudorandom properties. In our construction of stochastic codes for space bounded channels, the control code will be used to encode the seed s of the main construction, using randomness r. It will therefore be convenient to call the message, and randomness of the control code s and r respectively.

In this paper we will require (and make use of) additional properties of control codes that were not required in [GS16, SS16, KSS19]. Fortunately, we will show that the construction of control codes in [KSS19] (which is based on a linear error correcting code called there "raw Reed-Solomon code" also has these additional properties).

Repetition decoding and it's usefulness. The main new property that we require is "repetition decoding". This says that for every seed s that is encoded ℓ times, using an arbitrary choice od $r_1, \ldots, r_\ell$, to produce the "repetition codeword",

$$\mathrm{Enc}_{\mathrm{ctrl}}(s, r_1), \ldots, \mathrm{Enc}_{\mathrm{ctrl}}(s, r_\ell),$$

we require that given any sequence $v_1, \ldots, v_\ell \in \{0,1\}^n$, such that the average of the relative distances between $\mathrm{Enc}_{\mathrm{ctrl}}(s, r_i)$ and v_i is smaller than p, one can *uniquely decode* and obtain the original seed s.

This repetition decoding property is one of the keys that allow us to obtain unique decoding all the way up to $1/4$ in our final stochastic codes. More specifically, we will be able to obtain codes with repetition decoding up to $1/4$. This will be useful, because while an adversary that picks $v_1, \ldots, v_\ell$ can make a majority of the ℓ individual decodings decode to some $\bar{s} \neq s$ (by choosing a subset of more than half the indices, and injecting a large fraction of errors on indices in that subset) it cannot lead the repetition decoding algorithm to decode to a different string $\bar{s} \neq s$.

4.1 Definition and properties of control codes

We start by extending the notion of Hamming distance to stochastic codes. More precisely, in a stochastic code, each message s has many possible codewords $\mathrm{Enc}(s, r)$ for the 2^d choices of $r \in \{0,1\}^d$. The definition below says that the distance of a stochastic code is the minimum over all choices of s, s' and r, r' of the distance between $\mathrm{Enc}(s, r)$ and $\mathrm{Enc}(s', r')$.

Definition 4.1 (Hamming distance for stochastic codes)**.** *Let* $\mathrm{Enc} : \{0,1\}^k \times \{0,1\}^d \to \{0,1\}^n$ *be a function, and let* $s, s' \in \{0,1\}^k$ *and* $v \in \{0,1\}^n$.

- *We define* $\Delta^{\mathrm{Enc}}(s, v) = min_{r \in \{0,1\}^d}(\Delta(\mathrm{Enc}(s, r), v))$*, and* $\Delta^{\mathrm{Enc}}(v, s) = \Delta^{\mathrm{Enc}}(s, v)$ *(this is done, so that* Δ^{Enc} *is symmetric). We define* $\delta^{\mathrm{Enc}}(s, v) = \frac{\Delta^{\mathrm{Enc}}(s,v)}{n}$*, and* $\delta^{\mathrm{Enc}}(v, s) = \delta^{\mathrm{Enc}}(s, v)$.
- *We define* $\Delta^{\mathrm{Enc}}(s, s') = min_{r,r' \in \{0,1\}^d}(\Delta(\mathrm{Enc}(s, r), \mathrm{Enc}(s', r')))$*, and* $\delta^{\mathrm{Enc}}(s, s') = \frac{\Delta^{\mathrm{Enc}}(s,s')}{n}$.

It is standard that these definitions satisfy the triangle inequality.

- *The* **distance** *of* Enc *is* $\Delta(\mathrm{Enc}) = \min_{s,s' \in \{0,1\}^k} \Delta^{\mathrm{Enc}}(s, s')$ *and the* **relative distance** *of* Enc *is* $\delta(\mathrm{Enc}) = \min_{s,s' \in \{0,1\}^k} \delta^{\mathrm{Enc}}(s, s')$.

In the definition below we list several useful properties of control codes.

Definition 4.2 (Pseudorandom stochastic codes decodable from errors)**.** *Let* k, n, d *be parameters and let* $\mathrm{Enc} : \{0,1\}^k \times \{0,1\}^d \to \{0,1\}^n$ *be a function. We say that* Enc *is an encoding function for a stochastic code that is:*

- ϵ-**pseudorandom** *for a class $\mathcal{C}$ of functions from n bits to one bit, if for every $s \in \{0,1\}^k$, $\mathrm{Enc}(s, U_d)$ is ϵ-pseudorandom for $\mathcal{C}$.*
- **decodable from t errors**, *if $t \in [n]$, and there exists a function $\mathrm{Dec} : \{0,1\}^n \to \{0,1\}^k$ such that for every $s \in \{0,1\}^k$, and $v \in \{0,1\}^n$ such that $\Delta^{\mathrm{Enc}}(s, v) \leq t$, we have that $\mathrm{Dec}(v) = s$. We say that the code is* **decodable from t errors, with an a-approximation** *if in addition to s, Dec also outputs a number w such that $|w - \Delta(s, v)| \leq a$.*
- L-**list-decodable from t-errors**, *if the function Dec is allowed to output a list of messages of size at most L, and for every $s \in \{0,1\}^k$, and $v \in \{0,1\}^n$ such that $\Delta^{\mathrm{Enc}}(s, v) \leq t$, we have that $\mathrm{Dec}(v) \ni s$. We say that the code is L-***list-decodable from t errors, with an a-approximation** *if for every message s' in the output list, Dec also outputs a number w' such that $|w' - \Delta^{\mathrm{Enc}}(s', v)| \leq a$.*
- **repetition-decodable from t errors**, *if there exists a function rDec such that for every $s \in \{0,1\}^k$, if rDec receives as input a number $\ell \leq 2^n$ and $v_1, \ldots, v_\ell \in \{0,1\}^n$ such that $\frac{1}{\ell} \cdot \sum_{i \in [\ell]} \Delta^{\mathrm{Enc}}(s, v_i) \leq t$, we have that $\mathrm{rDec}(\ell, v_1, \ldots, v_\ell) = s$,*

In the definitions above, we use "δ **relative errors***" to mean "$t = \delta n$ errors", and "η-***relative approximation***" to mean "a-approximation for $a = \eta n$".*

A code has **encoding time** *[resp.* **decoding time***] $T(\cdot)$, if Enc [resp. Dec] can be computed in time $T(k + n + d)$. A code has* **repetition decoding amortized time** *$T(\cdot)$ if when given input $\ell, v_1, \ldots, v_\ell$, rDec can be computed in time $\ell \cdot T(k + n + d)$. (Naturally, this makes sense only for a family of encoding and decoding functions with varying block length n, message length $k(n)$ and seed length $d(n)$).*

4.2 Discussion and comparison to [GS16, SS16, KSS19]

Following [GS16], in the construction of stochastic codes for space bounded channels, given a message m and a seed s, we will use a control code to encode s. More specifically, we will choose seeds $r_1, \ldots, r_{n_{\mathrm{ctrl}}}$ at random, and we will put $\mathrm{Enc}(s, r_1), \ldots, \mathrm{Enc}(s, r_{n_{\mathrm{ctrl}}})$ as blocks in the final codeword. These (few) blocks are called "control blocks", and the rest of the blocks are called "data blocks".

Pseudorandomness. The notion of pseudorandomness was introduced by Guruswami and Smith [GS16]. Loosely speaking, pseudorandomness guarantees that when a channel C from $\mathcal{C}$ reads $\mathrm{Enc}(s, r)$ it learns no information about s. Furthermore, the construction will arrange it so that all blocks are pseudorandom, which loosely means that the channel cannot distinguish control blocks from non-control blocks.

Decoding and list-decoding. The notion of decoding considered in Definition 4.2 is tailored to decode even against Hamming channels. More precisely, the decoding algorithm is required to decode (or list decode) from a received word v that is close to some encoding $\mathrm{Enc}(s, r)$ in Hamming distance. Note however, that when Dec is applied on v, it is only required to retrieve s, and is not required to retrieve the seed r. (The usefulness of this notion was demonstrated in [SS16] and later used in [KSS19], whereas the initial paper of [GS16] required the decoding algorithm to also retrieve the seed r).

Approximation. We will use the construction of control codes by [KSS19]. In that construction decoding algorithms are not able to retrieve the seed r (although they can obtain a lot of information about it). This means that even when unique decoding $\mathrm{Dec}(v) = s$, we cannot necessarily efficiently compute the distance $\Delta^{\mathrm{Enc}}(s, v)$. (Of course, we can compute $\mathrm{Enc}(s, r)$ for all $r \in \{0,1\}^d$ and find the closest one, but this may take exponential time). We will need to be able to estimate these distances, and this is why we introduce the notion of an a-approximation (that was not used in previous works). We will observe that the decoding and list-decoding algorithms for the control code of [KSS19] indeed have such a α-relative approximation for $\alpha = o(1)$.

Repetition decoding. Finally, another component that we introduce in this paper is the notion of *repetition decoding* of a control code. This is done because if we restrict our attention to control blocks, this corresponds to encoding the same message s using many independent seeds. When decoding against space bounded channels, if we could identify which of the blocks are control blocks, then it makes sense to run repetition decoding on these blocks in order to obtain s. (Jumping ahead, we remark that identifying the control blocks will be quite tricky).

Using repetition decoding is a key idea in which our use of control codes deviates from previous work (and is crucial for achieving unique decoding, together with our approach to identify the control blocks).

We will argue below that the task of repetition decoding can be reduced to the task of list-decoding with a good approximation.

4.3 Constructions of control codes

In our main construction of stochastic codes for space bounded channels that induce p relative errors, we will require stochastic control codes that are pseudorandom, and repetition decodable from slightly more than p relative errors. The theorem below states the control code that will be used to prove Theorem 1.1.

Theorem 4.3 (Stochastic control codes with repetition decoding up to $\frac{1}{4}$)**.** *For every constant $\beta > 0$ there exists a constant $0 < \alpha \leq 0.1$ such that for every sufficiently large m, setting $n = (2^m - 1) \cdot m$, $k = n^\alpha$, $d = n \log n$, and $s = \frac{n^\alpha}{\log^3 n}$, there is a stochastic code $\mathrm{Enc} : \{0,1\}^k \times \{0,1\}^d \to \{0,1\}^n$ that is:*

- *Computable in time n^c for a constant c that does not depend on β.*
- *2^{-s}-pseudorandom for space s ROBPs.*
- *Repetition decodable from $\frac{1}{4} - \beta$ relative errors in amortized time n^c (and in particular decodable from $\frac{1}{4} - \beta$ relative errors in time n^c).*

4.3.1 Different tradeoffs in parameters

In the two theorems below, we consider different tradeoffs between the parameters (that lead to alternative tradeoffs in our main result).

Achieving almost linear time encoding and (randomized) decoding. We can push the running time of the encoding and repetition decoding to almost linear time. For this, we need to allow the repetition decoding algorithm to be randomized, and err with small probability.

Theorem 4.4 (Stochastic control codes with almost linear time encoding and repetition decoding)**.** *For every constant $\beta > 0$ there exists a constant $0 < \alpha \leq 0.1$ such that for every sufficiently large m, setting $n = ((2^m - 1) \cdot m)^c$ (for a universal constant c), $k = n^\alpha$, $d = n \log n$, and $s = \frac{n^\alpha}{\log^3 n}$, there is a stochastic code $\mathrm{Enc} : \{0,1\}^k \times \{0,1\}^d \to \{0,1\}^n$ that is:*

- *Computable in time $n \cdot \log^2 n$.*
- *2^{-s}-pseudorandom for space s ROBPs.*
- *Repetition decodable from $\frac{1}{4} - \beta$ relative errors by a randomized algorithm that runs in amortized time $O(n)$ and is guaranteed to perform repetition decoding correctly on any ℓ and $v_1, \ldots, v_\ell \in \{0,1\}^n$ with probability at least $1 - \ell \cdot 2^{-n^{1/c'}}$ where c' is a universal constant.*

Using a control code with a randomized repetition decoding algorithm in our main construction translates into a stochastic code for space bounded channels where the decoding algorithm is randomized. There is no reason not to allow this, see Remark 2.18.

Achieving codes that are pseudorandom for larger space for $p < \frac{1}{8}$. A weakness of Theorems 4.3 and 4.4 is that they obtain pseudorandomness for space roughly $s = n^\alpha$ where $\alpha > 0$ is an unspecified constant, that depends on β. (An inspection of the proof of [KSS19] reveals that $\alpha = \Omega(\beta^2)$). We can do better if $p < 1/8$ and obtain a code that is pseudorandom for space s that is roughly $n^{\frac{1}{2}}$.

Theorem 4.5 (Stochastic control codes for larger space with repetition decoding up to $\frac{1}{8}$)**.** *For every constant $0 < \alpha < \frac{1}{2}$, and for every sufficiently large m, setting $n = (2^m - 1) \cdot m$, $k = n^\alpha$, $d = n \log n$, and $s = \frac{n^\alpha}{\log^3 n}$, there is a stochastic code* $\mathrm{Enc} : \{0,1\}^k \times \{0,1\}^d \to \{0,1\}^n$ *that is:*

- *Computable in time n^c for a constant c.*
- *2^{-s}-pseudorandom for space s ROBPs.*
- *For every constant $p < 1/8$,* Enc *is repetition decodable from p relative errors in amortized time n^c (and in particular decodable from p relative errors in time n^c).*

In the remainder of this section we prove Theorems 4.3, 4.4 and 4.5.

4.4 Extending the control codes of [KSS19] to have repetition decoding

Kopparty, Shaltiel and Silbak [KSS19] used a binary code which they call the "Raw Reed-Solomon code" to construct control codes with the following properties:

Theorem 4.6 (Stochastic control codes of [KSS19])**.** *For every constant $0 < \alpha < \frac{1}{2}$, and for every sufficiently large m, setting $n = (2^m - 1) \cdot m$, $k = n^\alpha$, $d = n \log n$, and $s = \frac{n^\alpha}{\log^3 n}$, there is a stochastic code* $\mathrm{Enc} : \{0,1\}^k \times \{0,1\}^d \to \{0,1\}^n$ *that is:*

- *Computable in time n^c for a constant c that does not depend on α.*
- *2^{-s}-pseudorandom for space s ROBPs.*
- *$\delta(\mathrm{Enc}) \geq \frac{1}{2} - o(1)$.*
- *For every constant $p < 1/4$,* Enc *is decodable from p relative errors, with an $O(\frac{1}{\log n})$-relative approximation in time n^c.*
- *There exists a universal constant b such that for every $\beta \geq b\sqrt{\alpha}$,* Enc *is $O(1/\beta^2)$-list decodable from $(\frac{1}{2} - \beta)$ relative errors, with an $O(\frac{1}{\log n})$-relative approximation, and this list-decoding algorithm runs in time n^c.*
- *If $\alpha \leq 0.1$ then encoding n^c inputs takes "amortized time" $O(n \cdot \log^2 n)$, namely, for every n^c pairs, $(m_1, s_1), \ldots, (m_{n^c}, s_{n^c}) \in \{0,1\}^k \times \{0,1\}^d$, computing $(\mathrm{Enc}(m_i, S_i))_{(i \in [n^c])}$ takes time $n^c \cdot O(n \cdot \log^2 n)$.*

Theorem 4.6 is stated in [KSS19] without the statement on "distance" and "relative approximation" which were not considered in [KSS19]. Nevertheless, the proof of [KSS19] gives that the code has large distance, and that whenever decoding (or list-decoding), one obtains a $O(\frac{1}{\log n})$ relative approximation.[17]

Theorems 4.3, 4.4 and 4.5 guarantee repetition decoding (which does not appear in Theorem 4.6). In order to prove them, we prove the following lemma which shows that repetition decoding follows if the code has large distance and list-decoding with a good approximation.

[17] Loosely speaking the control code $\mathrm{Enc}(s, r)$ of [KSS19] works by treating $r = (r_1, r_2)$ as two independent seeds, and $\mathrm{Enc}(s, r) = \mathrm{Enc}^{RawRS}(r_1 \circ s) \oplus f(r_2)$ where Enc^{RawRS} is an encoding map for a (standard) error correcting code that is decodable from $1/4 - o(1)$ relative errors (which implies that it has relative distance $1/2 - o(1)$), and $f(r_2)$ is a string of relative hamming weight at most $\eta = \frac{1}{\log n}$. This implies that Enc has $\delta(\mathrm{Enc}) = 1/2 - o(1)$, and that when decoding (or list-decoding) a received word $v \in \{0,1\}^n$, one can retrieve s, r_1 such that for every r_2, $\mathrm{Enc}(s, r_1 \circ r_2)$ is within relative Hamming distance η from v. This gives an η-relative approximation to the distance $\delta^{\mathrm{Enc}}(s, v)$.

Lemma 4.7 (repetition decoding from distance and list-decoding with a good approximation)**.** *Let $p > 0$ and let* $\mathrm{Enc} : \{0,1\}^k \times \{0,1\}^d \to \{0,1\}^n$ *be a stochastic code such that:*

- Enc *is L-list decodable from $\rho = 2p + 3\eta$ relative errors, with an η-relative approximation.*
- Enc *has relative distance $\delta(\mathrm{Enc}) \geq \rho$.*

It follows that Enc *is repetition decodable from p relative errors. Furthermore, the amortized time of the repetition decoding algorithm* rDec *is bounded $O(T_{\mathrm{Dec}})$ where T_{Dec} is the running time of the list-decoding algorithm.*

The proof of Lemma 4.7 is given in Section 4.5. Theorem 4.3 immediately follows from Theorem 4.6 and Lemma 4.7. Theorem 4.5 follows by using the decoding algorithm in Theorem 4.6 as a 1-list-decoding algorithm from slightly less than $1/4$ relative errors, and applying Lemma 4.7. The proof of Theorem 4.4 (which is not used in this paper) is deferred to a later version (see Section 5.6 for a discussion of potential uses of this theorem).

4.5 Repetition decoding from distance and list-decoding: proof of Lemma 4.7

In this section we prove Lemma 4.7. We consider the following repetition decoding algorithm:

Algorithm rDec**:** On input ℓ and $v_1, \ldots, v_\ell \in \{0,1\}^n$, rDec does the following:

- For $i \in [\ell]$ apply $\mathrm{Dec}(v_i)$ to obtain a list of L pairs $(s_1^i, \omega_1^i), \ldots, (s_L^i, \omega_L^i)$ of a message and relative approximation.
- Overall, we obtained at most $\ell \cdot L$ such pairs. Let A be the set of $s \in \{0,1\}^k$ such that for at least half of $i \in [\ell]$, there exists a $j \in [L]$ such that $s_j^i = s$. Note that $|A| \leq 2 \cdot L$.
- For each $s \in A$, and for each $i \in [\ell]$ compute a number $p_{s,i}$ as follows: If s appears in the i'th list with an approximation ω', then $p_{s,i} = \omega'$. Otherwise, we set $p_{s,i} = \rho$.
- We output the s that has the minimal $p_s = \sum_{i \in [\ell]} p_{s,i}$ out of all strings that were considered. (If there does not exist such a unique s the algorithm fails).

In order to analyze the algorithm we need the following notion of truncated distance.

Definition 4.8 (truncated distance)**.** *For a function $\delta(\cdot,\cdot)$ and a number $\rho \geq 0$ we define $\delta_\rho(x,y) = min(\delta(x,y), \rho)$.*

With this notation, it immediately follows that:

Claim 4.9. *For every $s \in A$, $|p_s - \sum_{i \in [\ell]} \delta_\rho^{\mathrm{Enc}}(s, v_i)| \leq \eta$.*

It is standard that if δ is a function that satisfies the triangle inequality, then δ_ρ also satisfies the triangle inequality for every $\rho > 0$. As a corollary we get that:

Claim 4.10. *If $\delta(\mathrm{Enc}) \geq \rho$, then for every $s, s' \in \{0,1\}^k$, and $v_1, \ldots, v_\ell \in \{0,1\}^n$,*

$$\frac{1}{\ell} \cdot \sum_{i \in [\ell]} \delta_\rho^{\mathrm{Enc}}(s, v_i) + \frac{1}{\ell} \cdot \sum_{i \in [\ell]} \delta_\rho^{\mathrm{Enc}}(v_i, s') \geq \rho.$$

Proof. (of Claim 4.10) We have that $\delta^{\mathrm{Enc}}(s, s') \geq \rho$, which means that $\delta_\rho^{\mathrm{Enc}}(s, s') = \rho$. This means that for every $v \in \{0,1\}^n$,

$$\delta_\rho^{\mathrm{Enc}}(s, v) + \delta_\rho^{\mathrm{Enc}}(v, s') \geq \delta_\rho^{\mathrm{Enc}}(s, s') = \rho.$$

The claim follows by applying the inrquality above seperately for each i, and taking the average. □

We need to show that under the conditions of Lemma 4.7, the algorithm rDec is a correct repetition decoding algorithm. This is done in the next claim.

Claim 4.11. *For every $s \in \{0,1\}^k$, if* rDec *receives as input a number ℓ and $v_1, \ldots, v_\ell \in \{0,1\}^n$ such that $\frac{1}{\ell} \cdot \sum_{i \in [\ell]} \delta^{\mathrm{Enc}}(s, v_i) \leq p$, then:*

- *$s \in A$, and $p_s \leq p + \eta$.*
- *For every $s' \neq s$, such that $s' \in A$, $p_{s'} > p + \eta$.*

Together, this implies that $\mathrm{rDec}(\ell, v_1, \ldots, v_\ell) = s$ *as required.*

Proof. (of Claim 4.11). By Markov's inequality, for at least half of $i \in [\ell]$ we have that $\delta^{\mathrm{Enc}}(s, v_i) \leq 2p \leq \rho$. For such an i, s is one of the L strings obtained when applying $\mathrm{Dec}(v_i)$. This means that $s \in A$. The first item now follows using Claim 4.9:

$$\begin{aligned} p_s &\leq \sum_{i \in [\ell]} \delta_\rho^{\mathrm{Enc}}(s, v_i) + \eta \\ &\leq \sum_{i \in [\ell]} \delta^{\mathrm{Enc}}(s, v_i) + \eta \\ &\leq p + \eta. \end{aligned}$$

For the second item, we apply Claim 4.9 and Claim 4.10 to conclude that:

$$\begin{aligned} p_{s'} &\geq \frac{1}{\ell} \cdot \sum_{i \in [\ell]} \delta_\rho^{\mathrm{Enc}}(s', v_i) - \eta \\ &\geq \rho - \frac{1}{\ell} \cdot \sum_{i \in [\ell]} \delta_\rho^{\mathrm{Enc}}(s, v_i) - \eta \\ &\geq (2p + 3\eta) - p - \eta \\ &= p + 2\eta. \end{aligned}$$

□

We now consider the amortized running time of the repetition decoding algorithm rDec. We are assuming that $\ell \leq 2^n$ so that $\log \ell \leq n$. The running time T_{Dec} of the list-decoding algorithm is at least $n + L$ as it needs to read an input of length n and output a list of size L. The algorithm rDec calls Dec ℓ times (taking amortized time T_{Dec}). For each of the $2L = O(T_{\mathrm{Dec}})$ elements in A, the algorithm rDec sums up ℓ numbers. It follows that this step also takes amortized time $O(T_{\mathrm{Dec}})$. Overall, the amortized running time of the repetition decoding algorithm is $O(T_{\mathrm{Dec}})$ as required.

4.6 Stochastic control code in near linear time

In this section we provide a sketch of the proof of Theorem 4.4. The starting point is once again the code of Theorem 4.6. Let Enc be the code from this theorem (choosing the parameter β in Theorem 4.4 to be say 10β where this occurrence of β is the constant that is chosen in Theorem 4.4). Let c be the constant guaranteed in Theorem 4.4.

Let $u = n^c$. The code that we will construct will be denoted by $\mathrm{Enc}' : \{0,1\}^k \times \{0,1\}^{ud} \to \{0,1\}^N$ for $N = un$, and will be defined as follows:

$$\mathrm{Enc}'(s, (r_1, \ldots, r_u)) = (\mathrm{Enc}(s, r_1), \ldots, \mathrm{Enc}(s, r_u)).$$

First of all, we note that $k = n^\alpha = N^{\alpha'}$ for a constant $\alpha' = \alpha/(c+1)$.

By the last item of Theorem 4.6 Enc' can be computed in time $O(u \cdot n \cdot \log^2 n) = O(N \cdot \log^2 N)$, giving that Enc' can be computed in the required time.

By a standard hybrid argument, we have that Enc' is $(u \cdot 2^{-s})$-pseudorandom for space s ROBPs. Note that $s = n^\alpha / \log^3 n \geq N^{\alpha'} / \log^3 N$. This also means that $u2^{-s} \leq 2^{-s/2}$, and so by choosing $\alpha' > 0$ to be slightly smaller, we can use it as α in Theorem 4.4, and meet the requirement.

So far, we have seen that Enc' has sufficiently efficient encoding, and is sufficiently pseudorandom. We would like to show that Enc' has sufficiently efficient repetition decoding. This would follow from the argument in Lemma 4.7 if we could show that Enc' has sufficiently large distance and is also list-decodable with good parameters.

The distance property of Enc' follows immediately. We have already seen that bundling many copes of a code preserves relative distance, and so $\delta(\mathrm{Enc}') = \delta(\mathrm{Enc}) \geq \frac{1}{2} - o(1)$.

While for technical reasons, we can't use Lemma 4.7 directly, it is possible to imitate the algorithm at the basis of Lemma 4.7 using the list-decoding algorithm of Theorem 4.6 and randomized subsampling that saves time, and allows us to read a sample of the blocks of the codeword of Enc' rather than reading all of them. The exact details are deferred to a later version.

5 Explicit stochastic codes for space bounded channels

In this section we give our main construction of codes for space bounded channels, and prove Theorem 1.1. We start by restating the theorem in a more general way:

Theorem 5.1 (Explicit stochastic codes for space bounded channels)**.** *For every constants $p < \frac{1}{4}$, and $c_\nu > 1$, there exists a constant $\delta > 0$ such that for every sufficiently small constant $\epsilon > 0$, for infinitely many N, there is a stochastic code for $\mathrm{Spc}_p^{N^\delta}$ with rate $R = 1 - H(p) - \epsilon$, and success probability $1 - \nu$ for $\nu = 2^{-(\log N)^{c_\nu}}$. Furthermore, the encoding and decoding algorithms run in time polynomial in N.*

More specifically, there exists a universal polynomial $T_0(N)$ such that encoding and decoding algorithms run in time $T_0(N)$ for every sufficiently large N (where the choice of which N is sufficiently large depends on p and ϵ).

5.1 The construction

In this section we present our construction of stochastic codes for bounded channels. The construction is detailed in three figures: Figure 1 lists parameters and ingredients, Figure 2 describes the encoding algorithm, and Figure 3 describes the decoding algorithm. We start with some notation and definitions. We remark that an intuitive explanation of the construction appears in Section 1.3.

Partitioning codewords into control blocks and data blocks. The construction will think of codewords $c \in \{0,1\}^N$ as being composed of $n = n_{\mathrm{ctrl}} + n_{\mathrm{data}}$ blocks of length $b = N/n$. Given a subset $I \subseteq [n]$ of n_{ctrl} distinct indices, we can decompose c into its data part $c_{\mathrm{data}} \in \{0,1\}^{N_{\mathrm{data}} = n_{\mathrm{data}} \cdot b}$ and its control part $c_{\mathrm{ctrl}} \in \{0,1\}^{N_{\mathrm{ctrl}} = n_{\mathrm{ctrl}} \cdot b}$. Similarly, given strings c_{data} and c_{ctrl} we can prepare the codeword c (which we denote by $(c_{\mathrm{data}}, c_{\mathrm{ctrl}})^I$ by the reverse operation. This is stated formally in the definition below.

Definition 5.2 (Data and control portion of a codeword)**.** *We view strings $c \in \{0,1\}^N$ as composed of n blocks of length $b = N/n$, so that $c \in (\{0,1\}^b)^n$, and c_i denotes the b bit long i'th block of c.*

Let $I = \{i_1, \ldots, i_{n_{\mathrm{ctrl}}}\} \subseteq [n]$ be a subset of indices of size n_{ctrl}.

- *Given strings $c_{\mathrm{data}} \in \{0,1\}^{N_{\mathrm{data}}}$ and $c_{\mathrm{ctrl}} \in \{0,1\}^{N_{\mathrm{ctrl}}}$ we define an N bit string c denoted by $(c_{\mathrm{data}}, c_{\mathrm{ctrl}})^I$ as follows: We think of $c_{\mathrm{data}}, c_{\mathrm{ctrl}}, c$ as being composed of blocks of length b (that is*

Figure 1: Parameters and ingredients for stochastic code

Constants:

- $p > 0$ - The fraction of errors we need to recover from.
- $\epsilon > 0$ - The final code will have rate $R \geq 1 - H(p) - \epsilon$. We assume that $\epsilon > 0$ is sufficiently small in terms of p.
- $\delta > 0$ - we are aiming to construct codes for space $s := N^{\delta}$ channels.
- $c_\nu \geq 1$ - We set $\nu := 2^{-(\log N)^{c_\nu}}$.

Parameters that are allowed to vary with N:

- N - The length (in bits) of the codeword. Throughout, we assume that N is sufficiently large, and that other parameters are chosen as a function of N. Later choices will also restrict N to be a number of a special form. We divide the N output bits to $n := (\log N)^{c_\nu+10}$ blocks of length $b = N/n$.
- ℓ - This is the total length of a "control seed". Let $\ell' = \ell/3$. This will be the length of individual "seeds".
- s' - Pseudorandom components will be $2^{-s'}$-pseudorandom for space s' ROBPS.

Stochastic control code: The construction receives a stochastic code $\mathrm{Enc}_{\mathrm{ctrl}} : \{0,1\}^{\ell} \times \{0,1\}^{d} \to \{0,1\}^{b}$ such that:

- $\mathrm{Enc}_{\mathrm{ctrl}}$ is $(2^{-s'})$-pseudorandom for space s' ROBPs.
- $\mathrm{Enc}_{\mathrm{ctrl}}$ is repetition decodable from $p_{\mathrm{ctrl}} := p + \epsilon$ relative errors with decoding algorithm $\mathrm{rDec}_{\mathrm{ctrl}}$.

Requirements: $s' \leq \frac{\ell}{(\log N)^{c_\nu+15}}$, and there exists a constant $\xi > 0$ such that $s' \geq s \cdot N^{\xi} = N^{\delta+\xi}$.

Some more parameters:

- Let $\epsilon' > 0$ be a sufficiently small constant that will be chosen later as a function of p, ϵ.
- Let $n_{\mathrm{ctrl}} = \epsilon' \cdot n$, $n_{\mathrm{data}} = n - n_{\mathrm{ctrl}}$.
- Let $N_{\mathrm{ctrl}} = b \cdot n_{\mathrm{ctrl}}$ and $N_{\mathrm{data}} = b \cdot n_{\mathrm{data}}$. (Note that: $n = n_{\mathrm{ctrl}} + n_{\mathrm{data}}$, $N = N_{\mathrm{ctrl}} + N_{\mathrm{data}}$).
- Later on, in the decoding and analysis we introduce two parameters $\tau = \frac{n_{\mathrm{ctrl}} \cdot \epsilon_{\mathrm{samp}}}{2}$ and $\tau' = (\log N)^{c_\nu+1}$.

Other ingredients that are used:

BSC code: Let $\alpha = \frac{\epsilon}{10 \log \frac{1}{p}}$, $p_{\mathrm{BSC}} = p + \alpha$, and $R_{\mathrm{BSC}} = 1 - H(p_{\mathrm{BSC}}) - \epsilon/3$. We apply Theorem 3.10 using $p_{\mathrm{BSC}}, \epsilon/3, N_{\mathrm{data}}$ as choices for p, ϵ, n, respectively. Theorem 3.10 only guarantees the code for infinitely many block lengths, and so we require that $N_{\mathrm{data}} = (1 - \epsilon') \cdot N$ is one of these block lengths. This translates into a restriction on N (which is satisfied for infinitely many N). We obtain an encoding function $\mathrm{Enc}_{\mathrm{BSC}} : \{0,1\}^{R_{\mathrm{BSC}} \cdot N_{\mathrm{data}}} \to \{0,1\}^{N_{\mathrm{data}}}$.

t-wise independent permutation: Let $t = (\log N)^{c_\nu+2}$. We use a $(2^{-10 \cdot t}, t)$-wise permutation $\pi : \{0,1\}^{\ell'} \times [N_{\mathrm{data}}] \to [N_{\mathrm{data}}]$. By Theorem 2.9 we have an explicit construction with seed length $O(t \cdot \log N) \leq \ell'$.

Averaging Sampler: Let $\epsilon_{\mathrm{samp}} > 0$ be a sufficiently small constant to be chosen later. We use Theorem 2.7 to obtain an $(\epsilon_{\mathrm{samp}}, \frac{\nu}{N^3})$-sampler with distinct samples $\mathrm{Samp} : \{0,1\}^{\ell'} \to [n]^{n_{\mathrm{ctrl}}}$. By Theorem 2.7 we have an explicit construction with seed length $O(\log \frac{N^3}{\nu}) = O((\log N)^{c_\nu+1}) \leq \ell'$. We use n_{ctrl} samples, and indeed $n_{\mathrm{ctrl}} = \epsilon' \cdot n \geq \epsilon' \cdot (\log N)^{c_\nu+10} \gg \log \frac{N^3}{\nu}$ (as required in Theorem 2.7).

PRG for space s' ROBPs: Let $G_{\mathrm{robp}} : \{0,1\}^{O(s' \cdot (\log N_{\mathrm{data}})^2)} \to \{0,1\}^{N_{\mathrm{data}}/n}$ be the $(2^{-10s'})$-PRG for space s' ROBPs that is provided by Theorem 2.5. We define $G(x_1, \ldots, x_{n_{\mathrm{data}}}) = G_{\mathrm{robp}}(x_1), \ldots, G_{\mathrm{robp}}(x_{n_{\mathrm{data}}})$, and think of G as a function $G : \{0,1\}^{\ell'} \to \{0,1\}^{N_{\mathrm{data}}}$, and for this we verify that indeed $\ell' > n \cdot (s'(\log N)^3)$ for sufficiently large N. By a hybrid argument, we have that G is $2^{-2s'}$-pseudorandom for space s' ROBPs.

$c_{\mathrm{data}} \in (\{0,1\}^b)^{n_{\mathrm{data}}}$, $c_{\mathrm{ctrl}} \in (\{0,1\}^b)^{n_{\mathrm{ctrl}}}$ *and* $c \in (\{0,1\}^b)^n$*). We enumerate the indices in* $[n] \setminus I$ *by* $j_1, \ldots, j_{n_{\mathrm{data}}}$ *and set* $c_\ell = \begin{cases} (c_{\mathrm{ctrl}})_k & \text{if } \ell = i_k \text{ for some } k; \\ (c_{\mathrm{data}})_k & \text{if } \ell = j_k \text{ for some } k \end{cases}$

- *Given a string* $c \in \{0,1\}^N$ *(which we think of as* $c \in (\{0,1\}^b)^n$*) we define strings* $c^I_{\mathrm{data}}, c^I_{\mathrm{ctrl}}$ *by* $c^I_{\mathrm{ctrl}} = c|_I$ *and* $c^I_{\mathrm{data}} = c|_{[n]\setminus I}$*, (namely the strings restricted to the indices in* I*,* $[n] \setminus I$*, respectively).*

We omit the superscript I *when it is clear from the context.*

Figure 2: Encoding algorithm for stochastic code

Input:

- A message $m \in \{0,1\}^{R_{\mathrm{BSC}} \cdot N_{\mathrm{data}}}$. (This gives $R = \frac{R_{\mathrm{BSC}} \cdot N_{\mathrm{data}}}{N}$).
- A "random coin" for the stochastic encoding that consists of: a string $s = (s_{\mathrm{samp}}, s_\pi, s_{\mathrm{PRG}})$ where $s_{\mathrm{samp}}, s_\pi, s_{\mathrm{PRG}} \in \{0,1\}^{\ell'}$ so that $s \in \{0,1\}^\ell$, and $r_1, \ldots, r_{n_{\mathrm{ctrl}}} \in \{0,1\}^d$.

Output: A codeword $c = \mathrm{Enc}(m; (s, r_1, \ldots, r_{n_{\mathrm{ctrl}}}))$ of length N.

Operation:

Determine control blocks: Apply $\mathrm{Samp}(s_{\mathrm{samp}})$ to generate *control* $= \{i_1, \ldots, i_{n_{\mathrm{ctrl}}}\} \subseteq [n]$. These blocks will be called "control blocks", and the remaining n_{data} blocks will be called "data blocks".

Prepare data part: We prepare a string c_{data} of length N_{data} as follows:

- Encode m by $x = \mathrm{Enc}_{\mathrm{BSC}}(m)$.
- Generate an N_{data} bit string y by reordering the N_{data} bits of the encoding using the (inverse of) the permutation $\pi_{s_\pi}(\cdot) = \pi(s_\pi, \cdot)$. More precisely, $y = \pi^{-1}_{s_\pi}(x) = \pi^{-1}_{s_\pi}(\mathrm{Enc}_{\mathrm{BSC}}(m))$.
- Mask y using PRG. That is, $c_{\mathrm{data}} = y \oplus G(s_{\mathrm{PRG}}) = \pi^{-1}_{s_\pi}(\mathrm{Enc}_{\mathrm{BSC}}(m)) \oplus G(s_{\mathrm{PRG}})$.

Prepare control part: We prepare a string c_{ctrl} of length N_{ctrl} (which we view as n_{ctrl} blocks of length b) as follows:

- $(c_{\mathrm{ctrl}})_j = \mathrm{Enc}_{\mathrm{ctrl}}(s, r_j)$.

Merge data and control parts: We prepare the final output codeword $c \in \{0,1\}^N$ by merging c_{data} and c_{ctrl}. That is, $c = (c_{\mathrm{data}}, c_{\mathrm{ctrl}})^{\mathit{control}}$.

5.2 Stating the correctness of the construction

In this section we state a general theorem stating the correctness of the construction, assuming that it is supplied with the right ingredients.

Theorem 5.3 (correctness of the construction)**.** *For every constants* $\delta > 0$*,* $0 \le p < \frac{1}{4}$*,* $c_\nu \ge 1$*, and every sufficiently small constant* $\epsilon > 0$*, there exist constants* $\epsilon' > 0$*,* $\epsilon_{\mathrm{samp}} > 0$ *such that for infinitely many* N*, we have that for every choice of* ℓ, s'*, and every choice of* $\mathrm{Enc}_{\mathrm{ctrl}} : \{0,1\}^\ell \times \{0,1\}^d \to \{0,1\}^b$ *that satisfiy the requirements in Figure 1, the following holds: The encoding and decoding functions* $\mathrm{Enc} : \{0,1\}^{Rn} \times \{0,1\}^{\ell + n_{\mathrm{ctrl}} \cdot d} \to \{0,1\}^N$ *and* $\mathrm{Dec} : \{0,1\}^N \to \{0,1\}^{RN}$ *specified in Figures 2 and 3 using the ingredients and parameter choices in Figure 1 satisfy the following properties:*

- Enc *has rate* $R \ge 1 - H(p) - \epsilon$*.*
- Dec *is a decoding algorithm showing that* Enc *is decodable for space* $s = N^\delta$ *channels that induce at most* pN *errors, with probability* $1 - \nu$ *for* $\nu = 2^{-(\log N)^{c_\nu}}$*.*

Figure 3: Decoding algorithm for stochastic code

Input: A "received word" $\bar{v} \in \{0,1\}^N$.

Output: A messages $\bar{m} \in \{0,1\}^{RN}$.

Internal procedure *DecodeUsingCandidate*: On input $\bar{s} \in \{0,1\}^\ell$ (which we think of as a candidate for the control string) this procedure produces a message $\bar{m}(\bar{s})$ as follows:

Determine control blocks: Apply $\mathrm{Samp}(\bar{s}_{\mathrm{samp}})$ to generate $\overline{control} = \{\bar{i}_1, \ldots, \bar{i}_{n_{\mathrm{ctrl}}}\}$. Compute $\bar{v}_{\mathrm{data}} = \bar{v}_{\mathrm{data}}^{\overline{control}}$.

Unmask PRG: Compute $\bar{y} = \bar{v}_{\mathrm{data}} \oplus G(\bar{s}_{\mathrm{PRG}})$.

Reverse permutation: Let $\bar{x}$ be the N_{data} bit string obtained by "undoing" the permutation. More precisely, let $\pi_{\bar{s}_\pi}(\cdot) = \pi(\bar{s}_\pi, \cdot)$, and let $\bar{x} = \pi_{\bar{s}_\pi}(\bar{y}) = \pi_{\bar{s}_\pi}(\bar{v}_{\mathrm{data}} \oplus G(\bar{s}_{\mathrm{PRG}}))$.

Decode data: Compute $\bar{m} = \mathrm{Dec}_{\mathrm{BSC}}(\bar{x})$.

output: We use $\bar{y}(\bar{s})$, $\bar{x}(\bar{s})$ and $\bar{m}(\bar{s})$ to denote the variables $\bar{y}, \bar{x}, \bar{m}$ when *DecodeUsingCandidate* is applied on $\bar{s}$.

Operation of decoding algorithm: On input $\bar{v} \in \{0,1\}^N$:

Compute control candidates: For $i \in [n]$, let $\bar{s}_i = \mathrm{Dec}_{\mathrm{ctrl}}(\bar{v}_i)$. (Here $\bar{v}_i$ is the i'th block of $\bar{v}$ and $\mathrm{Dec}_{\mathrm{ctrl}}(\bar{v}_i) = \mathrm{rDec}_{\mathrm{ctrl}}(1, \bar{v}_i)$, recall that decoding is a special case of repetition decoding).
Let *candidates* $= \{\bar{s}_i : i \in [n]\}$.

Compute viable candidates: We say that $\bar{s} \in \{0,1\}^\ell$ is ***viable*** if for at least $\tau := \frac{n_{\mathrm{ctrl}} \cdot \epsilon_{\mathrm{samp}}}{2}$ choices of $i \in [n]$, $\bar{s} = \bar{s}_i$. Let *viable* be the set of all $\bar{s} \in$ *candidates* such that $\bar{s}$ is viable. Note that *viable* is of size at most n_{ctrl}/τ which is a constant (that depends on ϵ_{samp}).

Compute valid candidates: We say that $\bar{s}$ is ***successful***, if when computing the procedure *DecodeUsingCandidate*$(\bar{s})$, we obtain $\bar{x}(\bar{s})$ such that $\mathrm{outdist}(\bar{x}) \leq \lambda_1/4$. (Here λ_1 is the constant from Theorem 3.10). Let *valid* be the set of $\bar{s} \in$ *viable* which are successful, such a string $\bar{s}$ is called ***valid***.

Compute Active blocks: A block $i \in [n]$ is ***active*** if s_i is valid. Let a be the number of active blocks, and denote their indices by $j_1, \ldots, j_a \in [n]$.

Perform repetition decoding: Compute $s^* = \mathrm{rDec}_{\mathrm{ctrl}}(a; \bar{v}_{j_1}, \ldots, \bar{v}_{j_a})$.

Output message: Compute *DecodeUsingCandidate*(s^*) and output $\bar{m} = \bar{m}(s^*)$.

- *If* $\mathrm{Enc}_{\mathrm{ctrl}}, \mathrm{Dec}_{\mathrm{ctrl}}$ *can be computed in polynomial time, then* $\mathrm{Enc}, \mathrm{Dec}$ *can be computed in polynomial time.*

  More specifically, there exists a universal constant $c_0 > 1$ *such that the time of* $\mathrm{Enc}, \mathrm{Dec}$ *is bounded by* $O(N \cdot (\log N)^{c_0 \cdot c_\nu} + n \cdot T_{\mathrm{ctrl}} + T_{\mathrm{data}})$ *where* T_{data} *is a bound on the running time on the encoding and decoding time of* $\mathrm{Enc}_{\mathrm{BSC}}$ *and* T_{ctrl} *is a bound on the running time of running* $\mathrm{Enc}_{\mathrm{ctrl}}$ *and on the amortized running time of the repetition decoding algorithm* $\mathrm{rDec}_{\mathrm{ctrl}}$*, where both* $\mathrm{Enc}, \mathrm{rDec}$ *are applied with block length* $b = N/n$*. (Here, the constant hidden in the* $O(\cdot)$ *depends on the* p, ϵ, δ*).*

We prove Theorem 5.3 in Section 6. In the next section, we plug in specific ingredients to prove our main theorems.

5.3 High level intuition and comparison to [KSS19]

Our construction heavily builds on the list-decodable code of Kopparty, Shaltiel and Silbak [KSS19] (which in turn heavily builds on the construction of Guruswami and Smith [GS16] and the modifications made by

Shaltiel and Silbak [SS16]). The high level intuition is that we repeat the list-decoding algorithm of [KSS19], to end up with a constant size list of candidate messages, and unique decoding is achieved by pruning the list using additional steps.

Indeed, using our terminology (modulu some small changes to explain later on) the construction of [KSS19] achieves list decoding by taking all viable candidates $\bar{s} \in \textit{viable}$ (defined in the second step of the operation of the decoding algorithm) and outputting the list of messages

$$\{\bar{m}(\bar{s}) : \bar{s} \in \textit{viable}\} .$$

The proof of [KSS19] shows that the original message is in this list w.h.p.

Somewhat oversimplifying, our high level plan is to show that:

- *Data blocks* (namely blocks in $[n] \setminus \textit{control}$) are unlikely to be successful, and therefore, are unlikely to be declared active. In order to achieve this we will argue that on data candidates, the evasiveness of the code $\mathrm{Enc}_{\mathrm{BSC}}$ will make these candidates unsuccessful. This requires many additional ideas that are explained later on.
- We will also show that at least τ of the *control blocks* (namely blocks in *control*) will choose the correct candidate s as their candidate $\bar{s}$. This support of τ blocks will make this candidate viable. Furthermore, we will show that s is successful, and so these τ blocks will be declared active.
- We will show that any other control block is either not active, or within relative distance roughly p to $\mathrm{Enc}_{\mathrm{ctrl}}(s, U_d)$.
- Overall, we will get that if we apply repetition decoding on active blocks, we obtain the correct candidate s. This will allow us to identify the correct candidate s in the set *viable*, and we will output $\bar{m}(\bar{s})$ which is the correct message (w.h.p.).
- The actual argument is more complicated as we cannot rule out that few data blocks will be successful and declared active.

We now point out a few additional differences in the choice of parameters and ingredients relative to [KSS19] that are crucial to our approach, and will allow us to implement the plan above.

Evasiveness of the data encoding: The code from Theorem 3.10 meets the requirements of Theorem 3.3 and is therefore evasive. This means that when decoding is applied on uniform strings, it is unlikely to be successful. A significant portion of the analysis below is used to argue that each data block is sufficiently pseudorandom so that we can argue that w.h.p. data blocks are not successful.

Unique decoding of control blocks: In [KSS19] the decoding algorithm of the control code is *list-decoding* from slightly less than $\frac{1}{2}$ relative errors, and so, every block has several candidates. We are interested in $p < \frac{1}{4}$ and therefore can replace this by unique decoding (which is used to specify the candidate $\bar{s}$ of each block). Unique decoding is a special case of repetition decoding.

Repetition decoding of control blocks: We will be able to guarantee that the correct candidate s is valid. However, it is no hard to see that the channel can inject a related candidate $\bar{s}$ (that depends on s) so that $\bar{s}$ is also valid. Moreover, the channel can inject several such strings. How can we identify the correct candidate amongst all valid candidates? This is where repetition decoding comes in. We argue that although the incorrect candidate $\bar{s}$ is valid, and was decoded on at least τ of the control blocks, the correct candidate is the only valid candidate that is within small average distance from all the active blocks. This is key to achieving unique decoding for $p \geq \frac{1}{8}$. The high level intuition for the correctness of repetition decoding is that the channel has a budget of roughly p to spend on control blocks. Initially, the control portion of the codeword consists of control encodings of the correct string s. The channel

can place $1/4$ fraction of errors on almost all blocks, leading them to decode to some other related string $\bar{s}$. However, in doing so, on all these blocks, the received word will be pretty far from a control encoding of $\bar{s}$, and so, when we repetition decode, we will still get s (even though on most internal specific decdoings we get $\bar{s}$).

PRG with truly independent blocks: We want different blocks of $G(U_{\ell'})$ to be truly independent (and not just indistinguishable from truly independent). This will allow us to argue that even though $\mathrm{Dec}_{\mathrm{ctrl}}$ cannot be implemented by a small space ROBP, it's behavior on different blocks is independent. This will be crucial in showing that candidates of data blocks are not successful. The need to keep the seed short dictates using fewer longer blocks.

Shorter control piece: The method devised in [GS16] and used in [KSS19] is to have the encoding consist of a long data piece of length $N_{\mathrm{data}} = (1-\epsilon')N$, and a short control piece of length $\epsilon' N$. In these works the final list size is roughly polynomial in the fraction ϵ', and so it is natural to choose $\epsilon' = \Omega(\epsilon)$ to minimize the list size. For our pruning technique, it is beneficial if ϵ' is much smaller. Loosely speaking, this allows us to account for a deletion of an ϵ'-fraction of the data, as an additional "few errors" that can be later decoded. This leads us to choose ϵ' to be much smaller than in [KSS19].

5.4 Deriving Theorem 1.1

We now show that Theorem 5.1 (which generalizes Theorem 1.1) follows by picking specific components in Theorem 5.3. We are given constant $p < \frac{1}{4}$, $c_\nu > 1$ and a sufficiently small constant $\epsilon > 0$. Throughout we will assume that $p + \epsilon < \frac{1}{4}$.

We want to choose parameters ℓ, s', as well as a control code to plug into Theorem 5.3. We start with choosing the control code. Our plan is to use Theorem 4.3 as a control code. Theorem 4.3 allows us to choose $\beta > 0$, and we choose β to be sufficiently small so that $p + \epsilon \leq \frac{1}{4} - \beta$. By theorem 4.3 we obtain a constant $\alpha > 0$ so that we can obtain a control code $\mathrm{Enc}_{\mathrm{ctrl}} : \{0,1\}^{b^\alpha} \times \{0,1\}^{b \log b} \to \{0,1\}^b$. Recall that we need to set $b = N/n$. We are allowed to choose b of the form $(2^m - 1) \cdot m$ and such numbers are sufficiently dense so that by slightly changing n by a constant factor (which makes no difference in the construction and analysis) we can make sure that $b = N/n$ is of this form. The code we obtain has $\mathrm{poly}(b)$ time encoding and repetition decoding. This code is repetition decodable from $p_{\mathrm{ctrl}} = p + \epsilon$ relative errors. We choose $\ell = b^\alpha$ and $s' = \frac{\ell}{(\log N)^{c_\nu + 15}}$. It follows that $\mathrm{Enc}_{\mathrm{ctrl}}$ is $2^{-s'}$-pseudorandom for space s' ROBPs. This means that $\mathrm{Enc}_{\mathrm{ctrl}} : \{0,1\}^\ell \times \{0,1\}^d \to \{0,1\}^b$ satisfies the requirements from a control code in Figure 1.

We have that

$$s' = b^\alpha \log^3 N \geq \frac{N^\alpha}{n \cdot \log^3 N} \geq N^{\alpha/2}.$$

We choose $\delta = \xi = N^{\alpha/4}$ and set $s = N^\delta$. We indeed have that $s' \geq N^{\delta+\xi}$ as required in Figure 1. This means that the requirements in Figure 1 are met by our choices of ℓ and s'. It follows that we meet all the conditions of Theorem 5.3 and obtain that:

- Enc has rate $R \geq 1 - H(p) - \epsilon$.
- There is a decoding algorithm Dec showing that Enc is list decodable for space s channels that induce at most pN errors, with probability $1 - \nu$.
- The running time of Enc and Dec is some fixed polynomial T_0.

This completes the proof of Theorem 5.1

5.5 Deriving Theorem 1.2

In order to prove Theorem 1.2 we will use a nonexplicit stochastic code. More specifically, using the probabilistic method, it is easy to show that if we take ℓ and d to be sufficiently large as a function of b, but set $\ell, d = o(b)$, (for example, say $\ell, d = \Theta(b/\log\log b)$) and if furthermore, we choose $s' = \Omega(\ell)$, then a uniformly chosen stochastic code (that is a code where for every $s \in \{0,1\}^\ell$ and $r \in \{0,1\}^d$, the output $\mathrm{Enc}_{\mathrm{ctrl}}(s,r)$ is chosen uniformly and independently in $\{0,1\}^b$) is likely to have the following properties:

- $\mathrm{Enc}_{\mathrm{ctrl}}$ is $2^{-\Omega(\ell)}$-pseudorandom for space $\Omega(\ell)$ ROBPs.

  This follows because it is standard that for every family of M functions on b bits, and every ϵ, a uniformly chosen subset of size $t = O(\frac{\log M}{\epsilon})$ is with probability $1 - 2^{-\Omega(t)}$ a discrepancy set against the family, meaning that a uniformly chosen element from the subset is ϵ-pseudorandom for the family. The number of ROBPs of space s' is at most $M = 2^{O(N \cdot 2^{2s'})}$, and for every seed s, the set $\mathrm{Enc}_{\mathrm{ctrl}}(s,\cdot)$ was chosen uniformly. This means that assuming that $s' \geq \log N$, if we choose $s' \leq \ell/c$ and $\ell \leq d/c$ for a sufficiently large constant $c > 1$, we can argue that that for every s, with probability $1 - 2^{-\Omega(d)}$, $\mathrm{Enc}(s, U_d)$ is ϵ-pseudorandom for space s' ROBPs. We can do a union bound over all $2^\ell \leq 2^{\Omega(d)}$ choices of s, and obtain the pseudorandomness property.

- $\mathrm{Enc}_{\mathrm{ctrl}}$ is repetition decodable from $1/4 - o(1)$ relative errors.

  This follows because if we view $\mathrm{Enc}_{\mathrm{ctrl}}(s,r)$ as a function $\mathrm{Enc}(s \circ r)$, then by standard results on random codes, as we have taken the rate (which is $\ell + d/b$) to be $o(1)$, the relative distance of such a code will be $1/2 - o(1)$. This means that it is possible to uniquely decode (s,r) when given a received word v that is within distance $1/4 - o(1)$ from $\mathrm{Enc}(s,r)$. This immediately implies the existence of (a nonexplicit) repetition decoding algorithm up to $1/4 - o(1)$ relative errors.

This means that we obtain a stochastic control code which allows encoding of a string s of length $\ell = \Omega(b) = \Omega(N/n) = N^{1-o(1)}$ and fool ROBPs of size $\Omega(\ell) = N^{1-o(1)}$. Plugging such a code in our construction yields a final code with space $s = N^{1-o(1)}$. We do not go into the precise details of Theorem 5.3, because as we will soon explain that it is possible to improve Theorem 5.3 and achieve $s = N/\mathrm{polylog}(N)$ by using a nonexplicit PRG G_{robp}, instead of the choice made in Figure 1. We describe this approach below.

Linear codes with large dual distance. The argument above uses the nonexplicitness of $\mathrm{Enc}_{\mathrm{ctrl}}$ in a very strong way. However, we remark that the results of Koppparty, Shaltiel and Silbak [KSS19], together with the reduction from list-decoding to repetition decoding in Section 4, reduce the task of constructing control codes to that of constructing linear codes with distance roughly $2p$ and dual distance roughly s, with explicit list-decoding from roughly $2p$ relative errors. The control codes of [KSS19] are based on such a construction called the "Raw Reed-Solomon code" which only achieves $\ell = b^\alpha$ where $\alpha > 0$ is a constant that deteriorates when $\frac{1}{4} - p$ approaches zero. An explicit construction of codes that achieves larger dual distance (a probabilistic argument show that achieving $s = b^{1-o(1)}$ is possible) will translate into stochastic codes for space $N^{1-o(1)}$.

Using PRGs that fool size 2^s circuits. The proof of Theorem 5.3 becomes much easier if the generator G_{robp} from Figure 1, fools circuits rather than ROBPs. In that case, we needn't worry about "simulating adversaries by ROBPs" and can fool them directly.

This means that we no longer need to use locally-correctable and testable codes, and leads to a tighter connection between s and s', Specifcally, this allows $s = s'/\mathrm{polylog}N$ rather than $s = (s'/\mathrm{polylog}N)^{1-\xi}$ that is given by our analysis. This observation is useful in two setups:

Nonexplicit codes: It is standard that a uniformly chosen function G_{robp} will be $2^{-\Omega(\ell)}$-pseudorandom for circuits of size $2^{\Omega(\ell)}$. Using such nonexplicit PRGs, gives the space bound $s = N/(\log N)^{c \cdot c_\nu}$ for some

universal constant $c > 1$, and translates into a nonexplicit code for space $N/\text{polylog}N$ channels. We defer the precise details to a later version.

Explicit codes assuming one-way functions: Under the widely believed cryptographic assumption that one-way functions for subexponential size circuits exist, there are explicit PRGs that stretch ℓ bits into $M = 2^{\ell^\alpha}$ bits, and are $\frac{1}{M}$ pseudorandom for size M circuits. As explained earlier, using such PRGs removes the need for locally correctable and locally testable codes, meaning that the outer code in Theorem 3.10 can be chosen to be a code that is linear time encodable and decodable (as in [KSS19]). At the moment, assuming cryptographic assumptions give no advantage in final results over unconditional results, but we mention this possibility, as it may help in future research.

5.6 Discussion of other possible tradeoffs

In Section 1 we mention that it is possible to achieve explicit codes with some additional tradeoffs. We now sketch how to achieve these tradeoffs.

Codes for space $s = N^{\frac{1}{2}-o(1)}$ for $p < \frac{1}{8}$: As we have explained in the previous section, the reason that out explicit codes don't achieve space $N^{1-o(1)}$ is that we don't have sufficiently good explicit stochastic control codes. However, for $p < \frac{1}{8}$, Theorem 4.5 gives a stochastic control code which can achieve any $\alpha < \frac{1}{2}$. Plugging these codes into Theorem 5.3 gives the desired result.

Codes for space $s = N^{1-o(1)}$ for $p < p_1$ for some constant $p_1 > 0$: For small p, there are better constructions of stochastic control codes in [KSS19] (which are based on the algebraic geometric codes of Garcia and Stichtenoth [GS96]). Using these codes, gives the desired result. We defer the details to a later version.

Codes with almost linear time encoding and decoding: The list-decodable codes of Kopparty, Shaltiel and Silbak [KSS19] achieved quailinear (that is $N \cdot \text{polylog}(N)$) time encoding and (randomized) decoding. There are two reasons why the proof of Theorem 5.1 does not directly give the same time bounds:

- We use the locally-correctable and locally testable codes of Kopparty, Meir, Ron-Zewi and Saraf [KMRS17] from Theorem 2.12 rather than a code that is known to have linear time encoding and decoding which was used by previous work. Nevertheless, as explained in Remark 2.13, it is possible to obtain such codes with encoding and decoding time $n^{1+o(1)}$.
- For technical reasons, we need to choose the block size b to be much larger than the choice in [KSS19]. While this allows handling larger space (as can be seen in the code with space $s^{\frac{1}{2}-o(1)}$), it means that more work needs to be done in order to show that the control code has near linear time encoding. It is possible to achieve such control codes, with randomized (rather than deterministic) decoding. A precise statement was given in Theorem 4.4.

With these two modifications, we can apply Theorem 5.3 and obtain the same results as in Theorem 5.1 with encoding and (randomized) decoding in time $N^{1+o(1)}$.

6 Analysis of the construction of stochastic codes for space bounded channels

This section is devoted to proving Theorem 5.3, and show the correctness of the main construction.

The setup: Throughout the remainder of the section, we fix the setup of Theorem 5.3. Specifically, let $\delta > 0$, $0 \leq p < \frac{1}{4}$, $c_\nu \geq 1$ be constants, and let $\epsilon > 0$ be a sufficiently small constant. We will later choose sufficiently small constants $\epsilon', \epsilon_{\text{samp}} > 0$. In particular, we will choose ϵ_{samp} to be sufficiently small

so that $p_{\text{ctrl}} = p + \epsilon \geq p + 3\epsilon_{\text{samp}}$. Let N be sufficiently large, such that $N_{\text{data}} = (1-\epsilon')N$ is one of the infinitely many block lengths that are guaranteed in Theorem 3.10, as explained in Figure 1. We also receive a stochastic code $\text{Enc}_{\text{ctrl}} : \{0,1\}^{\ell} \times \{0,1\}^{d} \to \{0,1\}^{b}$, and we assume that all requirements in Figure 1 are satisfied.

Let $\text{Enc} : \{0,1\}^{RN} \times \{0,1\}^{\ell + n_{\text{ctrl}} \cdot d} \to \{0,1\}^{N}$ and $\text{Dec} : \{0,1\}^{N} \to \{0,1\}^{RN}$ be the functions specified in Figures 2, 3 using the ingredients and parameter choices in Figure 1.

6.1 Bounding the rate and running time of Enc, Dec

The rate of Enc. The rate R of Enc is given by:

$$R = \frac{R_{\text{BSC}} \cdot N_{\text{data}}}{N} = \frac{(1 - H(p_{\text{BSC}}) - \epsilon/3) \cdot (1-\epsilon') \cdot N}{N} \geq (1 - H(p+\alpha) - \epsilon/3) \cdot (1 - \epsilon/3).$$

We chose $\alpha = \frac{\epsilon}{10 \log \frac{1}{p}}$, so that $H(p+\alpha) \leq H(p) + \epsilon/10$. This holds because the derivative $H'(p)$ is decreasing in the interval $(0,1)$ and $H'(p) \leq \log(1/p)$. This means that $H(p+\alpha) \leq H(p) + \alpha \cdot H'(p) \leq H(p) + \epsilon/10$. Consequently,

$$R \geq (1 - H(p) - \epsilon/3 - \epsilon/10) \cdot (1 - \epsilon/3) \geq 1 - H(p) - \epsilon.$$

This proves the first item of Theorem 5.3

The running time of encoding. The encoding algorithm Enc of Figure 2 performs the following tasks:

- It applies the sampler of Theorem 2.7, to get $n_{\text{ctrl}} \leq n$ samples. This takes time $n \cdot \text{poly}(\log(N)^{c_\nu}) \leq N \cdot \text{poly}(\log(N)^{c_\nu})$.
- It applies the encoding of of Enc_{BSC} from Theorem 3.10. This takes time T_{data}.
- It applies the $(2^{-10t}, t)$-wise independent permutation π from Theorem 2.9, $N_{\text{data}} \leq N$ times for $t = \text{poly}((\log N)^{c_\nu})$. Each such application takes time $\text{poly}(t \cdot \log N) = \text{poly}((\log N)^{c_\nu})$, and overall, this takes time $N \cdot \text{poly}((\log N)^{c_\nu})$.
- It applies the PRG G, which in turn makes n calls to the PRG G_{robp} of Theorem 2.5. Each call obtains a pseudorandom string of length at most N that is $(2^{-10s'})$-pseudorandom for any-order space s' ROBPs (and takes time $N \cdot \text{poly}(\log N)$). Overall, n calls take time $N \cdot (\log N)^{c_\nu + O(1)}$.
- It applies Enc_{ctrl} on $n_{\text{ctrl}} \leq n$ pairs $(s, r_1), \ldots, (s, r_{n_{\text{ctrl}}}) \in \{0,1\}^{\ell} \times \{0,1\}^{d}$. Each application takes time T_{ctrl} and the overall time is $n \cdot T_{\text{ctrl}}$.

Overall, for a sufficiently large universal constant c_0, the total running time of Enc is bounded by $N \cdot (\log N)^{c_0 \cdot c_\nu} + n \cdot T_{\text{ctrl}} + T_{\text{data}}$. This proves the third item of Theorem 5.3.

The running time of decoding. The decoding algorithm Dec of Figure 3 performs additional steps (compared to the encoding algorithm). The additional steps are:

- It computes a list viable candidates. This list is of size at most n/τ which is a constant c that depends on $\epsilon', \epsilon_{\text{samp}}$ which in turn depend on p, ϵ.
- For each of the c candidates:
  - It applies the sampler of Theorem 2.7 (with the same parameter used in the encoding) to get t samples where $t = n_{\text{ctrl}} \leq n$. This takes time $n \cdot \text{poly}(\log(N)^{c_\nu}) \leq N \cdot \text{poly}(\log(N)^{c_\nu})$.
  - It applies the $(2^{-10t}, t)$-wise independent permutation π from Theorem 2.9 $N_{\text{data}} \leq N$ times for $t = \text{poly}((\log N)^{c_\nu})$ (same parameters as in encoding). Each such application takes time $\text{poly}(t \cdot \log N) = \text{poly}((\log N)^{c_\nu})$, and overall, this takes time $N \cdot \text{poly}((\log N)^{c_\nu})$.

- It applies the PRG G_{robp} of Theorem 2.5 n times, to obtain a pseudorandom string of length $N_{\mathrm{data}} \leq N$ that is $(2^{-s'})$-pseudorandom for any-order space s' ROBPs (same parameters as in encoding). This takes time $N \cdot (\log N)^{c_\nu + O(1)}$.
- It applies $\mathrm{Dec}_{\mathrm{BSC}}$ from Theorem 3.10. This takes time T_{data}.
- It computes repetition decoding using $a \leq n$ blocks, which takes time at most $n \cdot T_{\mathrm{ctrl}}$.

Overall, for a sufficiently large universal constant c_0, the total running time of Enc is bounded by $O(N \cdot (\log N)^{c_0 \cdot c_\nu} + n \cdot T_{\mathrm{ctrl}} + T_{\mathrm{data}})$. This proves the fourth item of Theorem 5.3.

6.2 Road map for arguing the correctness of decoding

The main part in proving Theorem 5.3 is showing that the decoding algorithm is correct. The remainder of this section is devoted to this proof, and in this subsection we give a roadmap of this proof.

The setup:

- Let $m \in \{0,1\}^{RN}$ be a message.
- Let $C : \{0,1\}^N \to \{0,1\}^N$ be a space s channel that induces at most pN errors.

We will keep these choices of m, C fixed throughout this section.

We need to show that w.h.p. the message m is decoded correctly when applying encoding, channel and decoding. We will refer to this experiment as the encoding/decoding experiment, and will denote it by $\mathrm{expr}^{\mathrm{ed}}(m, C)$. This experiment is described in full detail in Figure 4. Below is a brief sketch:

In this experiment $S \in \{0,1\}^\ell$ and $R \in (\{0,1\}^d)^{n_{\mathrm{ctrl}}}$ are chosen uniformly at random. $Z = \mathrm{Enc}(m, S, R)$ is the codeword, $E = C(Z)$ is the error pattern chosen by the channel, $\bar{V} = Z \oplus E$ is the received word given to the decoding, and $\bar{M} = \mathrm{Dec}(\bar{V})$ is the message returned by the decoding. We use the convention that capital letters denote the random variables associated with small letters used in the construction, and a complete specification of experiment $\mathrm{expr}^{\mathrm{ed}}(m, C)$ is given in Figure 4.

In order to complete the proof of Theorem 5.3 we need to show that the probability that the decoded message $\bar{M}$ is equal to m is large. That is, that:

$$\Pr_{\mathrm{expr}^{\mathrm{ed}}(m,C)}[\bar{M} = m] \geq 1 - \nu. \tag{2}$$

Recall that in the experiment, every candidate control string $\bar{s} \in \textit{VIABLE} \subseteq \textit{CANDIDATES}$ is used to produce a candidate message $\bar{M}(\bar{s})$. We first claim that w.h.p. the correct control string S is in *VIABLE* and that when decoding using this candidate we obtain the correct message m. (The earlier work of [KSS19] stopped here, and outputted the list of messages $\{\bar{M}(\bar{s}) : \bar{s} \in \textit{VIABLE}\}$). The next lemma is stating that this list indeed contains the original message m.

Lemma 6.1 (The correct message is list-decoded)**.**

$$\Pr_{\mathrm{expr}^{\mathrm{ed}}(m,C)}[S \in \textit{VIABLE and } \bar{M}(S) = m] \geq 1 - \nu/2.$$

Loosely speaking, this follows by the correctness of the list-decoding algorithm of [KSS19] which guarantees that the correct candidate control string appears in the list *VIABLE*, and that when decoding with this candidate, the original message m is obtained. We explain the technique of previous work [GS16, SS16, KSS19] in the next section.

The main contribution of this paper is that we achieve *unique decoding*. That is, our decoding algorithm is able to output a *single* candidate control string S^*, and we will show that w.h.p. $S^* = S$ (namely, that we identify the correct candidate). This is formally stated in the next lemma.

Figure 4: The encoding/decoding experiment $\mathrm{expr}^{\mathrm{ed}}(m, C)$.

Parameters: A message $m \in \{0,1\}^{RN}$ and a space s channel C.

Encoding phase: Choose uniformly at random $S \in \{0,1\}^{\ell}$ and $R \in (\{0,1\}^{d})^{n_{\mathrm{ctrl}}}$, and let $Z = \mathrm{Enc}(m, S, R)$. More specifically, divide S into three parts of length $\ell' = \ell/3$, so that $S = (S_{\mathrm{samp}}, S_{\mathrm{PRG}}, S_{\pi})$ and perform the following:

- $\mathit{CONTROL} = \{I_1 < \ldots < I_{n_{\mathrm{ctrl}}}\} = \mathrm{Samp}(S_{\mathrm{samp}})$.
- We denote the elements of $[n] \setminus \mathit{CONTROL}$ by $\{W_1, \ldots, W_{n_{\mathrm{data}}}\}$.
- $x = \mathrm{Enc}_{\mathrm{BSC}}(m)$
- $Y = \pi_{S_\pi}^{-1}(x)$.
- $Z \in \{0,1\}^{N}$ is defined as follows:
  - $Z_{\mathrm{data}}^{\mathit{CONTROL}} = Y \oplus G(S_{\mathrm{PRG}})$.
  - $Z_{\mathrm{ctrl}}^{\mathit{CONTROL}}$ is defined as follows: for every $j \in [n_{\mathrm{ctrl}}]$, $Z_{I_j} = \mathrm{Enc}_{\mathrm{ctrl}}(S, R_j)$.

Channel phase: Let $E = C(Z)$. More specifically:

- Apply C on Z and for $i \in [n]$, let $\mathrm{ST}_i \in \{0,1\}^{s}$ be the state of C after it reads the i'th block.
- Let $E = C(Z)$ and $\bar{V} = Z \oplus E$.

Decoding phase: Let $\bar{M} = \mathrm{Dec}(\bar{V})$. More specifically:

Compute viable candidates:

- For every $i \in [n]$, let $\bar{S}_i = \mathrm{Dec}_{\mathrm{ctrl}}(\bar{V}_i) = \mathrm{rDec}_{\mathrm{ctrl}}(1, \bar{V}_i)$.
- Let $\mathit{CANDIDATES} = \{\bar{S}_i : i \in [n]\}$.
- Let $\mathit{VIABLE} = \{\bar{s} \in \mathit{CANDIDATES} : \text{for at least } \tau \text{ choices of } i \in [n], \text{it holds that: } \bar{S}_i = \bar{s}\}$.

Decode using viable candidates: For every $\bar{s} \in \mathit{VIABLE}$, compute $\mathit{DecodeUsingCandidate}(\bar{s})$, more specifically:

- Let $\overline{\mathit{CONTROL}}(\bar{s}) = \mathrm{Samp}(\bar{s}_{\mathrm{samp}})$ and compute $\bar{V}_{\mathrm{data}}(\bar{s}) = \bar{V}(\bar{s})_{\mathrm{data}}^{\overline{\mathit{CONTROL}}}$.
- Let $\bar{Y}(\bar{s}) = \bar{V}_{\mathrm{data}}(\bar{s}) \oplus G(\bar{s}_{\mathrm{PRG}})$.
- Let $\bar{X}(\bar{s}) = \pi_{\bar{s}_\pi}(\bar{Y}(\bar{s}))$.
- Let $\bar{M}(\bar{s}) = \mathrm{Dec}_{\mathrm{BSC}}(\bar{X}(\bar{s}))$.

Compute valid candidates:

- For every $\bar{s} \in \mathit{VIABLE}$, determine whether $\bar{s}$ is successful, that is, if $\mathrm{outdist}(\bar{X}(\bar{s})) \leq \lambda_1/4$.
- Let VALID be the set of $\bar{s} \in \mathit{VIABLE}$ that are successful.

Compute active blocks:

- Let $\mathit{ACTIVE} = \{i : \bar{S}_i \in \mathit{VALID}\}$ and let $A = |\mathit{ACTIVE}|$.
- Let $J_1, \ldots, J_A$ be the indices in ACTIVE.

Perform repetition decoding: Let $S^* = \mathrm{rDec}_{\mathrm{ctrl}}(A, \bar{V}_{J_1}, \ldots, \bar{V}_{J_A})$.

Output message: Compute $\mathit{DecodeUsingCandidate}(S^*)$ and output $\bar{M} = \bar{M}(S^*)$.

Lemma 6.2 (The correct candidate survives pruning)**.**

$$\Pr_{\mathrm{expr}^{\mathrm{ed}}(m,C)}[S^* = S] \geq 1 - \nu/2.$$

Together, Lemmata 6.1 and 6.2 imply that with probability at least $1 - \nu$, we have that $S^* = S$ and $\bar{M} = \bar{M}(S^*) = \bar{M}(S) = m$. This means that (2) holds, and the correct message is decoded with probability

$1-\nu$, concluding the proof of Theorem 5.3.

It remains to prove Lemmas 6.1 and Lemma 6.2. The former is proven in Section 6.3 and the latter is proven in Section 6.4.

6.3 The correct message is list-decoded

In this subsection we prove Lemma 6.1. Loosely speaking, this follows because as the first step in our decoding, we use the list-decoding algorithm of [KSS19]. Specifically, we use the "milestones" technique of [KSS19] to prove that two "milestone events" (called the "control milestone" and "data milestone") occur with high probability in our experiment.

6.3.1 The milestones lemmas of [KSS19]

We first state the control milestone.

Lemma 6.3 (Control milestone from [KSS19])**.**

$$\Pr_{\text{expr}^{\text{ed}}(m,C)}[\frac{1}{n_{\text{ctrl}}}\cdot\sum_{i\in \textit{CONTROL}} wt(E_i) > p + 2\epsilon_{\text{samp}}] \leq \frac{\nu}{100}.$$

The control milestone states that the fraction of errors that were induced on the "control part" (namely the indices in *CONTROL*) is very close to p. Lemma 6.3 would follow immediately from the sampler guarantee if C is an *additive channel* (meaning that there exists a fixed $e \in \{0,1\}^N$ such that $C(Z) = e$). In that case, the error $E = e$ is independent of S_{samp} and so, when we sample a subset $\textit{CONTROL} \subseteq [n]$ of size n_{ctrl} using a sampler, we indeed obtain that:

$$\frac{1}{n_{\text{ctrl}}}\cdot\sum_{i\in \textit{CONTROL}} wt(E_i) \approx \frac{1}{n}\cdot\sum_{i\in[n]} wt(E_i) \leq p.$$

In our case, C is not necessarily an additive channel, and so as Z depends on S_{samp} (by construction), it might be that $E = C(Z)$ depends on S_{samp}, spoiling the argument above. Nevertheless, [KSS19] (building on earlier ideas of [GS16]) show that the pseudorandomness of G and Enc_{ctrl} can be used to argue that the channel C "cannot make $E = C(Z)$ depend on S_{samp}", so that the previous argument applies. We explain this argument in Remark 6.5 below.

We now state the data milestone. Loosely speaking, another property that holds if the channels C is additive and produces a fixed error pattern e is that when S_π is chosen uniformly and $\text{Enc}_{\text{BSC}}(m)$ is permuted by a permutation $\pi_{S_\pi}^{-1}$ (as is the case in our encoding), then from the "point of view of the decoding algorithm" the induced error pattern becomes distributed like $e' = \pi_{U_d}(e)$ (which is the scenario considered in Theorem 3.10).

Let A_m^ρ be the function from Theorem 3.10 when applied as in Figure 1 to give the code Enc_{BSC}. Recall that this is a function that receives an error pattern $e' \in \{0,1\}^{N_{\text{data}}}$ and for $\rho \leq \lambda_1$, by Lemma 3.11, if $A_m^\rho(e') = 1$ then applying the decoding algorithm Dec_{BSC} on the "received word" $v' = \text{Enc}_{\text{BSC}}(m) \oplus e'$, the message m is decoded, and that furthermore, during the concatenated decoding, $\text{outdist}(v')$ is at most ρ.

In Theorem 3.10 it is stated that in such a scenario, both $\Pr[\text{outdist}(v') \leq \lambda_1/10]$ and $\Pr[A_m^{\lambda_1/10}(e') = 1]$ are very close to one. "The data milestone" below states that this holds also in the case that C is not an additive channel. More precisely, the next lemma states that when applying the decoding algorithm with the correct string S, and computing the received data word $\bar{X}(S)$, if we set $E' = \text{Enc}_{\text{BSC}}(m) \oplus \bar{X}(S)$ to be the error pattern (relative to the encoding of the correct message m) then $A_m^{\lambda_1/5}(E')$ accepts with high probability. The precise statement follows:

Lemma 6.4 (Data milestone from [KSS19])**.** *Let $\lambda_1 > 0$ be the constant guaranteed in Theorem 3.10, and let $E' = \mathrm{Enc}_{\mathrm{BSC}}(m) \oplus \bar{X}(S)$. For a sufficiently small choice of the constant $\epsilon_{\mathrm{samp}} > 0$ the following holds:*

- $\Pr_{\mathrm{expr^{ed}}(m,C)}[\mathrm{outdist}(\bar{X}(S)) \leq \lambda_1/5] \geq 1 - \frac{\nu}{100}$.
- $\Pr_{\mathrm{expr^{ed}}(m,C)}[A_m^{\lambda_1/5}(E') = 1] \geq 1 - \frac{\nu}{100}$.

The proof of the milestones lemmas (Lemma 6.3 and Lemma 6.4) follows by the "milestones argument" used in [KSS19] building on [GS16]. Unfortunately, we cannot formally derive it by [KSS19] as we have made different parameter choices than the ones made in [KSS19] (but the proof follows in precisely the same way). Below is a sketch of the argument, and the precise details are deferred to the full version.

Remark 6.5 (High level intuition of the milestones argument)**.** *Loosely speaking, the milestones argument works as follows: Consider an alternative experiment to the encoding/decoding experiment, which we will call the "additive experiment". In this experiment, when the channel C produces the error vector E, rather than running C on the codeword Z, we run it on a uniform and independent string. This means that the error E generated in this experiment is* independent *of all other variables in the experiment, and in particular of S_{samp}, S_π. We claim that any property $P(E)$ of the error pattern such that:*

- *$P(E)$ holds with probability almost one in the additive experiment.*
- *$P(E)$ can be decided by a small space ROBP (that may depend on S_{samp}, S_π and m).*

We have that $P(E)$ holds with probability almost one in the real encoding/decoding experiment.

In order to show this, one claims that if P does not hold with probability almost one in the real encoding/decoding experiment, then P can be used to break the pseudorandomness of either G or $\mathrm{Enc}_{\mathrm{ctrl}}$. This follows using the fact that in the real encoding/decoding experiment, it can be argued that for every fixing of $m, S_{\mathrm{samp}}, S_\pi$, Z is pseudorandom for small space ROBPs.

We have already observed earlier that the properties $P(E)$ mentioned in Lemma 6.3 and Lemma 6.4 hold with probability almost one in the additive experiment, and it is not difficult to show that a (randomized) small space ROBP can decide these properties (w.h.p.).

6.3.2 Milestones imply list-decoding

Following [KSS19, GS16] the two milestone lemmas imply that the correct message is list-decoded. We now explain this argument. We first claim that the probability that the correct control candidate S is not viable is small.

Corollary 6.6. $\Pr_{\mathrm{expr^{ed}}(m,C)}[S \notin \mathit{VALID}] \leq \frac{\nu}{10}$*, and furhermore:*

$$\Pr_{\mathrm{expr^{ed}}(m,C)}[|\left\{j \in \mathit{CONTROL} : \bar{S}_{I_j} = S\right\}| < \tau] \leq \frac{\nu}{10}.$$

Proof. Lemma 6.3 says that:

$$\Pr_{\mathrm{expr^{ed}}(m,C)}[\frac{1}{n_{\mathrm{ctrl}}} \cdot \sum_{i \in \mathit{CONTROL}} \mathit{wt}(E_i) > p + 2\epsilon_{\mathrm{samp}}] \leq \frac{\nu}{100}.$$

By Markov's inequality, if

$$\frac{1}{n_{\mathrm{ctrl}}} \cdot \sum_{i \in \mathit{CONTROL}} \mathit{wt}(E_i) \leq p + 2\epsilon_{\mathrm{samp}},$$

then the fraction of $i \in [n_{\text{ctrl}}]$ such that $wt(E_i) > p + 3\epsilon_{\text{samp}}$ is at most $\frac{1}{1+\epsilon_{\text{samp}}} \leq 1 - \frac{\epsilon_{\text{samp}}}{2}$, for a sufficiently small choice of $\epsilon_{\text{samp}} > 0$. This means that for at least $\tau = \frac{\epsilon_{\text{samp}} \cdot n_{\text{ctrl}}}{2}$ of $i \in$ *CONTROL* we have that $wt(E_i) \leq p + 3\epsilon_{\text{samp}}$. We have already chosen ϵ_{samp} to be sufficiently small so that $p + 3\epsilon_{\text{samp}} \leq p + \epsilon = p_{\text{ctrl}}$. For every $i \in [$*CONTROL*$]$, if $wt(E_i) \leq p + 3\epsilon_{samp} \leq p_{\text{ctrl}}$ then by the properties of decoding (or repetition decoding) of Enc_{ctrl} we have that:

$$\text{Dec}_{\text{ctrl}}(\bar{V}_i) = \text{Dec}_{\text{ctrl}}(\text{Enc}(S, U_d) \oplus E_i) = S,$$

because $\text{Dec}_{\text{ctrl}}(\cdot) = \text{rDec}_{\text{ctrl}}(1, \cdot)$ decodes from p_{ctrl} relative errors. It follows that:

$$\Pr_{\text{expr}^{\text{ed}}(m,C)}[|\left\{j \in \textit{CONTROL} : \bar{S}_{I_j} = S\right\}| < \tau] \leq \frac{\nu}{10},$$

which proves the corollary. □

We now claim that when applying *DecodeUsingCandidate* on the correct control string S, we decode to $\bar{M}(S) = m$ with high probability.

Corollary 6.7. $\Pr_{\text{expr}^{\text{ed}}(m,C)}[\bar{M}(S) = m] \geq 1 - \nu/10$.

Proof. By the data milestone we have that for $E' = \text{Enc}_{\text{BSC}}(m) \oplus \bar{X}(S)$, it holds that:

$$\Pr_{\text{expr}^{\text{ed}}(m,C)}[A_m^{\lambda_1/5}(E') = 1] \geq 1 - \frac{\nu}{10}.$$

By Theorem 3.10, whenever $A_m^{\lambda_1/5}(e') = 1$ we have that $\text{Dec}_{\text{BSC}}(m \oplus e') = m$. By definition $\bar{X}(S) = \text{Enc}_{\text{BSC}}(m) \oplus E'$ and the corollary follows. □

Together, Corollaries 6.6 and 6.7 imply Lemma 6.1.

6.4 Strategy for proving that the correct message survives the pruning

In this section we state a strategy for proving Lemma 6.2. This is the main technical contribution of this paper, and it relies on the new machinery we have developped in Sections 3 and Section 4 as well as several additional ideas. We will state several lemmas from which we can derive Lemma 6.2. These lemmas will be proven in later sections.

Lemma 6.2 follows immediately from the next lemma by the repetition decoding properties of Enc_{ctrl}.

Lemma 6.8 (Active part is close to the correct control string)**.** *With probability at least* $1 - \nu/2$ *in the experiment* $\text{expr}^{\text{ed}}(m, C)$ *we have that:*

$$\frac{1}{A} \cdot \sum_{i \in \textit{ACTIVE}} \delta^{\text{Enc}_{\text{ctrl}}}(S, \bar{V}_i) \leq p_{\text{ctrl}}.$$

In order to prove Lemma 6.8 we will split active candidates $\bar{s}$ according to whether they came from the data blocks or from the control blocks (it is also possible that they came from both). We will argue that w.h.p. we have that:

- At least τ of the control blocks are active.
- Furthermore, on these active blocks the relative error induced by the channel is roughly p (which is smaller than p_{ctrl}).

If we could show that no data block is active, then Lemma 6.8 would follow (because on $i \in \mathit{CONTROL}$, $\bar{V}_i = Z_i \oplus E_i = \mathrm{Enc}_{\mathrm{ctrl}}(S, U_d) \oplus E_i$, meaning that the average distance in Lemma 6.8 is bounded by roughly p). We cannot rule out the possibility that few data blocks are active, but we will be able to show that:

- The number of data blocks that are active is small.

Together, the three properties will suffice to prove Lemma 6.8. These three properties are stated formally in the next definition and lemma.

Definition 6.9. *We will use the following notation:*

- *Let* $\mathit{ACC} = \mathit{CONTROL} \cap \mathit{ACTIVE}$.
- *Let* $\mathit{ACD} = ([n] \setminus \mathit{CONTROL}) \cap \mathit{ACTIVE}$.

Lemma 6.10. *With probability at least* $1 - \nu/2$ *in the experiment* $\mathrm{expr}^{\mathrm{ed}}(m, C)$ *we have that:*

- $|\mathit{ACC}| \geq \tau = \frac{\epsilon_{\mathrm{samp}} \cdot n_{\mathrm{ctrl}}}{2} = \frac{\epsilon_{\mathrm{samp}} \cdot \epsilon' \cdot n}{2}$.
- $\frac{1}{|\mathit{ACC}|} \cdot \sum_{i \in \mathit{ACC}} \mathsf{wt}(E_i) \leq p + 2\epsilon_{\mathrm{samp}}$.
- $|\mathit{ACD}| \leq \frac{n}{\log n}$.

We first prove that Lemma 6.8 follows from Lemma 6.10.

Proof. (of Lemma 6.8) For every $i \in \mathit{CONTROL}$, we have that

$$\bar{V}_i = Z_i \oplus E_i = \mathrm{Enc}_{\mathrm{ctrl}}(S, U_d).$$

It follows by Lemma 6.10 that:

$$\frac{1}{|\mathit{ACC}|} \cdot \sum_{i \in \mathit{ACC}} \delta^{\mathrm{Enc}_{\mathrm{ctrl}}}(S, \bar{V}_i) \leq \frac{1}{|\mathit{ACC}|} \cdot \sum_{i \in \mathit{ACC}} \mathsf{wt}(E_i) \leq p + 2\epsilon_{\mathrm{samp}}.$$

By definition $\mathit{ACTIVE} = \mathit{ACC} \cup \mathit{ACD}$ and $A = |\mathit{ACTIVE}|$. It follows that:

$$\begin{aligned}
\frac{1}{A} \cdot \sum_{i \in \mathit{ACTIVE}} \delta^{\mathrm{Enc}_{\mathrm{ctrl}}}(S, \bar{V}_i) &= \frac{1}{A} \cdot \sum_{i \in \mathit{ACC}} \delta^{\mathrm{Enc}_{\mathrm{ctrl}}}(S, \bar{V}_i) + \frac{1}{A} \cdot \sum_{i \in \mathit{ACD} \setminus \mathit{ACC}} \delta^{\mathrm{Enc}_{\mathrm{ctrl}}}(S, \bar{V}_i) \\
&\leq \frac{|\mathit{ACC}|}{A} \cdot (p + 2\epsilon_{\mathrm{samp}}) + \frac{1}{A} \cdot |\mathit{ACD}| \\
&\leq p + 2\epsilon_{\mathrm{samp}} + o(1) \\
&\leq p + 3\epsilon_{\mathrm{samp}},
\end{aligned}$$

and we have chosen $p + 3\epsilon_{\mathrm{samp}} \leq p_{\mathrm{ctrl}}$. □

The next lemma states that it is unlikely that many data blocks can agree on a successful candidate $\bar{s}$. This means in particular that a candidate $\bar{s}$ cannot become valid if it only has support from data blocks. More specifically, if a candidate does not "receive votes" from control blocks, then either it does not receive enough votes from data blocks to make it viable, or, if it does, then it will be unsuccessful, and will not be declared valid. For this purpose we introduce another threshold parameter, τ' that was already mentioned in Figure 1, and set to be:

$$\tau' = (\log N)^{c_\nu + 1}.$$

Lemma 6.11 (Behavior on data blocks)**.**

$$\Pr_{\mathrm{expr}^{\mathrm{ed}}(m,C)}[\exists j_1 < \ldots < j_{\tau'} \in [n] \setminus \textit{CONTROL} \text{ s.t. } \bar{S}_{j_1} = \ldots = \bar{S}_{j_{\tau'}} \textit{ and } \mathrm{outdist}(\bar{X}(\bar{S}_{j_1})) \leq \lambda_1/4] \leq \frac{\nu}{10}.$$

We now prove that Lemma 6.10 follows from Lemma 6.11.

Proof. (of Lemma 6.10) By Lemma 6.3 we have that:

$$\Pr_{\mathrm{expr}^{\mathrm{ed}}(m,C)}[\frac{1}{n_{\mathrm{ctrl}}} \cdot \sum_{i \in \textit{CONTROL}} \textit{wt}(E_i) \leq p + 2\epsilon_{\mathrm{samp}}] \geq 1 - \frac{\nu}{100}.$$

By Corollary 6.6 we have that:

$$\Pr_{\mathrm{expr}^{\mathrm{ed}}(m,C)}[|\left\{j \in [n_{\mathrm{ctrl}}] : \bar{S}_{I_j} = S\right\}| \geq \tau] \geq 1 - \nu/10.$$

By Lemma 6.4 we have that:

$$\Pr_{\mathrm{expr}^{\mathrm{ed}}(m,C)}[\mathrm{outdist}(\bar{X}(S)) \leq \lambda_1/5] \geq 1 - \nu/100.$$

By Lemma 6.11 we have that:

$$\Pr_{\mathrm{expr}^{\mathrm{ed}}(m,C)}[\forall j_1 < \ldots < j_{\tau'} \in [n] \setminus \textit{CONTROL} \text{ if } \bar{S}_{j_1} = \ldots = \bar{S}_{j_{\tau'}}, \text{ then } \mathrm{outdist}(\bar{X}(\bar{S}_{j_1})) > \lambda_1/4] \geq 1 - \frac{\nu}{10}.$$

Assume that these four events occur (this happens with probability at least $1 - \nu/2$). The second event says that S is viable. The third event implies that S is successful. It follows that S is valid. This means that every block i for which $\bar{S}_i = S$ is active. In particular, there are at least τ such blocks in *CONTROL* and the first item follows.

For the second item, we note that for every $i \in \textit{CONTROL}$, if $\textit{wt}(E_i) \leq p_{\mathrm{ctrl}}$ then

$$\bar{S}_i = \mathrm{Dec}_{\mathrm{ctrl}}(\bar{V}_i) = \mathrm{Dec}_{\mathrm{ctrl}}(Z_i \oplus E_i) = \mathrm{Dec}_{\mathrm{ctrl}}(\mathrm{Enc}_{\mathrm{ctrl}}(S, U_d) \oplus E_i) = S.$$

Therefore, for every $i \in \textit{CONTROL}$, if $\textit{wt}(E_i) \leq p_{\mathrm{ctrl}}$ then i is active, and $i \in \textit{ACC}$. This means that on every $i \in \textit{CONTROL} \setminus \textit{ACC}$ we have that $\textit{wt}(E_i) > p_{\mathrm{ctrl}}$. It follows that:

$$\begin{aligned}
\sum_{i \in \textit{ACC}} \mathsf{wt}(E_i) &= \sum_{i \in \textit{CONTROL}} \mathsf{wt}(E_i) - \sum_{i \in \textit{CONTROL} \setminus \textit{ACC}} \mathsf{wt}(E_i) \\
&\leq n_{\mathrm{ctrl}} \cdot (p + 2\epsilon_{\mathrm{samp}}) - (n_{\mathrm{ctrl}} - |\textit{ACC}|) \cdot p_{\mathrm{ctrl}} \\
&\leq n_{\mathrm{ctrl}} \cdot (p + 2\epsilon_{\mathrm{samp}}) - (n_{\mathrm{ctrl}} - |\textit{ACC}|) \cdot (p + 2\epsilon_{\mathrm{samp}}) \\
&\leq |\textit{ACC}| \cdot (p + 2\epsilon_{\mathrm{samp}}).
\end{aligned}$$

Where the second to last item uses that $p + 2\epsilon_{\mathrm{samp}} \leq p_{\mathrm{ctrl}}$. This gives the second item.

We now prove the third item. For a candidate $\bar{s} \in \textit{CANDIDATES}$ to become valid, it must first be viable and have the support of at least τ blocks (meaning that there are at least τ blocks i such that $\bar{S}_i = \bar{s}$). Secondly $\bar{s}$ needs to be successful. The number of candidates that collected support from at least $\tau/2$ blocks in *CONTROL* is bounded by $\frac{n_{\mathrm{ctrl}}}{\tau/2} = \frac{2n_{\mathrm{ctrl}}}{\tau}$. By the fourth event, any candidate that collected support from more than $\tau/2 \geq \tau'$ candidates from $[n] \setminus \textit{CONTROL}$ is not successful, and therefore not valid. It follows that the number of blocks $i \in [n] \setminus \textit{CONTROL}$ for which $\bar{S}_i$ is valid is bounded by $\frac{2n_{\mathrm{ctrl}}}{\tau} \cdot \tau'$. This is because every such block has to be successful, and so, in order to be viable, must be equal to one of the $\frac{2n_{\mathrm{ctrl}}}{\tau}$ candidates that received at least $\tau/2$ votes from blocks in *CONTROL* (as otherwise, the candidate cannot collect τ votes and become viable). For each choice of the $\frac{2n_{\mathrm{ctrl}}}{\tau}$ candidates that received at least $\tau/2$ votes from blocks in *CONTROL*, there are at most τ' blocks in $[n] \setminus \textit{CONTROL}$ which have the same candidate. Overall, we get that $|\textit{ACD}|$ (that is the number of active blocks in $[n] \setminus \textit{CONTROL}$) is at most $\frac{2n_{\mathrm{ctrl}}}{\tau} \cdot \tau' \leq \frac{n}{\log n}$ as required. □

This means that in order to prove Lemma 6.2, we now need to prove Lemma 6.11. The remainder of this section is devoted to proving Lemma 6.11. This is done in Section 6.5.

6.5 Behavior on data blocks: Proof of Lemma 6.11

We will prove Lemma 6.11 by using the pseudorandomness properties of our ingredients to connect the statement of Lemma 6.11 (which is in the encoding/decoding experiment, to a similar statement that occurs in an experiment where the codeword Z is chosen uniformly. We will then argue that the lemma holds in the latter experiment. In order to relate the two experiments, we will introduce several hybrid experiments.

6.5.1 The adversarial experiment

In order to prove Lemma 6.11 we will show that for every choice of $s_{\mathrm{samp}}, s_\pi \in \{0,1\}^{\ell'}$ the lemma holds conditioned on the event $\{S_{\mathrm{samp}} = s_{\mathrm{samp}}, S_\pi = s_\pi\}$. It will be convenient to describe the experiment that comes up in the lemma as a separate experiment (with the additional conditioning on specific values of s_{samp}, s_π). We call "the adversarial encoding/decoding experiment" and denote it by $\mathrm{expr}^{\mathrm{ad}}_{s_{\mathrm{samp}},s_\pi}(m,C)$. This experiment is stated precisely, in Figure 5.

In this experiment, we imagine that there is an "adversary" which tries to choose a channel C and an implementation of the ingredients of our decoding algorithm so that the "bad event" in Lemma 6.11 occurs with probability that is not small. More specifically, in the adversarial experiment we are only interested in a subset *DCANDIDATES* $\subseteq$ *CANDIDATES* of candidates that came from data blocks, and the adversary wins if there exists such a candidate $\bar{s} \in$ *DCANDIDATES* that was obtained on τ' data blocks, and is successful (meaning that: $\mathrm{outdist}(\bar{X}(\bar{s})) \leq \lambda_1/4$). Other candidates in *DCANDIDATES* are said to be "rejected".

The adversarial experiment is defined so that it immediately holds that:

Lemma 6.12. *If for every* $s_{\mathrm{samp}}, s_\pi \in \{0,1\}^{\ell'}$, $\Pr_{\mathrm{expr}^{\mathrm{ad}}_{s_{\mathrm{samp}},s_\pi}(m,C)}[\textit{Adversary wins}] \leq \nu/10$ *then Lemma 6.11 holds.*

Therefore, in order to prove Lemma 6.11 it is sufficient to prove the following Lemma.

Lemma 6.13. *If for every* $s_{\mathrm{samp}}, s_\pi \in \{0,1\}^{\ell'}$,

$$\Pr_{\mathrm{expr}^{\mathrm{ad}}_{s_{\mathrm{samp}},s_\pi}(m,C)}[\textit{Adversary wins}] \leq \nu/50.$$

The remainder of this section is devoted to proving Lemma 6.13.

6.5.2 Road map for proving Lemma 6.13.

We want to prove that the probability that the adversary wins the adversarial experiment is small. Our plan is the following:

- We will use the pseudorandomness of $\mathrm{Enc}_{\mathrm{ctrl}}$ and G to argue that if the adversary wins the adversarial experiment, then it wins in a version of the experiment where pseudorandom strings are replaced with random strings.
- We will use the evasiveness of the code $\mathrm{Enc}_{\mathrm{BSC}}$ to argue that the adversary cannot win the experiment where pseudorandom strings are replaced by random strings. (Recall that evasiveness says that a small space channel cannot make the decoding algorithm $\mathrm{Dec}_{\mathrm{BSC}}$ decode when the channel corrupts a random word).

Figure 5: The adversarial experiment $\mathrm{expr}^{\mathrm{ad}}_{s_{\mathrm{samp}},s_\pi}(m, C)$.

Parameters: A message $m \in \{0,1\}^{RN}$, a space s channel C, a sampler seed s_{samp} and a permutation seed s_π.

Encoding phase: Choose uniformly at random $S_{\mathrm{PRG}} \in \{0,1\}^{\ell'}$ and $R \in (\{0,1\}^d)^{n_{\mathrm{ctrl}}}$, and let $Z = \mathrm{Enc}(m, (s_{\mathrm{samp}}, s_\pi, S_{\mathrm{PRG}}), R)$. More specifically:

- Let $S = (s_{samp}, s_\pi, S_{\mathrm{PRG}})$. ***comment:*** In this experiment S_{PRG} is chosen at random, while s_{samp} and s_π are fixed.
- *control* $= \{i_1 < \ldots < i_{n_{\mathrm{ctrl}}}\} = \mathrm{Samp}(s_{\mathrm{samp}})$. ***comment:*** In this experiment *CONTROL* is fixed to a constant *control*.
- We denote the elements of $[n] \setminus$ *control* by $\{w_1, \ldots, w_{n_{\mathrm{data}}}\}$.
- $x = \mathrm{Enc}_{\mathrm{BSC}}(m)$
- $y = \pi^{-1}_{s_\pi}(x)$. ***comment:*** In this experiment, y is constant.
- $Z \in \{0,1\}^N$ is defined as follows:
  - $Z^{\textit{control}}_{\mathrm{data}} = y \oplus G(S_{\mathrm{PRG}})$.
  - $Z^{\textit{control}}_{\mathrm{ctrl}}$ is defined as follows: for every $j \in [n_{\mathrm{ctrl}}]$, $Z_{i_j} = \mathrm{Enc}_{\mathrm{ctrl}}(S, R_j)$.

Channel phase: Let $E = C(Z)$. More specifically:

- Apply C on Z and for $i \in [n]$, let $\mathrm{ST}_i \in \{0,1\}^s$ be the state of C after it reads the i'th block.
- Let $E = C(Z)$ and $\bar{V} = Z \oplus E$.

Decoding phase: Let $\bar{M} = \mathrm{Dec}(\bar{V})$. More specifically:

Check whether candidates from data blocks are successful: ***comment:*** In this experiment, we will only be interested in candidates from data blocks, and the adversary wins if there is a candidate $\bar{s}$ that is successful, and was decoded at least τ' times in data blocks.

- For every $i \in [n]$, let $\bar{S}_i = \mathrm{Dec}_{\mathrm{ctrl}}(\bar{V}_i) = \mathrm{rDec}_{\mathrm{ctrl}}(1, \bar{V}_i)$.
- Let *DCANDIDATES* $= \left\{\bar{S}_i : i \in [n] \setminus \textit{control}\right\}$. ***comment:*** In this experiment, we are only interested in candidates from data blocks.
- For every $\bar{s} \in$ *DCANDIDATES*, compute *DecodeUsingCandidate*$(\bar{s})$, more specifically:
  - Let $\overline{\textit{CONTROL}}(\bar{s}) = \mathrm{Samp}(\bar{s}_{\mathrm{samp}})$ and compute $\bar{V}_{\mathrm{data}}(\bar{s}) = \bar{V}(\bar{s})^{\overline{\textit{CONTROL}}}_{\mathrm{data}}$.
  - Let $\bar{Y}(\bar{s}) = \bar{V}_{\mathrm{data}}(\bar{s}) \oplus G(\bar{s}_{\mathrm{PRG}})$.
  - Let $\bar{X}(\bar{s}) = \pi_{\bar{s}_\pi}(\bar{Y}(\bar{s}))$.

Outcome of experiment: We say that the ***adversary wins*** if there exists $\bar{s} \in$ *DCANDIDATES* such that for at least τ' choices of $i \in [n] \setminus$ *control*, we have that $\bar{S}_i = \bar{s}$, and $\mathrm{outdist}(\bar{X}(\bar{s})) \leq \lambda_1/4$. (Every $\bar{s} \in$ *DCANDIDATES* that was not used to win the experiment is said to be "rejected").

If the adversary in the adversarial experiment could be implemented by a small space ROBP, then the first step in our plan would follow directly from the pseudorandomness of $\mathrm{Enc}_{\mathrm{ctrl}}$ and G.

A significant difficulty in implementing the first step in this plan is that the adversary (that tries to win this experiment) is *not* a small space ROBP. More precisely, this adversary is a procedure that receives the codeword Z, corrupts it by a space s channel C, and then performs additional computation on the received word $\bar{V} = Z \oplus C(Z)$. In this additional computation (that is described in detail in Figure 5):

- The adversary applies $\mathrm{Dec}_{\mathrm{ctrl}}$ (to compute the set *DCANDIDATES* of data candidates).
- On every $\bar{s} \in$ *DCANDIDATES*, the adversary applies the sampler $\mathrm{Samp}(\bar{s}_{\mathrm{samp}})$, permutation $\pi_{\bar{s}_\pi}$ and PRG $G(\bar{s}_{\mathrm{PRG}})$.

- The adversary also needs to compute $\mathrm{outdist}(\bar{X}(\bar{s}))$ on every candidate $\bar{s}$.

None of these steps are computable by small space ROBPs. Moreover, a generator G which needs to fool polynomial time adversaries that are able to apply G, inherently implies cryptographic assumptions such as one way functions, and therefore, we have no hope of arguing that the adversary cannot distinguish unless we assume the existence of one-way functions. (We remark that under the assumption that one-way functions exist, this argument can be performed).

Our goal is to argue that pseudorandomness against ROBPs (which we have unconditionally) suffices. We will need additional ideas in order to implement the first step and connect the probability that the adversary wins the adversarial experiment to the probability that it wins in a version of the experiment in which pseudorandom strings are replaced with random strings. Here is an outline of our plan (with pointers to relevant sections):

- We first consider a hybrid experiment (called the simulated adversarial experiment) in which the (pseudorandom) control blocks are replaced with random strings. By carefully analyzing the behavior of our decoding algorithm, and using the pseudorandomenss of $\mathrm{Enc}_{\mathrm{ctrl}}$, we can argue that if the adversary wins the adversarial experiment, then it also wins the simulated adversarial experiment. (Here, a difficulty is that it is only the channel C that is being fooled by the pseudorandomness of $\mathrm{Enc}_{\mathrm{ctrl}}$, while later steps in the computation of the adversary are not fooled). This part of the argument is done in Section 6.5.3.
- We then consider another hybrid experiment in which we replace (pseudorandom) data blocks with random strings. We would like to argue that if the adversary wins the simulated adversarial experiment, then it also wins this experiment. Again, a difficulty is that we cannot use the pseudorandomness of G directly, as the adversary (which is not simulated by a small space ROBP) is not fooled by G.
- We have chosen the generator G so that different data blocks are *truly independent* and not just indistinguishable from independent by small space ROBPs. We will argue that (once $\bar{S}_\pi$ and the states $\mathrm{ST} = (\mathrm{ST}_{w_1}, \ldots, \mathrm{ST}_{w_{n_{\mathrm{data}}}})$ of the channel C at the end of data blocks are fixed) the error $E_{w_1}, \ldots, E_{w_{n_{\mathrm{data}}}}$ that the channel induces on data blocks are independent (as each such error E_{w_j} is determined by the state of C at the beginning of the j'th data block, and the content of the j'th data block). This intuitively means that the adversary cannot make the candidates $\bar{S}_{w_j}$ of τ' different data blocks agree on the same value, unless this value can be "guessed in advance". This means that if the adversary wins the simulated adversarial experiment, then (except for small probability) it wins with a string $\bar{s}$ that can be guessed in advance. We call such strings "heavy" and it can be shown that their number is not too large. This is done in Section 6.5.4.
- For every fixed heavy string $\bar{s}$, we can consider a weaker version of the adversary which is hardwired with $\mathrm{Samp}(\bar{s}_{\mathrm{samp}}), \pi_{\bar{s}_\pi}$ and $G(\bar{s}_{\mathrm{PRG}})$, and only tries to win using $\bar{s}$. This "fixed candidate adversary" does not need to compute $\mathrm{Dec}_{\mathrm{ctrl}}$, Samp, π or G. However, it still needs to compute $\mathrm{outdist}(\bar{X}(\bar{s}))$.
- We use the local correcting and testing properties of the outer code in the construction of $\mathrm{Enc}_{\mathrm{BSC}}$ to show that on a fixed string $\bar{s}$, a small space ROBP can approximate $\mathrm{outdist}(\bar{X}(\bar{s}))$. More precisely, by Lemma 3.12 a small space ROBP *can* approximate the outer distance.
- This means that the "fixed candidate adversary" can be computed (at least approximately) by a small space ROBP, and so we can use the pseudorandomness of G to argue that the fixed candidate adversary cannot distinguish between the simulated adversarial experiment and the uniform adversarial experiment. This implies that if the adversary wins the simulated adversarial experiment, then one of a small number of fixed candidate adversaries wins the adversarial uniform experiment. This part of the argument is done in Section 6.5.5.

- Finally, we argue that by the evasiveness properties of $\mathrm{Enc}_{\mathrm{BSC}}$ for every fixed candidate adversary, the probability that it wins the adversarial uniform experiment is small. By doing a union bound over the small number of fixed candidate adversaries for heavy candidates, we conclude that the probability that the initial adversary wins the initial adversarial experiment is small, which is what we wanted to prove. This part of the argument is done in Section 6.5.6.

6.5.3 The simulated adversarial experiment

We will introduce several hybrid experiments in order to implement the plan sketched in Section 6.5.2. The first such experiment will be called the "simulated adversarial experiment" and denoted by $\mathrm{expr}^{\mathrm{sa}}_{s_{\mathrm{samp}},s_\pi}(m,C)$. In this experiment we make two modifications (relative to the adversarial experiment):

1. In the encoding phase, when preparing Z, rather than preparing $Z^{\textit{control}}_{\mathrm{ctrl}}$ in the way specified in the encoding algorithm, we will instead pick $Z^{\textit{control}}_{\mathrm{ctrl}} \in \{0,1\}^{N_{\mathrm{ctrl}}}$ uniformly at random, independently of all other variables.
2. In the outcome of the experiment, rather than allowing the adversary to win if $\mathrm{outdist}(\bar{X}(\bar{s})) \leq \lambda_1/4$, we will be more lenient and replace $\lambda_1/4$ by $\lambda_1/2$.

The next lemma states that in order to bound the probability that the adversary wins the adversarial experiment (and prove Lemma 6.13), it is sufficient to bound the probability that the adversary wins the simulated adversarial experiment.

Lemma 6.14. *For every $s_{samp}, s_\pi \in \{0,1\}^{\ell'}$,*

$$\Pr_{\mathrm{expr}^{\mathrm{ad}}_{s_{\mathrm{samp}},s_\pi}(m,C)}[\textit{Adversary wins}] \leq \Pr_{\mathrm{expr}^{\mathrm{sa}}_{s_{\mathrm{samp}},s_\pi}(m,C)}[\textit{Adversary wins}] + \frac{\nu}{100}.$$

Proof. Consider a version of these two experiments in which we also fix S_{PRG} to some constant s_{PRG}. As S_{PRG} is identically distributed in both experiments, it is sufficient to show the lemma for every such fixing. Let us fix $m, C, s_{\mathrm{samp}}, s_\pi, s_{\mathrm{PRG}}$ and to avoid clutter, we will denote by $\mathrm{expr}^{\mathrm{ad}}$ and $\mathrm{expr}^{\mathrm{sa}}$ the two experiments with all the fixings (including $S_{\mathrm{PRG}} = s_{\mathrm{PRG}}$).

We consider hybrid experiments between the adversarial experiment and the simulated experiment. More specifically, for fixed $m, C, s_{\mathrm{samp}}, s_\pi, s_{\mathrm{PRG}}$ (which we will omit from the notation to avoid clutter) for $0 \leq k \leq n_{\mathrm{ctrl}}$, let $\mathrm{expr}^{(k)}$ denote the experiment which is like $\mathrm{expr}^{\mathrm{ad}}$ with the following modifications:

1. In the encoding phase, when preparing Z, rather than preparing $Z^{\textit{control}}_{\mathrm{ctrl}}$ in the way specified in the decoding algorithm, we will do the following:
   - For $j \leq k$, we set $Z_{i_j} = \mathrm{Enc}_{\mathrm{ctrl}}((s_{\mathrm{samp}}, s_\pi, s_{\mathrm{PRG}}), R_j)$ (as in experiment $\mathrm{expr}^{\mathrm{ad}}$).
   - For $j > k$, we set Z_{i_j} to be a uniform b bit string (as in experiment $\mathrm{expr}^{\mathrm{sa}}$).
2. When deciding whether the adversary wins, we replace $\lambda_1/4$ by $\lambda_1/2 - \frac{k\lambda_1}{4n_{\mathrm{ctrl}}}$.

With these choices, it follows that:

- $\mathrm{expr}^{(0)} = \mathrm{expr}^{\mathrm{sa}}$, and
- $\mathrm{expr}^{(n_{\mathrm{ctrl}})} = \mathrm{expr}^{\mathrm{ad}}$.

Thus, by a hybrid argument, it is sufficient to show that for every $0 \leq k < n_{\mathrm{ctrl}}$,

$$\Pr_{\mathrm{expr}^{(k+1)}}[\text{Adversary wins}] \leq \Pr_{\mathrm{expr}^{(k)}}[\text{Adversary wins}] + 2^{-s'}.$$

Let us use the superscript (k) for random variables from experiment $\mathrm{expr}^{(k)}$. As is standard in hybrid arguments we can imagine that all these experiments happen in the same probability space, where initially, the only difference between the k'th experiment, and the $(k+1)$'th experiment is in the way the block $Z_{i_{k+1}}$ is selected. This view allows us to write events that mixes random variables from different experiment.

The key observation is that w.h.p. the sequence of states that the ROBP C sees at the end of blocks, is essentially identical in the two experiments. Specifically, we claim that:

Claim 6.15. *The statistical distance between* $(\mathrm{ST}_1^{(k)}, \ldots, \mathrm{ST}_n^{(k)})$ *and* $(\mathrm{ST}_1^{(k+1)}, \ldots, \mathrm{ST}_n^{(k+1)})$ *is at most* $2^{-s'}$.

Proof. (of claim) Let $i^* = i_{k+1}$. The two distributions are identical on $\mathrm{ST}_1, \ldots, \mathrm{ST}_{i^*-1}$, by definition. On the block i^*, $Z_{i^*}^{(k+1)} = \mathrm{Enc}_{\mathrm{ctrl}}((s_{\mathrm{samp}}, s_\pi, s_{\mathrm{PRG}}), R_{i^*}^{(k)})$ while $Z_{i^*}^{(k)}$ is uniform and independent.

It follows from the pseudorandomness property of $\mathrm{Enc}_{\mathrm{ctrl}}$ that the two distributions are $2^{-s'}$-close in statistical distance. This is because if the distributions of output states on the two distributions were $2^{-s'}$ far, then C would give rise to a space $s \leq s'$ ROBP distinguishing $\mathrm{Enc}_{\mathrm{ctrl}}((s_{\mathrm{samp}}, s_\pi, s_{\mathrm{PRG}}), U_d)$ from uniform (contradicting the pseudorandomness of $\mathrm{Enc}_{\mathrm{ctrl}}$).

The two experiments $\mathrm{expr}^{(k)}$ and $\mathrm{expr}^{(k+1)}$ produce the same distribution of states $\mathrm{ST}_{i^*+1}, \ldots, \mathrm{ST}_n$ if $\mathrm{ST}_{i^*}^{(k)} = \mathrm{ST}_{i^*}^{(k+1)}$, and the claim follows. □

We also have that:

Claim 6.16. *If* $(\mathrm{ST}_1^{(k)}, \ldots, \mathrm{ST}_n^{(k)}) = (\mathrm{ST}_1^{(k+1)}, \ldots, \mathrm{ST}_n^{(k+1)})$ *then:*

- *For every* $i \neq i_{k+1}$, $E_i^{(k)} = E_i^{(k+1)}$, *and* $\bar{V}_i^{(k)} = \bar{V}_i^{(k+1)}$.
- *DCANDIDATES*$^{(k)}$ = *DCANDIDATES*$^{(k+1)}$.
- *The Hamming distance of* $\bar{V}^{(k)}$ *and* $\bar{V}^{(k+1)}$ *is at most* b.
- *For every* $\bar{s} \in$ *DCANDIDATES*$^{(k)}$, *the Hamming distance of* $\bar{V}_{\mathrm{data}}^{(k)}(\bar{s})$ *and* $\bar{V}_{\mathrm{data}}^{(k+1)}(\bar{s})$ *is at most* b.

Proof. (of claim) Note that for all $i \neq i_{k+1}$, $Z_i^{(k)} = Z_i^{(k+1)}$. As C is an ROBP, we have that the error E_i that it induces on the i'th block, is a function of it's state at the beginning of the block (namely, ST_{i-1}) and of the block content (namely, Z_i). We also have that $\bar{V}_i = Z_i \oplus E_i$. This gives the first item.

In both experiments *DCANDIDATES* is a function only of $\bar{V}_{\mathrm{data}}^{\textit{control}}$ and does not depend on $\bar{V}_{i_{k+1}}$. This gives the second item.

The third item follows because except for the block i_{k+1} which is of length b, $\bar{V}$ is identical in the two experiments.

The fourth item follows from the second and third item. □

We are finally ready to tackle the main claim, namely that:

Claim 6.17.

$$\Pr_{\mathrm{expr}^{(k+1)}}[\textit{Adversary wins}] \leq \Pr_{\mathrm{expr}^{(k)}}[\textit{Adversary wins}] + 2^{-s'}.$$

Proof. By the two previous claims, except with probability $2^{-s'}$, all the four consequences of the previous claim hold. We will show that when this happens, if the adversary wins the $(k+1)$'th experiment then the adversary wins the k'th experiment, proving the claim.

If the $(k+1)$'th experiment is successful, then let $\bar{s} \in$ *DCANDIDATES*$^{(k+1)}$ be the control string that was used by the adversary to win the $(k+1)$'th experiment. We will show that $\bar{s}$ (which also belongs to *DCANDIDATES*$^{(k)}$ = *DCANDIDATES*$^{(k+1)}$) is not rejected in the k'th is experiment.

We have that the Hamming distance between $\bar{V}^{(k)}_{\text{data}}(\bar{s})$ and $\bar{V}^{(k+1)}_{\text{data}}(\bar{s})$ is at most b. It follows that the Hamming distance between $\bar{X}^{(k)}(\bar{s})$ and $\bar{X}^{(k+1)}(\bar{s})$ is at most b. We would like to express this distance as a relative distance. We have that:

$$\delta(\bar{X}^{(k)}(\bar{s}), \bar{X}^{(k+1)}(\bar{s})) \leq \frac{b}{N_{\text{data}}} = \frac{1}{n_{\text{data}}} \leq \frac{2}{n}$$

Let n_{in} be the (constant) block length from Theorem 3.10. We are allowed to choose the constant $\epsilon' > 0$ as a function of λ_1 and n_{in}. In particular, if we can choose $\epsilon' > 0$ so that $\frac{\lambda_1}{4\epsilon' \cdot n_{\text{in}}} \geq 2$. With this choice we can continue and have that:

$$\delta(\bar{X}^{(k)}(\bar{s}), \bar{X}^{(k+1)}(\bar{s})) \leq \frac{\lambda_1}{4\epsilon' \cdot n \cdot n_{\text{in}}} = \frac{\lambda_1}{4n_{\text{ctrl}} \cdot n_{\text{in}}},$$

(where the last inequality follows because we can choose $\epsilon' > 0$ to be sufficiently small as a function of λ_1, n_{in}, so that $\frac{\lambda_1}{4\epsilon' \cdot n_{\text{in}}} \geq 2$). We know that in the $(k+1)$'th experiment $\bar{s}$ was used to win the experiment. This means that

$$\text{outdist}(\bar{X}^{(k+1)}(\bar{s})) \leq \frac{\lambda_1}{2} - \frac{(k+1)\lambda_1}{4n_{\text{ctrl}}} \leq \lambda_1.$$

We will use Lemma 2.16 to bound $\text{outdist}(\bar{X}^{(k)}(\bar{s}))$ in terms of $\text{outdist}(\bar{X}^{(k+1)}(\bar{s}))$ and $\delta(\bar{X}^{(k)}(\bar{s}), \bar{X}^{(k+1)}(\bar{s}))$. In order to apply the lemma we need to check that:

$$\delta(\bar{X}^{(k)}(\bar{s}), \bar{X}^{(k+1)}(\bar{s})) \leq \frac{\lambda_1 - \text{outdist}(\bar{X}^{(k+1)}(\bar{s}))}{n_{\text{in}}}.$$

We have set up the parameters so that this indeed holds, and we can conclude that:

$$\text{outdist}(\bar{X}^{(k)}(\bar{s})) \leq \frac{\lambda_1}{2} - \frac{(k+1)\lambda_1}{4n_{\text{ctrl}}} + \frac{\lambda_1}{4n_{\text{ctrl}} \cdot n_{\text{in}}} \cdot n_{\text{in}} \leq \frac{\lambda_1}{2} - \frac{k\lambda_1}{4n_{\text{ctrl}}}.$$

This means that the adversary wins the k'th experiment using $\bar{s}$, and the claim and lemma follow. □

Overall, we obtain that:

$$\Pr_{\text{expr}^{\text{ad}}_{s_{\text{samp}}, s_\pi}(m,C)}[\text{Adversary wins}] \leq \Pr_{\text{expr}^{\text{sa}}_{s_{\text{samp}}, s_\pi}(m,C)}[\text{Adversary wins}] + n_{\text{ctrl}} \cdot 2^{-s'},$$

and by our choice of parameters $n_{\text{ctrl}} \cdot 2^{-s'} \leq \frac{\nu}{100}$. □

By Lemma 6.14 we are left with the task of bounding the probability that the adversary wins the simulated uniform experiment.

6.5.4 Bounding the probability that the adversary wins with a light candidate

We are working in the simulated adversarial experiment $\text{expr}^{\text{sa}}_{s_{\text{samp}}, s_\pi}(m, C)$ for some fixed s_{samp}, s_π. We now define a set H of candidate strings $\bar{s}$ which we call "heavy". Recall that $\text{ST} = (\text{ST}_1, \ldots, \text{ST}_n)$ are the sequence of states of C when reading Z, and note that ST is of length sn. For every $q \in (\{0,1\}^s)^n$ we consider the experiment expr^q which will be defined as

$$\text{expr}^q_{s_{\text{samp}}, s_\pi}(m, C) = (\text{expr}^{\text{sa}}_{s_{\text{samp}}, s_\pi}(m, C) | \text{ST} = q).$$

To avoid clutter, we will refer to this experiment as expr^q within this section. We will use a superscript of q to denote the random variables of the experiment expr^q. Random variables without superscripts are in the experiment $\text{expr}^q_{s_{\text{samp}}, s_\pi}(m, C)$.

We define the following sets:

$$H_{q,i} = \left\{ \bar{s} : \Pr[\bar{S}_i^q = \bar{s}] \geq \frac{1}{n^2} \right\}.$$
$$H_q = \bigcup_{i \in [n]} H_{q,i}.$$
$$H = \bigcup_{q \in \{0,1\}^{sn}} H_q.$$

It immediately follows that:

Lemma 6.18. *For every $q \in (\{0,1\}^s)^n$ such that $\Pr[\mathrm{ST} = q] > 0$, $|H_q| \leq n^3$, and consequently $|H| \leq 2^{sn} \cdot n^3$.*

Proof. By definition for every $q \in (\{0,1\}^s)^n$ such that $\Pr[\mathrm{ST} = q] > 0$, and every $i \in [n]$, $|H_{q,i}| \leq n^2$, and H_q is the union of n such sets. □

We claim that in the experiment expr^q, candidates are independent.

Lemma 6.19. *For every $q \in (\{0,1\}^s)^n$ such that $\Pr[\mathrm{ST} = q] > 0$, $\bar{S}_1^q, \ldots, \bar{S}_n^q$ are independent.*

Proof. The definition of G' was tailored so that $(Z_{\mathrm{data}})_1, \ldots, (Z_{\mathrm{data}})_{n_{\mathrm{data}}}$ are truly independent (and not just indistinguishable from independent). As Z_{ctrl} is uniform and independent of Z_{data} we have that $Z_1, \ldots, Z_n$ are independent. The variables $Z_1^q, \ldots, Z_n^q$ are also independent, as the condition $\{\mathrm{ST} = q\}$ can be seen as the conjunction of n separate conditions of the form $\{\mathrm{ST}_i = q_i\}$. It also follows that $E_1^q, \ldots, E_n^q$ are independent. This is because having fixed the states in ST to q, the error that the channel induces on a block is a function of the state at the beginning of the block (which is fixed) and the contents of the block. This gives that each E_i^q is a function only of Z_i^q. As $\bar{V}_i^q = Z_i^q \oplus E_i^q$, this gives that $\bar{V}_1^q, \ldots, \bar{V}_n^q$ are independent. As $\bar{S}_i^q = \mathrm{Dec}_{\mathrm{ctrl}}(\bar{V}_i^q)$, this gives that $\bar{S}_1^q, \ldots, \bar{S}_n^q$ are independent. □

The next lemma shows that the contribution of $\bar{s} \notin H$ to the win probability of the adversary. This will allow us to focus on candidates in H.

Lemma 6.20. *For every $s_{\mathrm{samp}}, s_\pi \in \{0,1\}^{\ell'}$,*

$$\Pr_{\mathrm{expr}^{\mathrm{sa}}_{s_{\mathrm{samp}}, s_\pi}(m,C)}[\textit{Adversary wins}] \leq \Pr_{\mathrm{expr}^{\mathrm{sa}}_{s_{\mathrm{samp}}, s_\pi}(m,C)}[\exists \bar{s} \in H : \textit{The adversary wins with } \bar{s}] + \nu/100.$$

Proof. We first claim that for every $q \in (\{0,1\}^s)^n$ such that $\Pr[\mathrm{ST} = q] > 0$:

$$\Pr_{\mathrm{expr}^q}[\text{Adversary wins}] \leq \Pr_{\mathrm{expr}^q}[\exists \bar{s} \in H : \text{The adversary wins with } \bar{s}] + \nu/100.$$

The lemma follows from this claim by noting that in the right hand side of the claim, q can be replaced by ST (as $\mathrm{ST} = q$ in expr^q), using the law of total probability.

We now prove the claim. Fix some $q \in (\{0,1\}^s)^n$ such that $\Pr[\mathrm{ST} = q] > 0$. By Lemma 6.19 we have that $\bar{S}_1^q, \ldots, \bar{S}_n^q$ are independent. In order for the adversary to win expr^q, there has to be a value $\bar{s} \in \textit{DCANDIDATES}^q$, and a subset B of τ' distinct indices in $[n] \setminus \textit{control}$, such that for every $j \in B$, $\bar{S}_j^q = \bar{s}$. We now show that the latter event has low probability if $\bar{s} \notin H_q$.

For every $i \in [n] \setminus \textit{control}$, and $\bar{s} \notin H_q$, we have that $\Pr[\bar{S}_i^q = \bar{s}] < 1/n^2$. For every fixed subset $B = \{i'_1, \ldots, i'_{\tau'}\}$, the probability that $\bar{S}_{i'_1}^q = \ldots = \bar{S}_{i'_{\tau'}}^q = \bar{s}$ for some $\bar{s} \notin H_q$, is smaller than $(\frac{1}{n^2})^{\tau'-1}$ because conditioned on $\bar{S}_{i'_1}^q \notin H_q$, each of the remaining $\tau' - 1$ independent candidates have probability less

than $1/n^2$ to be equal $\bar{S}^q_{i'_1}$. Taking a union bound over all $\binom{n}{\tau'} \leq n^{\tau'}$ subsets B of size τ', the probability that $\bar{S}^q_{i'_1} = \ldots = \bar{S}^q_{i'_{\tau'}} = \bar{s}$ for some $\bar{s} \notin H_q$, is smaller than:

$$\binom{n}{\tau'}\left(\frac{1}{n^2}\right)^{\tau'-1} \leq \frac{n^{\tau'}}{n^{2\tau'-2}} \leq \frac{1}{2^{\tau'}} = \frac{1}{2^{(\log N)^{c_\nu+1}}} \leq \frac{\nu}{100},$$

where the last inequality follows because $\tau' \geq (\log N)^{c_\nu+1}$. The claim now follows, because if the adversary wins expr^q, then except for probability $\nu/100$, the adversary wins using an element $\bar{s} \in H$. □

6.5.5 The fixed candidate adversary

We will now consider a version of the adversary that tries to win with some fixed candidate $\bar{s} \in \{0,1\}^\ell$. This adversary doesn't bother to check for viability or validity of $\bar{s}$. It only checks whether $\bar{s}$ is successful. This candidates is called "the fixed candidate adversary".

Figure 6: The fixed candidate adversary $\mathrm{Adv}^{C,\bar{s}}_{s_{\mathrm{samp}},s_\pi}(Z_{\mathrm{data}})$.

Parameters: A space s channel C, a sampler seed $s_{\mathrm{samp}} \in \{0,1\}^{\ell'}$, a permutation seed $s_\pi \in \{0,1\}^{\ell'}$, a control string $\bar{s}$.

Input: A string $Z_{\mathrm{data}} \in \{0,1\}^{N_{\mathrm{data}}}$.

Additional parameters: These additional parameters (which are functions of the parameters) are "hardwired" to the adversary.

- Let *control* $= \mathrm{Samp}(s_{\mathrm{samp}})$
- Let $\overline{\textit{control}} = \mathrm{Samp}(\bar{s}_{\mathrm{samp}})$.
- $G(\bar{s}_{\mathrm{PRG}})$.
- $\pi_{\bar{s}_\pi}(\cdot)$.

Operation:

Encoding phase:

- Choose uniformly at random $Z_{ctrl} \in \{0,1\}^{N_{\mathrm{ctrl}}}$.
- Prepare $Z = (Z_{\mathrm{ctrl}}, Z_{\mathrm{data}})^{\textit{control}}$. That is, prepare a string Z according to *control* where placing Z_{ctrl} and Z_{data} in the control and data blocks.

Channel phase: Let $E = C(Z)$ and $\bar{V} = Z \oplus E$.

Decoding phase: Try to win with $\bar{s}$. That is:

- Let $\bar{V}_{\mathrm{data}}(\bar{s}) = \bar{V}(\bar{s})^{\overline{\textit{control}}}_{\mathrm{data}}$.
- Let $\bar{Y}(\bar{s}) = \bar{V}_{\mathrm{data}}(\bar{s}) \oplus G(\bar{s}_{\mathrm{PRG}})$.
- Let $\bar{X}(\bar{s}) = \pi_{\bar{s}_\pi}(\bar{Y}(\bar{s}))$.
- Compute $\mathrm{outdist}(\bar{X}(\bar{s}))$.

Outcome of experiment: We say that the ***adversary wins*** if $\mathrm{outdist}(\bar{X}(\bar{s})) \leq \lambda_1/2$.

The precise definition of the fixed candidate adversary $\mathrm{Adv}^{C,\bar{s}}_{s_{\mathrm{samp}},s_\pi}(Z_{\mathrm{data}})$ is given in Figure 6. The definition of the fixed candidate adversary was tailored so that by a union bound over all $\bar{s} \in H$, we have that:

Lemma 6.21. *For every $s_{\mathrm{samp}}, s_\pi \in \{0,1\}^{\ell'}$. Let $x = \mathrm{Enc}_{\mathrm{BSC}}(m)$ and $y = \pi_{s_\pi}^{-1}(x)$ (as is done in the simulated adversarial experiment $\mathrm{expr}^{\mathrm{sa}}_{s_{\mathrm{samp}},s_\pi}(m,C)$).*

$$\Pr_{\mathrm{expr}^{\mathrm{sa}}_{s_{\mathrm{samp}},s_\pi}(m,C)}[\exists \bar{s} \in H : \textit{The adversary wins with } \bar{s}] \leq \sum_{\bar{s}\in H} \Pr_{S_{\mathrm{PRG}}\leftarrow U_{\ell'}}[\mathrm{Adv}^{C,\bar{s}}_{s_{\mathrm{samp}},s_\pi}(y \oplus G(S_{\mathrm{PRG}})) \textit{ wins}].$$

More precisely, this follows because the fixed state adversary is more relaxed than the adversary defined in the earlier experiment, and so whenever the latter wins, the former also wins.

Continuing with our plan, we now aim to bound the probability that the fixed candidate adversary Adv wins when it receives input $y \oplus G(S_{\mathrm{PRG}})$. If we could argue that Adv can be implemented by a space s' ROBP, then (as we have that G is pseudorandom for space s ROBP) we could bound this probability by the probability that Adv wins when it receives input $U_{N_{\mathrm{data}}}$. While we won't be able to argue that Adv can be implemented by a small space ROBP, we can prove that Adv can be approximated by a small space ROBP, which will allow us to use the argument above (accounting for the approximation error). We first introduce the notion of approximation that we will use.

Definition 6.22 (Adversary achieves small outer distance)**.** *We say that $\mathrm{Adv}^{C,\bar{s}}_{s_{\mathrm{samp}},s_\pi}$* **achieves distance** *λ on input z if when running $\mathrm{Adv}^{C,\bar{s}}_{s_{\mathrm{samp}},s_\pi}(z)$, when computing $\mathrm{outdist}(\bar{X}(\bar{s}))$ this outer distance is at most λ.*

With this definition, saying that Adv wins on z is equivalent to saying that Adv achieves distance $\lambda_1/2$. We will prove that:

Lemma 6.23 (Fixed candidate adversary is approximated by ROBP)**.** *There exists a function $\rho = 2^{-\frac{s'}{N^{o(1)}}}$ such that for every $0 \leq \eta \leq 1$, every $s_{\mathrm{samp}}, s_\pi \in \{0,1\}^{\ell'}$, and every $\bar{s} \in \{0,1\}^{\ell}$, there exist a distribution $\overline{\mathrm{Adv}}^{C,\bar{s}}_{s_{\mathrm{samp}},s_\pi,\eta}$ over space s' ROBPs such that for every input $w \in \{0,1\}^{N_{\mathrm{data}}}$, and every $\eta < 1 - \frac{1}{100}$:*

- $\Pr[\mathrm{Adv}^{C,\bar{s}}_{s_{\mathrm{samp}},s_\pi}(w) \textit{ achieves distance } \eta \cdot \lambda_1] \leq \Pr_{D\leftarrow\overline{\mathrm{Adv}}^{C,\bar{s}}_{s_{\mathrm{samp}},s_\pi,\eta+\frac{1}{100}}}[D(w)=1] + \rho.$
- $\Pr_{D\leftarrow\overline{\mathrm{Adv}}^{C,\bar{s}}_{s_{\mathrm{samp}},s_\pi,\eta}}[D(w)=1] \leq \Pr[\mathrm{Adv}^{C,\bar{s}}_{s_{\mathrm{samp}},s_\pi}(w) \textit{ achieves distance } (\eta+\frac{1}{100})\lambda_1] + \rho.$

Proof. Lemma 6.23 follows directly from Lemma 3.12, by considering a distribution $\overline{\mathrm{Adv}}^{C,\bar{s}}_{s_{\mathrm{samp}},s_\pi,\eta}$ over ROBPs of space s' which does the following: On input w, simulate $\mathrm{Adv}^{C,\bar{s}}_{s_{\mathrm{samp}},s_\pi}(w)$ while replacing the step of computing $\mathrm{outdist}(\bar{X}(\bar{s}))$ with the approximation guaranteed in Lemma 3.12 (that can be performed by a space s' ROBP) and outputting one if the obtained approximation is smaller than $\eta \cdot \lambda_1$. □

We can now implement our plan, and bound the probability that Adv achieves small distance on $y \oplus G(S_{\mathrm{PRG}})$ by the probability that Adv achieves small distance on $y \oplus U_{N_{\mathrm{data}}}$. Specifically:

Lemma 6.24. *For every $s_{\mathrm{samp}}, s_\pi \in \{0,1\}^{\ell'}$ and every $\bar{s} \in \{0,1\}^{\ell}$,*

$$\Pr_{S_{\mathrm{PRG}}\leftarrow U_{\ell'}}[\mathrm{Adv}^{C,\bar{s}}_{s_{\mathrm{samp}},s_\pi}(y\oplus G(S_{\mathrm{PRG}})) \textit{ wins}] \leq \Pr_{W\leftarrow U_{N_{\mathrm{data}}}}[\mathrm{Adv}^{C,\bar{s}}_{s_{\mathrm{samp}},s_\pi}(y\oplus W) \textit{ achieves distance } \tfrac{3}{4}\cdot\lambda_1] + 2^{-\frac{s'}{N^{o(1)}}}.$$

Proof. Within this proof, to avoid clutter, we omit the mention of C, s_{samp}, s_π and $\bar{s}$ which are fixed throughout. By using Lemma 6.23 and the fact that G is $2^{-2s'}$-pseudorandom against space s' ROBPs we have

that:

$$\begin{aligned}
\Pr_{S_{\mathrm{PRG}} \leftarrow U_{\ell'}}[\mathrm{Adv}(y \oplus G(S_{\mathrm{PRG}})) \text{ wins}] &\leq \Pr_{S_{\mathrm{PRG}} \leftarrow U_{\ell'}}[\mathrm{Adv}(y \oplus G(S_{\mathrm{PRG}})) \text{ achieves } \lambda_1/2] \\
&\leq \Pr_{\substack{S_{\mathrm{PRG}} \leftarrow U_{\ell'} \\ D \leftarrow \overline{\mathrm{Adv}}_{\frac{1}{2}+\frac{1}{100}}}}[D(y \oplus G(S_{\mathrm{PRG}})) = 1] + \rho \\
&\leq \Pr_{\substack{W \leftarrow U_{N_{\mathrm{data}}} \\ D \leftarrow \overline{\mathrm{Adv}}_{\frac{1}{2}+\frac{1}{100}}}}[D(y \oplus W) = 1] + \rho + 2^{-2s'} \\
&\leq \Pr_{W \leftarrow U_{N_{\mathrm{data}}}}[\mathrm{Adv}(y \oplus W) \text{ achieves distance } \tfrac{3}{4} \cdot \lambda_1] + 2\rho + 2^{-2s'} \\
&\leq \Pr_{W \leftarrow U_{N_{\mathrm{data}}}}[\mathrm{Adv}(y \oplus W) \text{ achieves distance } \tfrac{3}{4} \cdot \lambda_1] + 2^{-\frac{s'}{N^{o(1)}}}.
\end{aligned}$$

□

Note that as W is uniform, the distribution $y \oplus W$ is uniform and identical to W, and therefore, the input of Adv in the right hand side could be replaced by W. We are therefore left with the task of bounding:

$$\Pr_{W \leftarrow U_{N_{\mathrm{data}}}}[\mathrm{Adv}^{C,\bar{s}}_{s_{\mathrm{samp}},s_\pi}(W) \text{ achieves distance } \tfrac{3}{4} \cdot \lambda_1].$$

In this experiment, the channel is applied on a uniform input, and we need to bound the probability that the channel can achieve small outer distance. This is very similar to the evasiveness experiment of Theorem 3.3. We will use Theorem 3.3 to argue that the evasiveness property $\mathrm{Enc}_{\mathrm{BSC}}$ gives a bound on this probability.

6.5.6 Using evasiveness to bound the success probability of the fixed candidate adversary

In this section we conclude the proof of Lemma 6.13 and Lemma 6.11 by bounding the probability that the fixed state adversary achieves small distance. More specifically, we will prove that:

Lemma 6.25. *For every $s_{\mathrm{samp}}, s_\pi \in \{0,1\}^{\ell'}$, and every $\bar{s} \in \{0,1\}^{\ell}$,*

$$\Pr_{W \leftarrow U_{N_{\mathrm{data}}}}[\mathrm{Adv}^{C,\bar{s}}_{s_{\mathrm{samp}},s_\pi}(W) \textit{ achieves distance } \tfrac{3}{4} \cdot \lambda_1] \leq 2^{-\Omega(N)}.$$

Proof. Within this proof, we will refer to $\mathrm{Adv}^{C,\bar{s}}_{s_{\mathrm{samp}},s_\pi}$ as Adv to avoid clutter. The adversary Adv receievs an input $W \leftarrow U_{N_{\mathrm{data}}}$ and uses it to prepare Z. We will be considering two partitions of Z into control and data. The first one is according to *control* and the second is according to $\overline{\textit{control}}$. In order to avoid confusion let us use $Z_{\mathrm{ctrl}} = Z^{\textit{control}}_{\mathrm{ctrl}}$, $Z_{\mathrm{data}} = Z^{\textit{control}}_{\mathrm{data}} = W$ (as is done in the description of the adversary). We use $\bar{Z}_{\mathrm{ctrl}} = Z^{\overline{\textit{control}}}_{\mathrm{ctrl}}$ and $\bar{Z}_{\mathrm{data}} = Z^{\overline{\textit{control}}}_{\mathrm{ctrl}}$. This notation is consistent in the sense that all "barred" variables use the partition $\overline{\textit{control}}$.

We will prove the lemma for every fixing of $\bar{Z}_{\mathrm{ctrl}}$, and note that as $\bar{Z}_{\mathrm{ctrl}}$ and $\bar{Z}_{\mathrm{data}}$ are independent, fixing $\bar{Z}_{\mathrm{ctrl}}$ doesn't affect $\bar{Z}_{\mathrm{data}}$. For every such fixing, we can think of C as space s ROBP C' that receives a uniformly chosen $\bar{Z}_{\mathrm{data}}$ (using the hardwired fixed choice of $\bar{Z}_{\mathrm{ctrl}}$. The ROBP C' induces at most

$$pN = \frac{pN_{\mathrm{data}}}{1-\epsilon'} \leq p \cdot (1+2\epsilon') \cdot N_{\mathrm{data}} \leq (p+\epsilon') \cdot N_{\mathrm{data}}$$

errors, meaning that it induces at most $p + \epsilon' \leq p_{\mathrm{BSC}}$ relative errors (where the last inequality follows by choosing $\epsilon' > 0$ to be sufficiently small). Consequently, we can think about this experiment as if $\bar{Z}_{\mathrm{data}}$ is

chosen uniformly, a channel $C' \in \mathrm{Spc}^s_{p+\epsilon'}$ is applied so that $\bar{V}_{\mathrm{data}}(\bar{s}) = \bar{Z}_{\mathrm{data}} \oplus C'(\bar{Z}_{\mathrm{data}})$. Then, $\bar{V}_{\mathrm{data}}(\bar{s})$ is xored with a fixed string $G(\bar{s})$ to obtain $\bar{Y}(\bar{s})$, and then a fixed permutation $\sigma = \pi_{\bar{s}_\pi}$ is applied on $\bar{Y}(\bar{s})$ to give $\bar{X}(\bar{s})$.

This experiment is identical to the experiment in Theorem 3.3 except for xoring with a fixed string $G(\bar{s})$. We now explain that we can imagine that our experiment does not xor with the fixed string $G(\bar{s})$. More precisely, let us imagine a distribution $W' = W \oplus G(\bar{s}_{\mathrm{PRG}})$ (which is also distributed like $U_{N_{\mathrm{data}}}$) and a channel $C''(w) = C'(w \oplus G(\bar{s}))$ (which is also a space s channel). It follows that:

- The noise generated on W' is $C''(W') = C'(W)$.
- The word that is obtained applying C'' on W' is $\bar{V}(\bar{s}) = W' \oplus C''(W') = W \oplus C'(W) \oplus G(\bar{s}_{\mathrm{PRG}})$.
- This means that when xoring with $G(\bar{s}_{\mathrm{PRG}})$ we obtain

$$\begin{aligned}\bar{Y}(\bar{s}) &= \bar{V}(\bar{s}) \oplus G(\bar{s}_{\mathrm{PRG}}) \\ &= W' \oplus C''(W') \oplus G(\bar{s}_{\mathrm{PRG}}) \\ &= W \oplus C'(W) \oplus G(\bar{s}_{\mathrm{PRG}}) \oplus G(\bar{s}_{\mathrm{PRG}}) \\ &= W \oplus C'(W).\end{aligned}$$

- This means that $\bar{X}(\bar{s}) = \sigma(W \oplus C'(W))$ for a uniformly chosen W, and so, this is exactly the experiment considered in Theorem 3.3.

The outer code of $\mathrm{Enc}_{\mathrm{BSC}}$ corrects from λ_1 relative errors (where λ_1 is a constant that depends on ϵ). (This parameter is called λ in Theorem 3.3). This means that if we meet the conditions of Theorem 3.3, then the probability that $\mathrm{outdist}(\bar{X})(\bar{s})) \leq 0.99\lambda_1$ in our experiment is at most $2^{-\Omega_\epsilon(n_{\mathrm{out}})} = 2^{-\Omega_\epsilon(N_{\mathrm{data}})}$ (where the last inequality is because in $\mathrm{Enc}_{\mathrm{BSC}}$ the inner code has constant block length n_{in}). We now verify that we meet all the conditions of Theorem 3.3.

- The code $\mathrm{Enc}_{\mathrm{BSC}}$ was obtained from Theorem 3.10 choosing $p_{\mathrm{BSC}} = p + \alpha$.
- The constant λ_1 is a constant that goes to zero when ϵ goes to zero, and so we are allowed to assume that it is sufficiently small as a function of ϵ.
- The inner code in $\mathrm{Enc}_{\mathrm{BSC}}$ is using maximum likelihood decoding from $p_{\mathrm{BSC}} + \epsilon''$ relative errors, where ϵ'' is a constant that goes to zero with ϵ, and so we are allowed to assume that it is sufficiently small as a function of ϵ. Overall, the decoding reach of the code $\mathrm{Enc}_{\mathrm{BSC}}$ is at most $p + \alpha + \epsilon'$.
- We have chosen s so that $s = o(N) = o(n_{\mathrm{out}})$ and so we meet the condition on s.
- Following this discussion, we indeed have that for sufficiently small $\epsilon > 0$, we can choose the parameters so that:

$$\lambda_1 + (p + \alpha + \epsilon'') + (p + \epsilon') < \frac{1}{2} - \gamma_1,$$

  for some constant $\gamma_1 > 0$ (that depends on ϵ) and we indeed meet the conditions of Theorem 3.3.

We conclude that

$$\Pr_{W \leftarrow U_{N_{\mathrm{data}}}}[\mathrm{Adv}^{C,\bar{s}}_{s_{\mathrm{samp}},s_\pi}(W) \text{ achieves distance } \tfrac{3}{4} \cdot \lambda_1] \leq 2^{-\Omega_\epsilon(N)}.$$

We are suppressing constants that depend on ϵ and so for constant $\epsilon > 0$ this is $2^{-\Omega(N)}$. □

6.5.7 Putting things together: Proof of Lemmas 6.13 and 6.11

We finally have all the tools to Prove Lemma 6.13 which by Lemma 6.12 implies Lemma 6.11.

Proof. (of Lemma 6.13) For every $s_{samp}, s_\pi \in \{0,1\}^{\ell'}$, we bound $a := \Pr_{\mathrm{expr}^{\mathrm{ad}}_{s_{\mathrm{samp}},s_\pi}(m,C)}[\text{Adversary wins}]$ by applying Lemmas 6.14, 6.20, 6.21, 6.18, 6.24, 6.25 in sequence to get that for $x = \mathrm{Enc}_{\mathrm{BSC}}(m)$ and $y = \pi^{-1}_{s_\pi}(x)$:

$$
\begin{aligned}
a &= \Pr_{\mathrm{expr}^{\mathrm{ad}}_{s_{\mathrm{samp}},s_\pi}(m,C)}[\text{Adversary wins}] \\
&\leq \Pr_{\mathrm{expr}^{\mathrm{sa}}_{s_{\mathrm{samp}},s_\pi}(m,C)}[\text{Adversary wins}] + \frac{\nu}{100} \\
&\leq \Pr_{\mathrm{expr}^{\mathrm{sa}}_{s_{\mathrm{samp}},s_\pi}(m,C)}[\exists \bar{s} \in H : \text{The adversary wins with } \bar{s}] + 2 \cdot \frac{\nu}{100} \\
&\leq \sum_{\bar{s} \in H} \Pr_{S_{\mathrm{PRG}} \leftarrow U_{\ell'}}[\mathrm{Adv}^{C,\bar{s}}_{s_{\mathrm{samp}},s_\pi}(y \oplus G(S_{\mathrm{PRG}})) \text{ wins}] + 2 \cdot \frac{\nu}{100} \\
&\leq 2^{sn} \cdot n^3 \cdot \Pr_{S_{\mathrm{PRG}} \leftarrow U_{\ell'}}[\mathrm{Adv}^{C,\bar{s}}_{s_{\mathrm{samp}},s_\pi}(y \oplus G(S_{\mathrm{PRG}})) \text{ wins}] + 2 \cdot \frac{\nu}{100} \\
&\leq 2^{sn} \cdot n^3 \cdot \left(\Pr_{W \leftarrow U_{N_{\mathrm{data}}}}[\mathrm{Adv}^{C,\bar{s}}_{s_{\mathrm{samp}},s_\pi}(y \oplus W) \text{ achieves distance } \tfrac{3}{4} \cdot \lambda_1] + 2^{-\frac{s'}{N^{o(1)}}} \right) + 2 \cdot \frac{\nu}{100} \\
&= 2^{sn} \cdot n^3 \cdot \left(\Pr_{W \leftarrow U_{N_{\mathrm{data}}}}[\mathrm{Adv}^{C,\bar{s}}_{s_{\mathrm{samp}},s_\pi}(W) \text{ achieves distance } \tfrac{3}{4} \cdot \lambda_1] + 2^{-\frac{s'}{N^{o(1)}}} \right) + 2 \cdot \frac{\nu}{100} \\
&\leq 2^{sn} \cdot n^3 \cdot \left(2^{-\Omega(N)} + 2^{-\frac{s'}{N^{o(1)}}} \right) + 2 \cdot \frac{\nu}{100} \\
&\leq 2^{\left(2sn - \frac{s'}{N^{o(1)}}\right)} + 2 \cdot \frac{\nu}{100} \\
&\leq \frac{\nu}{10},
\end{aligned}
$$

where the last inequality follows by our choice that $\frac{s'}{s} \geq N^{\xi}$. □

7 Conclusion and Open Problems

Our construction give codes that for any $0 \leq p < \frac{1}{4}$ achieverate $1 - H(p)$ for space $N^{\Theta(1)}$ channels, while achieving nearly linear time, (specifically time $N^{1+o(1)}$) algorithms for encoding and decoding. Is it possible to get linear time algorithms? We remark that the method of [GS16, KSS19] (which we also use) requires at least time $N \cdot \log N$ when evaluating the t-wise independent permutation. This means that additional ideas are needed in order to achieve this goal, and it cannot be achieved by "optimizing the components".

This paper completely resolves the capacity of space $N^{1-o(1)}$ channels (even for space $N/\mathrm{polylog}(N)$). A natural open problem is to achieve explicit construction of codes for space $N^{1-o(1)}$ channels, and rate $1 - H(p)$. In this paper, we give such constructions for any $p \leq p_1$ where $p_1 > 0$ is some universal constant.

As explained in Section 1 and Section 5, our constructions rely on explicit constructions of list decodable linear codes with large dual distance. Loosely speaking, plugging in a code that has relative distance roughly $2p$, absolute dual distance slightly larger than s, and explicit list decoding from relative $2p$ errors, translates (using our machinery) to a stochastic code for Spc^s_p channels with rate $1 - H(p)$. Our main constriction utilizes the linear codes of Kopparty, Shaltiel and Silbak [KSS19]. For every $p < \frac{1}{4}$, these codes achieve $s = N^{\delta}$ for a constant $\delta > 0$ that depends on p. The additional aforementioned result (of handling large space

channels for $p \leq p_1$) follows by using the algebraic geometric codes of Garcia and Stichtenoth [GS96] which achieve larger dual distance, but smaller p. Any improvement in constructions of explicit list-decodable linear codes with large dual distance, immediately implies improvements in our construction.

Acknowledgements

We thank Iftach Haitner, Swastik Kopparty, Or Meir, Noga Ron-Zewi and Tal Yankovich for helpful discussions.

References

[BHR05] Eli Ben-Sasson, Prahladh Harsha, and Sofya Raskhodnikova. Some 3cnf properties are hard to test. *SIAM Journal on Computing*, 35(1):1–21, 2005. 16

[CJL15] Zitan Chen, Sidharth Jaggi, and Michael Langberg. A characterization of the capacity of online (causal) binary channels. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pages 287–296, 2015. 3

[DJLS13] Bikash Kumar Dey, Sidharth Jaggi, Michael Langberg, and Anand D Sarwate. Upper bounds on the capacity of binary channels with causal adversaries. *IEEE Transactions on Information Theory*, 59(6):3753–3763, 2013. 3

[DPW18] Stefan Dziembowski, Krzysztof Pietrzak, and Daniel Wichs. Non-malleable codes. *J. ACM*, 65(4):20:1–20:32, 2018. 12

[FK18] Michael A. Forbes and Zander Kelley. Pseudorandom generators for read-once branching programs, in any order. In *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS*, pages 946–955, 2018. 14

[For65] G David Forney. *Concatenated codes.* PhD thesis, Massachusetts Institute of Technology, 1965. 1, 3, 25, 26

[GI05] Venkatesan Guruswami and Piotr Indyk. Linear-time encodable/decodable codes with near-optimal rate. *IEEE Transactions on Information Theory*, 51(10):3393–3400, 2005. 1, 25, 26

[GKS13] Alan Guo, Swastik Kopparty, and Madhu Sudan. New affine-invariant codes from lifting. In *Proceedings of the 4th conference on Innovations in Theoretical Computer Science*, pages 529–540, 2013. 6, 8, 17

[Gol97] Oded Goldreich. A sample of samplers - a computational perspective on sampling (survey). *Electronic Colloquium on Computational Complexity (ECCC)*, 4(20), 1997. 14

[GS96] A. Garcia and H. Stichtenoth. On the asymptotic behavior of some towers of function fields over finite fields. *Journal of Number Theory*, 61(2):248–273, 1996. 42, 63

[GS16] Venkatesan Guruswami and Adam Smith. Optimal rate code constructions for computationally simple channels. *Journal of the ACM (JACM)*, 63(4):35, 2016. 1, 2, 3, 4, 5, 9, 12, 20, 21, 25, 26, 29, 30, 38, 40, 44, 46, 47, 62

[HL11] Ishay Haviv and Michael Langberg. Beating the gilbert-varshamov bound for online channels. In *2011 IEEE International Symposium on Information Theory Proceedings, ISIT 2011, St. Petersburg, Russia, July 31 - August 5, 2011*, pages 1392–1396, 2011.

[HRW19] Brett Hemenway, Noga Ron-Zewi, and Mary Wootters. Local list recovery of high-rate tensor codes and applications. *SIAM Journal on Computing*, (0):FOCS17–157, 2019. 17

[INW94] R. Impagliazzo, N. Nisan, and A. Wigderson. Pseudorandomness for network algorithms. In *Proceedings of the ACM Symposium on Theory of Computing*, pages 356–364, 1994. 14

[KMRS17] Swastik Kopparty, Or Meir, Noga Ron-Zewi, and Shubhangi Saraf. High-rate locally correctable and locally testable codes with sub-polynomial query complexity. *Journal of the ACM (JACM)*, 64(2):11, 2017. 6, 8, 16, 17, 21, 26, 42

[KNR09] E. Kaplan, M. Naor, and O. Reingold. Derandomized constructions of k-wise (almost) independent permutations. *Algorithmica*, 55(1):113–133, 2009. 15

[KSS19] Swastik Kopparty, Ronen Shaltiel, and Jad Silbak. Quasilinear time list-decodable codes for space bounded channels. *To appear in the 60th Annual Symposium on Foundations of Computer Science (FOCS)*, 2019. 1, 2, 3, 4, 5, 6, 9, 12, 14, 19, 20, 21, 25, 26, 27, 29, 30, 32, 38, 39, 40, 41, 42, 44, 46, 47, 62

[KSY14] Swastik Kopparty, Shubhangi Saraf, and Sergey Yekhanin. High-rate codes with sublinear-time decoding. *Journal of the ACM (JACM)*, 61(5):1–20, 2014. 17

[Lan04] Michael Langberg. Private codes or succinct random codes that are (almost) perfect. In *45th Symposium on Foundations of Computer Science (FOCS 2004)*, pages 325–334, 2004. 13

[Lip94] Richard J. Lipton. A new approach to information theory. In *11th Annual Symposium on Theoretical Aspects of Computer Science*, pages 699–708, 1994. 9, 12

[MPSW10] Silvio Micali, Chris Peikert, Madhu Sudan, and David A. Wilson. Optimal error correction for computationally bounded noise. *IEEE Trans. Information Theory*, 56(11):5673–5680, 2010. 13

[MRRW77] Robert McEliece, Eugene Rodemich, Howard Rumsey, and Lloyd Welch. New upper bounds on the rate of a code via the delsarte-macwilliams inequalities. *IEEE Transactions on Information Theory*, 23(2):157–166, 1977. 1, 3

[Nis92] N. Nisan. Pseudorandom generators for space-bounded computation. *Combinatorica*, 12(4):449–461, 1992. 14

[NZ96] N. Nisan and D. Zuckerman. Randomness is linear in space. *J. Comput. Syst. Sci.*, 52(1):43–52, 1996. 14

[RR99] R. Raz and O. Reingold. On recycling the randomness of states in space bounded computation. In *Proceedings of the Thirty-First Annual ACM Symposium on Theory of Computing*, pages 159–168, 1999. 14

[Smi07] Adam D. 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, SODA*, pages 395–404, 2007. 9, 12, 21, 25, 26

[SS16] Ronen Shaltiel and Jad Silbak. Explicit list-decodable codes with optimal rate for computationally bounded channels. In *APPROX/RANDOM*, pages 45:1–45:38, 2016. 1, 2, 4, 12, 20, 29, 30, 39, 44

[Vad04] Salil P. Vadhan. Constructing locally computable extractors and cryptosystems in the bounded-storage model. *J. Cryptology*, 17(1):43–77, 2004. 14

[Vid15] Michael Viderman. A combination of testability and decodability by tensor products. *Random Structures & Algorithms*, 46(3):572–598, 2015. 17

ECCC ISSN 1433-8092

https://eccc.weizmann.ac.il