


# New Constructions of Pseudorandom Codes

Surendra Ghentiyala ✉ 

Cornell University, Ithaca, NY, USA

Venkatesan Guruswami ✉ 

Simons Institute for the Theory of Computing and Department of EECS and Mathematics,  
University of California, Berkeley, CA, USA

---

## Abstract

Introduced in [9], pseudorandom error-correcting codes (PRCs) are a new cryptographic primitive with applications in watermarking generative AI models. These are codes where a collection of polynomially many codewords is computationally indistinguishable from random for an adversary that does not have the secret key, but anyone with the secret key is able to efficiently decode corrupted codewords. In this work, we examine the assumptions under which PRCs with robustness to a constant error rate exist.

1. We show that if both the planted hyperloop assumption introduced in [6] and security of a version of Goldreich’s PRG hold, then there exist public-key PRCs for which no efficient adversary can distinguish a polynomial number of codewords from random with better than  $o(1)$  advantage.
2. We revisit the construction of [9] and show that it can be based on a wider range of assumptions than presented in [9]. To do this, we introduce a weakened version of the planted XOR assumption which we call the weak planted XOR assumption and which may be of independent interest.
3. We initiate the study of PRCs which are secure against space-bounded adversaries. We show how to construct secret-key PRCs of length  $O(n)$  which are *unconditionally* indistinguishable from random by  $\text{poly}(n)$  time,  $O(n^{1.5-\epsilon})$  space adversaries.

**2012 ACM Subject Classification** Theory of computation  $\rightarrow$  Computational complexity and cryptography

**Keywords and phrases** Error-correcting codes, Watermarking, Pseudorandomness

**Digital Object Identifier** 10.4230/LIPIcs.APPROX/RANDOM.2025.54

**Category** RANDOM

**Related Version** *Full Version:* <https://arxiv.org/abs/2409.07580> [16]

**Funding** *Surendra Ghentiyala:* This work is supported in part by the NSF under Grants Nos. CCF-2122230 and CCF-2312296, a Packard Foundation Fellowship, and a generous gift from Google. This work was done while the author was visiting the Simons Institute for the Theory of Computing.

*Venkatesan Guruswami:* Research supported in part by a Simons Investigator award, and NSF grant CCF-2211972.

**Acknowledgements** The authors would like to thank Yinuo Zhang for helpful discussion. They would also like to thank Sam Gunn and Noah Stephens-Davidowitz for reviewing early drafts of this work. We also wish to thank Miranda Christ for the observation that our warmup implies that secret-key PRCs with  $\omega(1)$  alphabet size and robustness to constant rate errors is trivial to achieve.

## 1 Introduction

The ability of malicious actors to easily and cheaply generate large amounts of AI generated content is becoming a larger issue as generative AI models progress. Digital watermarking mitigates some of these concerns by offering a way to generate AI content which is later easily recognizable (to someone with the secret key) as AI generated. One may also hope to recover other information possibly embedded in the watermark at the time of creation, like date



© Surendra Ghentiyala and Venkatesan Guruswami;  
licensed under Creative Commons License CC-BY 4.0

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2025).

Editors: Alina Ene and Eshan Chattopadhyay; Article No. 54; pp. 54:1–54:22



Leibniz International Proceedings in Informatics  
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

of generation. Cryptographic watermarking leverages cryptography to give watermarking schemes with provable guarantees, as opposed to ad-hoc schemes. In this work, we expand the set of assumptions under which one can achieve cryptographic watermarking.

## 1.1 PRCs and Applications

An exciting recent work by Christ and Gunn [9] introduced the notion of pseudorandom error-correcting codes (PRCs) with the intent to watermark generative AI models. Informally, pseudorandom error-correcting codes are keyed coding schemes with the following three properties (see Definition 15 and Definition 16 for details).

1. Pseudorandomness: codewords are computationally indistinguishable from random for any algorithm which does not have the secret key.
2. Robustness: anyone with the secret key can decode corrupted codewords with overwhelming probability.
3. Soundness: any fixed  $x \in \{0, 1\}^n$  has a negligible probability (where the probability is over the key generation algorithm) of being decoded to a message by the decoding algorithm.

One of the beautiful insights of [9] is that PRCs can be used to watermark generative models if one simply reinterprets the generative model as a channel which corrupts the randomness it uses. Consider an abstracted polynomial time generative algorithm **Generate** that as part of its input takes in a random input seed  $r \in \{0, 1\}^n$ , and produces content  $t \in \{0, 1\}^n$ . We model an adversary trying to evade detection by a channel  $\mathcal{E}' : \{0, 1\}^n \rightarrow \{0, 1\}^n$  which corrupts the content  $t$  into  $\tilde{t}$ . Furthermore, we assume that there exists an algorithm **Recover** which recovers an approximation  $\tilde{r}$  of  $r$  from  $\tilde{t}$ . The channel  $\mathcal{E} = \text{Recover} \circ \mathcal{E}' \circ \text{Generate}$  then acts as a corrupting channel for the input seed  $r$ .

Say we wish to watermark the output of our generative model with some message  $m$ . Let  $c$  be a PRC codeword for  $m$ . Notice that if we run **Generate** seeded with  $c$ , rather than truly random  $r$ , we obtain several desirable properties.

1. Undetectability [10]: the pseudorandomness of PRC outputs guarantees that watermarked content is computationally indistinguishable from unwatermarked content. This guarantees that the quality of the outputs is not degraded by watermarking.
2. Tamper resistance: applying our PRC's robust decoding algorithm to the tampered AI generated content  $\mathcal{E}(c)$  lets us recover the watermark  $m$ . Therefore, the watermark is not removed by the tampering by  $\mathcal{E}'$  to the generated content.
3. Few false positives: the soundness property guarantees that for any fixed human generated text  $z_1 \dots z_n$ , with overwhelming probability, the decoding algorithm will not flag it as a corrupted codeword (and thus watermarked text).

In this work, we are concerned with new constructions of pseudorandom codes. The assumptions required for the construction of pseudorandom codes in [9] are relatively strong (see Section 1.3.2), and were subsequently weakened in the case of secret-key PRCs to the existence of a local weak pseudorandom function family [18].

We restrict ourselves to constructing zero-bit PRCs (the encoded message is always “1”, see Definition 18) which are robust to all channels which introduce errors at a rate of  $1/2 - \varepsilon$  (for constant  $\varepsilon$ ). As shown in [9], such constructions can be bootstrapped into constant rate PRCs (see Definition 17). Furthermore, [9, 18] show how to bootstrap such constructions into codes which are robust to other types of errors than just substitution errors.

## 1.2 Watermarking large language models

We wish to emphasize that the framework of watermarking using PRCs is not restricted to any one type of generative AI model. However, to help make the philosophy of watermarking using PRCs more concrete, we review how [9] instantiate a PRC based scheme for watermarking large language models (LLMs).

Imagine an abstracted model of an LLM which works over the binary alphabet and always outputs text of length  $n$ . Concretely, consider an efficiently computable function  $f : \{0, 1\}^* \times \{0, 1\}^* \rightarrow [0, 1]$  which takes in the prompt and the output text so far as the input and outputs the probability  $p \in [0, 1]$  that the next token will be 1. The use of the binary alphabet in  $f$  is without loss of generality since all tokens can be represented in binary. Text generation on a prompt  $y \in \{0, 1\}^*$  works by iteratively sampling  $z_i \leftarrow \text{Ber}(f(y, z_1 \dots z_{i-1}))$  for all  $i \in [1, n]$ . The final output of the LLM is then  $z_1 \dots z_n$ .

Let us now consider a different procedure to sample from the same distribution. We first sample  $x_1 \dots x_n$ , each independently from  $\text{Ber}(1/2)$ . To generate from the LLM on a prompt  $y \in \{0, 1\}^*$ , we iteratively sample  $z_i$  for  $i \in [1, n]$  as follows. Let  $p_i = f(y, z_1, \dots, z_{i-1})$ , if  $p_i \leq 1/2$ , sample  $z_i$  from  $\text{Ber}(2p_i x_i)$ , otherwise sample  $z_i$  from  $\text{Ber}(1 - (1 - x_i)(1 - p_i))$ . Note that since each  $x_i$  is sampled uniformly from  $\text{Ber}(1/2)$ ,  $z_i$  is still distributed as  $\text{Ber}(p_i)$ , and therefore the output distribution of the LLM on a prompt  $y$  remains unchanged from the previous example.

The key now is to create an LLM that samples  $x_1 \dots x_n$  from a pseudorandom error-correcting code. We will call this new LLM the watermarking LLM. We assume that the original LLM is a polynomial time algorithm (formally, we need a family of LLMs parameterized by input length for the notion of a polynomial time algorithm to make sense, but we omit such details for the sake of exposition). Therefore, the *pseudorandomness property* guarantees that the output distribution of the watermarking LLM is computationally indistinguishable from the case when  $x_1 \dots x_n$  are sampled at random, which we just saw is the same as the original LLM output distribution.

Furthermore, notice that if  $0 < p_i < 1$ , then  $z_i = x_i$  with probability greater than  $1/2$ . Therefore, if many  $p_i$  are bounded away from one, the output of the watermarking LLM is relatively close to the codeword  $x_1 \dots x_n$ . The watermarking LLM takes the codeword  $x_1 \dots x_n$  as one of its inputs and outputs  $z_1 \dots z_n$ , and in this way it functions as a corrupting channel. For sufficiently high entropy outputs, many  $p_i$  are sufficiently close to  $1/2$ , therefore  $z_1 \dots z_n$  is relatively close to  $x_1 \dots x_n$ , and anyone with the secret key can decode  $z_1 \dots z_n$ , thereby confirming that the output has been watermarked. Furthermore, the LLM output  $z = z_1 \dots z_n$  is also robust to corruptions by an adversary trying to evade detection since  $\tilde{z}$  will still be decoded by someone with a secret key assuming that  $\Delta(z, \tilde{z})$  is small (which it will be if the adversary does not make significant changes to  $z$ ). Therefore, watermarked and edited text corresponds to corrupted PRC codewords.

For a discussion of how to watermark LLM text using PRCs as well as a other application of PRCs (robust steganography), we refer the reader to [9].

## 1.3 Our results

For the purpose of watermarking, our PRCs usually need to be robust to  $p$ -bounded channels (see Definition 14). Informally, these are channels where an adversary can arbitrarily flip any  $pn$  bits of a codeword.

We begin with a warmup in which we present a construction under the minimal cryptographic assumption of one-way functions (Theorem 20). We view this warmup as a way to build intuition about PRCs and what assumptions we use to construct them. We also

view it as an interesting result telling us what parameters we should try to achieve in our constructions. The simple warmup PRC scheme shows that it is easy to build PRCs which are robust to any sub-constant noise rate over the binary alphabet or robust to any constant noise rate over increasing alphabet sizes. We therefore restrict our attention in the following sections to constructing PRCs which are robust to a constant noise rate, which surprisingly turns out to require much stronger assumptions and involved constructions.

### 1.3.1 Planted hyperloop construction

The planted hyperloop assumption, introduced in [6], asserts that a random 5-hypergraph is distinguishable from a random 5-hypergraph with a special  $\Theta(\log n)$  size 3-hypergraph planted in it with advantage at most  $o(1)$ . [6] show that if both the planted hyperloop assumption and the security of Goldreich's PRG [17] instantiated with the predicate  $P_5(x_1, \dots, x_5) = x_1 \oplus x_2 \oplus x_3 \oplus x_4 x_5$  hold, then public key cryptography exists. We show that that similar assumptions imply a type of public-key PRC.

► **Theorem 1** (informal version of Theorem 21). *Under the assumption used to construct public key cryptography in [6] and  $o(1)$ -pseudorandomness of Goldreich's PRG instantiated with the  $P_5(x_1, \dots, x_5) = x_1 \oplus x_2 \oplus x_3 \oplus x_4 x_5$  predicate, there exist public-key PRCs robust to  $p$ -bounded channels for constant  $p < 1/2$  and with  $o(1)$  pseudorandomness against PPT adversaries.*

Informally, by  $\gamma$  pseudorandomness here, we mean that any PPT algorithm can distinguish a polynomial number of samples from random with at most  $\gamma$  advantage.

This result mostly follows from observing that the [6] construction exhibits some robustness to errors. We do require some care since [6] are able to apply standard techniques to amplify security and correctness in their public key cryptography scheme, whereas such amplification techniques would break the robust decoding property needed in PRCs.

There are at least two ways to interpret Theorem 21. The more obvious is simply the construction of public-key PRCs from studied cryptographic assumptions. However, one can also view it as suggesting that either the planted hyperloop assumption or the security of Goldreich's PRG with the  $P_5$  predicate is a surprisingly strong assumption. In particular, since the only other known construction of public-key PRCs relies on fairly strong assumptions (see Section 1.3.2), this puts the assumptions of Theorem 21 into a select group of assumptions implying public-key PRCs.

### 1.3.2 Revisiting [9] and planted XOR assumption

In Section 5, we revisit the assumptions under which [9] construct PRCs. Their construction is secure if either of the following hold

1. The planted XOR assumption and polynomial security of LPN with constant noise rate
2.  $2^{O(\sqrt{n})}$  security of LPN

We revisit the first of these assumptions. While polynomial security of LPN with constant noise rate is a very well established cryptography assumption, the planted XOR assumption (introduced in [2]) is relatively new. It is therefore the most critical vulnerability in the [9] construction. Informally, the planted XOR assumption says that a random matrix  $G \in \{0, 1\}^{m \times n}$  modified so that  $O(\log n)$  rows xor to  $0^m$  is computationally indistinguishable from a truly random matrix. We therefore generalize and relax the assumption to what we call the weak planted XOR assumption (see Assumption 34). Informally, the weak planted XOR assumption (with noise rate  $\varepsilon$ ) says that a random matrix  $G \in \{0, 1\}^{m \times n}$  modified so that  $O(\log n)$  rows xor to a vector  $v$  sampled from  $\text{Ber}(m, \varepsilon)$  is computationally indistinguishable from a truly random matrix.

We observe that both LPN and the weak planted XOR assumption have a noise rate parameter,  $\eta$  and  $\varepsilon$  respectively. We show that there is a wide range of points along the  $\varepsilon, \eta$  parameter trade-off curve for which pseudorandom codes robust to a constant noise rate exist.

► **Theorem 32.** *For efficiently computable  $m = \text{poly}(n), t = O(\log n), \eta = o(1), \varepsilon = O(\log(m)/(\eta m))$  which are functions of  $n$  and constant  $p \in [0, 1/2)$ , if  $\text{XOR}_{m,t,\varepsilon}$  holds and  $\text{LPN}[\eta]$  holds, then there exists a  $(1 - \text{negl}(n), 1 - \text{negl}(n), \text{negl}(n))$ -public-key PRC which is robust to all  $p$ -bounded channels and pseudorandom against all PPT adversaries.*

One can choose to read this result as saying more about the planted XOR assumption than the construction of PRCs. We will see that adding noise to the planted xor assumption seems to weaken it (for which we have some minor evidence Section 5.2) and interacts nicely with the LPN assumption. This seems to imply that the weak planted xor assumption may be the next natural variant of the planted xor assumption to study.

### 1.3.3 Unconditional PRCs for space-bounded adversaries

A natural and fundamental question in this area is whether we can prove the unconditional existence of PRCs (not based on cryptographic conjectures). To this end, we initiate the study of PRCs which are pseudorandom against polynomial time, space-bounded adversaries. Here we rely on the results showing that the problem of learning sparse parities (possibly with noise) is hard for space-bounded adversaries [25, 21, 14].

► **Theorem 2 (informal).** *There exists a zero-bit PRC with codeword length  $O(n)$  that is robust to error rate  $p$  for any constant  $p < 1/2$  and is **unconditionally** pseudorandom against adversaries which have  $O(n^{3/2-\varepsilon})$  space and  $\text{poly}(n)$  time.*

We emphasize that these results are for the one-way space model in which the adversary has  $O(n^{3/2-\varepsilon})$  and tries to distinguish between a stream of random bits and a stream of codewords (and must write down any bits from the stream it wishes to recall later).

For context, in the space-bound cryptography setting, the best secret-key cryptography scheme where the communicating parties must store the entire length  $O(n)$  ciphertext is only secure against  $O(n^2)$  space adversaries. So while the gap between the power of the adversary and that of the players is admittedly small in our PRC (which is essentially a secret-key scheme with a robust decoding algorithm), the gap is still somewhat close to the best known for security against space bounded adversaries (where there is no requirement of robustness). To our knowledge, we are the first to study robust decoding schemes in the cryptographic space-bounded setting.

Unfortunately, our scheme is unlikely to be practical for watermarking generative AI as most generative models use more than  $O(n^{3/2})$  auxiliary space (where  $n$  is the size of the output of the generative model). One can therefore view this result as a first step towards practical unconditional PRCs. As the field of space-bounded cryptography progresses, one may hope we will eventually be able to construct PRCs which are pseudorandom against  $O(n^5)$  space and  $\text{poly}(n)$  time adversaries, which may indeed be practical for watermarking generative AI. Conversely, we believe that our scheme may already be useful for other use cases, such as robust steganography for particular types of steganographic channels (see [9] for details on robust steganography).

## 1.4 Further directions

1. The construction of public-key pseudorandom codes from unstructured assumptions is possibly the biggest question left open by this work. All known constructions rely on structured assumptions like hardness of the learning parity with noise problem or the planted hyperloop assumption. Even the construction of public-key PRCs from such a strong assumption as indistinguishability obfuscation would be new and interesting.
2. The weak planted XOR assumption introduced in Section 5 merits more cryptanalytic study. Can we find more evidence that such an assumption is indeed weaker than the standard planted XOR assumption?
3. It may also be interesting to study from a theoretical perspective whether there exist a general set of generative model properties such that watermarking models with those properties using some explicit error correcting codes (from some class of constructions) rather than a PRC does not significantly degrade quality of model outputs.

## 1.5 Related work

The idea of pseudorandom error-correcting codes was introduced in [9] with the intent of watermarking generative AI. They constructed a binary zero-bit encryption scheme robust to the bounded adversarial substitution channel and used that to construct a binary, constant rate PRC robust to both the bounded substitution channel and the random deletion channel. Followup work by Golowich and Moitra [18] showed a construction of pseudorandom codes where the alphabet size grows polynomially in the output length of the code from a zero-bit PRC. They showed how to use such large alphabet pseudorandom codes to watermark LLM texts so that they are robust to bounded edit-distance channels (channels allowing insertions, substitutions, and deletions). Interestingly, their construction assumes the existence of a  $O(\log n)$ -local weak pseudorandom function family. This is quite similar to Section 4 which (among other things), assumes the security of Goldreich’s PRG, which is  $O(1)$ -local.

PRCs are perhaps most closely related to backdoored pseudorandom generators. Backdoored PRGs (first introduced in [26]) are pseudorandom generators where anyone with a secret key can distinguish PRG outputs from random. Zero-bit PRCs can just as well be thought of as backdoored PRGs where the mechanism to distinguish PRG outputs from random is robust to errors in its input.

The planted hyperloop construction of public-key cryptography [6] is itself based on [4] and [17]. These all belong to lines of work labeled expander-based cryptography which utilize or change the structure of expander graphs to build cryptographic primitives [24].

Section 6 is based on [25, 21, 14], which show that the problem of learning sparse parities with noise is hard for space-bounded algorithms. These results are intimately connected to the area of space-bounded cryptography. In space-bounded cryptography (introduced in [23]), it is assumed all adversaries are space-bounded (have at most, say,  $o(n^2)$  space, where  $n$  is the message length). Unlike traditional cryptography, researchers have been able to prove unconditional results in the bounded storage setting [7, 12, 13].

## 2 Preliminaries

### 2.1 Notation

We will use the notation  $\binom{[n]}{k}$  to denote the set of all size  $k$  subsets of  $[n]$ . We also often use the notation  $x_{[a,b]}$  to denote bits  $a$  through  $b$  (inclusive) of the string  $x$ . We write  $\text{Ber}(n, \eta)$  to denote the distribution  $x_1 x_2 \dots x_n$  where each bit  $x_i \in \{0, 1\}$  is sampled independently from



$\text{Ber}(\eta)$ . We write  $\text{BSC}(p)$  to denote the binary symmetric channel with crossover probability  $p$ . This is the channel where each bit is flipped with probability  $p$  and remains the same with probability  $1 - p$ . For  $x, y \in \{0, 1\}^n$ ,  $\Delta(x, y) = |\{i : x_i \neq y_i\}|$  is the Hamming distance between  $x$  and  $y$ . We also use  $\mathcal{S}_{t,n} = \{x \in \{0, 1\}^n : |x| = t\}$  to denote the Hamming sphere of dimension  $n$  and radius  $t$ .

We will write  $x_1, \dots, x_n \leftarrow \mathcal{D}$  to denote sampling  $x_1, \dots, x_n$  each independently from a distribution  $\mathcal{D}$  and also occasionally overload this notation by writing  $x_1, \dots, x_n \leftarrow S$  to denote sampling  $x_1, \dots, x_n$  each independently and uniformly from the set  $S$ .

If  $a \in \{0, 1\}^n$  and  $b \in \{0, 1\}^m$ , then  $ab \in \{0, 1\}^{n+m}$  denotes the concatenation of  $a$  and  $b$ . For a matrix  $G \in \{0, 1\}^{n \times m}$ ,  $G_i \in \{0, 1\}^m$  is row  $i$  of  $G$ .

## 2.2 Probability and combinatorics

► **Definition 3.** We say a string  $a \in \{0, 1\}^n$  is  $\delta$ -biased if  $|\{i : a_i = 0\} - \{i : a_i = 1\}| \leq \delta n$ .

► **Lemma 4.** Let  $p_1, \dots, p_n \in [0, 1/2]$ , if  $X_i \sim \text{Bern}(p_i)$ , then

$$\Pr_{X_1, \dots, X_n} [X_1 \oplus \dots \oplus X_n = 0] = \frac{1}{2} \left( 1 + \prod_{i=1}^n (1 - 2p_i) \right).$$

► **Lemma 5** (Chernoff Bound [8]). Let  $X_1, \dots, X_n$  be independent random variables, each distributed as  $\text{Ber}(p)$ . Let  $\mu = np$ , and  $X = X_1 + \dots + X_n$ . If  $\delta \geq 0$ , then

$$\Pr_{X_1, \dots, X_n} [X \geq (1 + \delta)\mu] \leq e^{-\delta^2 \mu / (2 + \delta)}.$$

If  $0 < \delta < 1$ , then

$$\Pr_{X_1, \dots, X_n} [X \leq (1 - \delta)\mu] \leq e^{-\delta^2 \mu / 3}.$$

We will use the same insights as [9] to reduce the case of  $p$ -bounded adversarial channels to the case of the hypergeometric channel. For this we need the following lemma regarding the hypergeometric distribution. Let  $\text{Hyp}(N, K, n)$  denote the distribution of the number of good elements chosen when choosing  $n$  elements without replacement from a population of size  $N$  which contains  $K$  good elements.

► **Lemma 6** ([19]). Let  $X \sim \text{Hyp}(N, K, n)$  and  $p = K/N$ . Then for any  $0 < t < K/N$ ,

$$\Pr[X \geq (p + \varepsilon)n] \leq e^{-2\varepsilon^2 n}.$$

► **Lemma 7.** If  $0 \leq t \leq m \leq n$ ,  $X \sim \text{Hyp}(n, m, t)$ , then

$$\frac{1}{2} + \frac{1}{2} \min_{\frac{m-t}{n} \leq p_i \leq \frac{m}{n-t}} \prod_{i=1}^t (1 - 2p_i) \leq \Pr[X \text{ is even}] \leq \frac{1}{2} + \frac{1}{2} \max_{\frac{m-t}{n} \leq p_i \leq \frac{m}{n-t}} \prod_{i=1}^t (1 - 2p_i).$$

► **Corollary 8.** If  $0 \leq t \leq m \leq n$ ,  $X \sim \text{Hyp}(n, m, t)$  and  $p$  is a value maximizing  $|1 - 2p|$  subject to  $(m - t)/n \leq p \leq m/(n - t)$ , then

$$\Pr[X \text{ is even}] \leq \frac{1}{2} + \frac{1}{2} |1 - 2p|^t, \quad \text{and} \quad \Pr[X \text{ is odd}] \leq \frac{1}{2} + \frac{1}{2} |1 - 2p|^t.$$

► **Lemma 9.** Let  $X_1, \dots, X_Q$  be uniformly distributed over  $[N]$ .

$$\Pr_{X_1, \dots, X_Q} [\exists i \neq j, X_i = X_j] \leq \frac{Q^2}{N}.$$

► **Definition 10.** The statistical distance (also known as the total variation distance) of two distribution  $X$  and  $Y$  on a finite domain  $D$  is defined as

$$\Delta(X, Y) = \frac{1}{2} \sum_{z \in D} |\Pr[X = z] - \Pr[Y = z]|$$

We say two distributions  $X$  and  $Y$  are statistically indistinguishable if  $\Delta(X, Y) = \text{negl}(n)$ .

► **Fact 11.** Let  $A$  be a set and  $B \subseteq A$ . If  $X$  is uniformly distributed over  $A$ , and  $Y$  is uniformly distributed over  $B$ , then  $\Delta(X, Y) = 1 - |B|/|A|$ .

### 2.3 Indistinguishability and LPN

For a class of functions  $\varepsilon$ , we say two distribution ensembles  $\{D_n\}_{n \in \mathbb{N}}, \{E_n\}_{n \in \mathbb{N}}$  are  $\varepsilon$ -indistinguishable if for any probabilistic polynomial time, non-uniform adversary  $\mathcal{A}$ , there exists a function  $\varepsilon' \in \varepsilon$  such that

$$\left| \Pr_{x \leftarrow D_n} [\mathcal{A}(x) = 1] - \Pr_{x \leftarrow E_n} [\mathcal{A}(x) = 1] \right| \leq \varepsilon'(n)$$

We say that  $\{D_n\}_{n \in \mathbb{N}}$  and  $\{E_n\}_{n \in \mathbb{N}}$  are computationally indistinguishable if they are  $\text{negl}(n)$ -indistinguishable.

The learning parity with noise (LPN) assumption is going to be critical in Section 5. For a linear code specified by a generator matrix  $G$ , an LPN sample is generated by sampling a random codeword  $Gs$ , and then adding some Bernoulli distributed noise  $e$  to it. The LPN assumption says that an LPN sample is indistinguishable from random. Intuitively, the assumption says that noisy codewords from a random linear code are indistinguishable from random.

► **Assumption 12.** For  $\eta : \mathbb{N} \rightarrow \mathbb{R}$  which is a function of  $n$ , the  $\text{LPN}[\eta]$  assumption states that for all  $m = \text{poly}(n)$  and all probabilistic  $\text{poly}(n)$  time algorithm  $\mathcal{A}$ ,

$$\left| \Pr_{\substack{G \leftarrow \mathbb{F}_2^{n \times m}, \\ s \leftarrow \mathbb{F}_2^m, \\ e \leftarrow \text{Ber}(n, \eta)}} [\mathcal{A}(G, Gs + e) = 1] - \Pr_{\substack{G \leftarrow \mathbb{F}_2^{n \times m}, \\ u \leftarrow \mathbb{F}_2^n}} [\mathcal{A}(G, u) = 1] \right| = \text{negl}(n)$$

While Assumption 12 is stated for a single LPN sample, for a randomly sampled  $G$ , a polynomial number of LPN samples would still be computationally indistinguishable from random. This follows from a standard hybrid argument.

In our construction, we will actually rely on the following assumption where the secret  $s$  is sampled from the same distribution as the noise. Lemma 2 of [5] shows Assumption 13 implied by Assumption 12.

► **Assumption 13.** For  $\eta : \mathbb{N} \rightarrow \mathbb{R}$  which is a function of  $n$ , the  $\text{LPN}[\eta]$  assumption states that for all  $m = \text{poly}(n)$  and all probabilistic  $\text{poly}(n)$  time algorithm  $\mathcal{A}$ ,

$$\left| \Pr_{\substack{G \leftarrow \mathbb{F}_2^{n \times m}, \\ s \leftarrow \text{Ber}(m, \eta), \\ e \leftarrow \text{Ber}(n, \eta)}} [\mathcal{A}(G, Gs + e) = 1] - \Pr_{\substack{G \leftarrow \mathbb{F}_2^{n \times m}, \\ u \leftarrow \mathbb{F}_2^n}} [\mathcal{A}(G, u) = 1] \right| = \text{negl}(n)$$



## 2.4 Pseudorandom Codes

► **Definition 14** ([9]). We say that a length-preserving binary channel  $\mathcal{E} : \{0, 1\}^* \rightarrow \{0, 1\}^*$  is  $p$ -bounded if for all  $n \in \mathbb{N}$ ,  $\Pr_{x \leftarrow \{0, 1\}^n} [|\mathcal{E}(x) \oplus x| > pn] \leq \text{negl}(n)$ .

We now define secret and public key pseudorandom codes.

► **Definition 15** (Secret-key PRC [9]). Let  $\Sigma$  be a fixed alphabet. An  $(\alpha, \beta, \gamma)$ -secret-key pseudorandom error-correcting code (abbreviated as secret-key PRC) with robustness to a channel  $\mathcal{E} : \Sigma^* \rightarrow \Sigma^*$  and pseudorandomness against a class of adversaries  $\mathcal{C}$  is a triple of polynomial time randomized algorithms (KeyGen, Encode, Decode) satisfying

- (Syntax) There exists functions  $\ell, n, k : \mathbb{N} \rightarrow \mathbb{N}$  such that for all  $\lambda \in \mathbb{N}$ ,  $\text{KeyGen}(1^\lambda) \in \{0, 1\}^{\ell(\lambda)}$ ,  $\text{Encode} : \{1^\lambda\} \times \{0, 1\}^{\ell(\lambda)} \times \Sigma^{k(\lambda)} \rightarrow \Sigma^{n(\lambda)}$ , and  $\text{Decode} : \{1^\lambda\} \times \{0, 1\}^{\ell(\lambda)} \times \Sigma^* \rightarrow \Sigma^{k(\lambda)} \cup \{\perp\}$ .
- (Error correction, or robustness) For any  $\lambda \in \mathbb{N}$  and any message  $m \in \Sigma^{k(\lambda)}$ ,

$$\Pr_{\text{sk} \leftarrow \text{KeyGen}(1^\lambda)} [\text{Decode}(1^\lambda, \text{sk}, \mathcal{E}(x)) = m : x \leftarrow \text{Encode}(1^\lambda, \text{sk}, m)] \geq \alpha$$

- (Soundness) For any fixed  $c \in \Sigma^*$ ,

$$\Pr_{\text{sk} \leftarrow \text{KeyGen}(1^\lambda)} [\text{Decode}(1^\lambda, \text{sk}, c) = \perp] \geq \beta$$

- (Pseudorandomness) For any adversary  $\mathcal{A} \in \mathcal{C}$ ,

$$\left| \Pr_{\text{sk} \leftarrow \text{KeyGen}(1^\lambda)} [\mathcal{A}^{\text{Encode}(1^\lambda, \text{sk}, \cdot)}(1^\lambda) = 1] - \Pr_{\mathcal{U}} [\mathcal{A}^{\mathcal{U}}(1^\lambda) = 1] \right| \leq \gamma$$

where  $\mathcal{A}^{\mathcal{U}}$  means that the adversary has access to an oracle that, on any (even previously queried) input, outputs a freshly drawn uniform value from  $\Sigma^{n(\lambda)}$ .

► **Definition 16** (Public-key PRC [9]). Let  $\Sigma$  be a fixed alphabet. An  $(\alpha, \beta, \gamma)$ -public-key pseudorandom error-correcting code (abbreviated as public-key PRC) with robustness to a channel  $\mathcal{E} : \Sigma^* \rightarrow \Sigma^*$  and pseudorandomness against a class of adversaries  $\mathcal{C}$  is a triple of polynomial time randomized algorithms (KeyGen, Encode, Decode) satisfying

- (Syntax) There exists functions  $\ell_{\text{Dec}}, \ell_{\text{Enc}}, n, k : \mathbb{N} \rightarrow \mathbb{N}$  such that for all  $\lambda \in \mathbb{N}$ ,  $\text{KeyGen}(1^\lambda) \in \{0, 1\}^{\ell_{\text{Dec}}(\lambda)} \times \{0, 1\}^{\ell_{\text{Enc}}(\lambda)}$ ,  $\text{Encode} : \{1^\lambda\} \times \{0, 1\}^{\ell_{\text{Enc}}(\lambda)} \times \Sigma^{k(\lambda)} \rightarrow \Sigma^{n(\lambda)}$ , and  $\text{Decode} : \{1^\lambda\} \times \{0, 1\}^{\ell_{\text{Dec}}(\lambda)} \times \Sigma^* \rightarrow \Sigma^{k(\lambda)} \cup \{\perp\}$ .
- (Error correction, or robustness) For any  $\lambda \in \mathbb{N}$  and any message  $m \in \Sigma^{k(\lambda)}$ ,

$$\Pr_{(\text{sk}, \text{pk}) \leftarrow \text{KeyGen}(1^\lambda)} [\text{Decode}(1^\lambda, \text{sk}, \mathcal{E}(x)) = m : x \leftarrow \text{Encode}(1^\lambda, \text{pk}, m)] \geq \alpha$$

- (Soundness) For any fixed  $c \in \Sigma^*$ ,

$$\Pr_{(\text{sk}, \text{pk}) \leftarrow \text{KeyGen}(1^\lambda)} [\text{Decode}(1^\lambda, \text{sk}, c) = \perp] \geq \beta$$

- (Pseudorandomness) For any adversary  $\mathcal{A} \in \mathcal{C}$ ,

$$\left| \Pr_{(\text{sk}, \text{pk}) \leftarrow \text{KeyGen}(1^\lambda)} [\mathcal{A}^{\text{Encode}(1^\lambda, \text{pk}, \cdot)}(1^\lambda, \text{pk}) = 1] - \Pr_{\mathcal{U}} [\mathcal{A}^{\mathcal{U}}(1^\lambda, \text{pk}) = 1] \right| \leq \gamma$$

where  $\mathcal{A}^{\mathcal{U}}$  means that the adversary has access to an oracle that, on any (even previously queried) input, outputs a freshly drawn uniform value from  $\Sigma^{n(\lambda)}$ .

► **Definition 17.** For Definition 16 and Definition 15, we define  $k(\lambda)/n(\lambda)$  as the rate of a PRC.

► **Definition 18.** We say a PRC scheme is a zero-bit PRC scheme if the only message  $m$  that is ever encrypted is 1.

The image of the decoding function of a zero-bit scheme should only be  $\{1, \perp\}$  since we know that 0 is never encoded by the PRC. Informally, a zero-bit PRC requires only that we distinguish corrupted PRC outputs from strings which are not PRC outputs. We will focus on zero-bit PRCs since when  $\mathcal{C}$  is all PPT algorithms, [9] shows that the existence of a zero-bit secret-key or public-key PRC implies the existence of a secret-key or public-key PRC respectively which has essentially the same robustness as the original but a worse rate. See [9] for a formal statement.

Say we have a zero-bit encryption scheme where corrupted codewords are identified as such with probability  $\alpha(\lambda)$ , random words are identified as codewords with probability  $\alpha(\lambda) - 1/\text{poly}(n)$ , and any polynomial number of codewords are  $\gamma$ -indistinguishable from random. This is not quite a PRC since we do not have the soundness property. However, our next lemma shows that we can use such a scheme to construct a  $(1 - \text{negl}(\lambda), 1 - \text{negl}(\lambda), \gamma)$  zero-bit PRC.

► **Lemma 19.** Suppose that there exist PPT algorithms  $(\text{KeyGen}, \text{Encode}, \text{Decode})$  such that

1. There exists functions  $\ell_{\text{Dec}}, \ell_{\text{Enc}}, n, k : \mathbb{N} \rightarrow \mathbb{N}$  such that for all  $\lambda \in \mathbb{N}$ ,  $\text{KeyGen}(1^\lambda) \in \{0, 1\}^{\ell_{\text{Dec}}(\lambda)} \times \{0, 1\}^{\ell_{\text{Enc}}(\lambda)}$ ,  $\text{Encode} : \{1^\lambda\} \times \{0, 1\}^{\ell_{\text{Enc}}(\lambda)} \times \{1\} \rightarrow \Sigma^{n(\lambda)}$ , and  $\text{Decode} : \{1^\lambda\} \times \{0, 1\}^{\ell_{\text{Dec}}(\lambda)} \times \Sigma^* \rightarrow \{1, \perp\}$ .
2.  $n(\lambda) = \text{poly}(\lambda)$ .
3. For every  $d \leq p \cdot n(\lambda)$ ,  $d$ -hypergeometric channel  $\mathcal{E}$ , and a  $1 - \text{negl}(\lambda)$  fraction of keys  $(\text{sk}, \text{pk}) \leftarrow \text{KeyGen}(1^\lambda)$ ,

$$\Pr_{\mathcal{E}}[\text{Decode}(1^\lambda, \text{sk}, \mathcal{E}(x)) = 1 : x \leftarrow \text{Encode}(1^\lambda, \text{pk}, 1)] \geq \alpha(\lambda)$$

where the randomness is over the randomness of the encoding algorithm and the errors of  $\mathcal{E}$ .

4. There exists a  $\delta(n) = 1/\text{poly}(n)$  where  $\alpha(\lambda) - \delta(\lambda) \geq 1/\text{poly}(\lambda)$  such that for a  $1 - \text{negl}(\lambda)$  fraction of keys  $(\text{pk}, \text{sk}) \leftarrow \text{KeyGen}(1^\lambda)$ ,

$$\Pr_{x \leftarrow \{0, 1\}^n}[\text{Decode}(1^\lambda, \text{sk}, x) = 1] \leq \delta(\lambda) .$$

5. For any  $q = \text{poly}(\lambda)$ ,  $X_1, \dots, X_q \leftarrow \text{Enc}(1^\lambda, \text{pk}, 1)$  is  $\gamma$ -indistinguishable from  $Y_1, \dots, Y_q \leftarrow \{0, 1\}^{n(\lambda)}$ .

Then for every constant  $\varepsilon > 0$ , there exists of a zero-bit  $(1 - \text{negl}(\lambda), 1 - \text{negl}(\lambda), \gamma(\lambda))$ -public-key PRC robust to any  $(p - \varepsilon)$ -bounded channel and pseudorandom against any PPT adversary.

**Proof.** See full version. ◀

### 3 A warmup

Here we give informal an description of a zero-bit PRC schemes with is only robust to  $o(n)$  errors and pseudorandom against any PPT adversary. Our reasons for presenting this warmup are twofold. First, we believe it provides intuition into what types of assumptions are good for constructing PRCs. Second, we find it interesting that building PRCs which are

robust to any sub-constant error rate is surprisingly easy (Theorem 20) but building PRCs which are robust to a constant error rate seems to require much more involved constructions. Very similar PRF based constructions have already been described in [9, 10, 20, 1]. Though we note that those seem to achieve robustness to any  $o(1/\log n)$  error rate, whereas the following scheme is robust to any  $o(1)$  error rate.

We will examine a simple construction of PRCs, which, for *any*  $\tau(n) = \omega(1)$ , is robust against  $\text{BSC}(1/\tau(n))$ . We choose  $\text{BSC}(1/\tau(n))$  instead of  $(1/\tau(n))$ -bounded channels only for ease of presentation and one can achieve robustness to  $p$ -bounded channels by applying techniques similar to those used in Lemma 19.

► **Theorem 20.** *Let  $p(n)$  be any  $o(1)$  function. If one-way functions exist, then there exists a  $(1 - \text{negl}(n), 1 - \text{negl}(n), \text{negl}(n))$ -private-key PRC scheme robust to  $\text{BSC}(p(n))$  and pseudorandom against all PPT adversaries.*

**Proof.** See full version. ◀

We see that the minimal cryptographic assumption of one-way functions lets us achieve robustness to any  $o(1)$  error rate. We also observe that the presented PRC can be turned into a PRC over a larger alphabet  $|\Sigma| = \tau(n)$  which is robust to a constant error rate by simply identifying each block of  $\log_2(\tau(n))$  bits with a symbol in  $\Sigma$ . This has the benefit that the probability that some block of  $\sqrt{\tau(n)} \log(n)$  bits (in particular, one corresponding to  $x_i$ ) remains unchanged when the codeword is subjected to a constant error rate channel goes up to  $1 - \text{negl}(n)$ . This resolves the bottleneck in our robustness proof above and allows us to achieve robustness to a constant error rate. Therefore, it is trivial to construct PRCs with superconstant alphabet size. This sets a baseline and tells us that for a scheme to be considered non-trivial, it must be robust to a constant error rate and have constant alphabet size.

The critical weakness of the PRF construction is that for a codeword to be decoded correctly, all  $\omega(\log n)$  bits of some  $x_i$  must remain intact. One approach to fix this (used to construct secret-key PRCs in [18]) is to use a local weak PRF family so that if  $\Delta(x, x')$  being small implies that  $\Delta(f(x), f'(x))$  is small. A different approach is to aim for a scheme in which the decoding algorithm looks at a small number of bits of the codeword. In particular, if the decoding algorithm only looks at  $O(\log n)$  of the the codeword, we may be able to achieve robustness to a constant error rate. This is the intuition that guides all of our upcoming schemes. This is also the approach guiding the scheme proposed in [9].

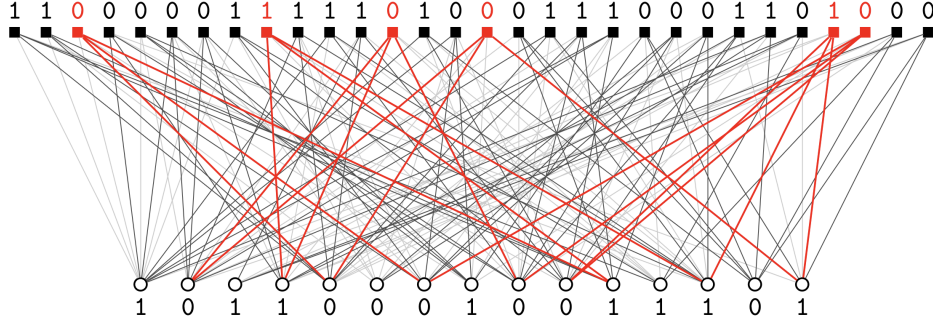
## 4 Planted hyperloop construction

In this section, we will use the planted hyperloop assumption and the security of Goldreich's PRG instantiated with the  $P_5$  predicate to construct a public-key PRC scheme.

► **Theorem 21.** *Let  $\delta, m, \ell, t$  be the parameters specified in Assumption 24 and  $p$  be a constant in  $[0, 1/2)$ . Under Assumption 24 and Assumption 25, there exists a  $(1 - \text{negl}(n), 1 - \text{negl}(n), o(1))$ -public-key PRC robust to any  $p$ -bounded channel and pseudorandom against all PPT adversaries.*

### 4.1 The assumptions

We begin by reviewing the assumptions used by [6] to construct public-key cryptography. All hypergraphs are assumed to have ordered hyperedges (a hyperedge is an ordered tuple of vertices rather than a set of vertices).



■ **Figure 1** A public key and PRG output with a single planted hyperloop  $L_0$ . The secret key is marked in red (from [6]).

► **Definition 22.** A *hyperloop* is a 3-hypergraph where each vertex has degree two and we define the size of a hyperloop as the number of hyperedges it contains.

The construction of [6] plants  $t = 2^{\Theta(\ell)}$  hyperloops  $S_1, \dots, S_t$  of size  $\ell = O(\log n)$  into a random hypergraph to create a 5-hypergraph  $H$ . The secret key is the set of  $t$  hyperloops and the public key is  $H$ . We now formally present this construction.

► **Construction 23 ([6]).** Let  $L_0$  be a fixed hyperloop of size  $\ell = O(\log n)$ .  $H$  is sampled as follows. Let:

1.  $L$  be the union of  $t = 2^{\Theta(\ell)}$  vertex-disjoint copies of  $L_0$ ,
2.  $Q$  be a random 3-hypergraph with  $n$  vertices and hyperedge probability  $O(n^{-3/2-\delta})$ ,
3.  $P = Q \cup L$  where  $L$  is planted on a random subset of the vertices of  $Q$ ,
4. If  $P$  has more than  $n^{3/2-\delta}$  hyperedges, output  $H = \perp$ . Otherwise,  $P'$  is obtained by adding random 3-hyperedges to  $P$  until it has  $m = n^{3/2-\delta}$  hyperedges.
5.  $H$  is obtained by randomly adding 2 vertices to each hyperedge in  $P'$  (where those 2 vertices will be the last two in the ordered hyperedge)

The public key is the 5-hypergraph  $H$  and the secret key is  $S_1, \dots, S_t$  where  $S_i \subseteq \{1, \dots, m\}$  are they hyperedges corresponding to the  $i^{\text{th}}$  planted copy of  $L_0$ .

For consistency, we define  $F_{\perp}(x) = 0^{(n^{-3/2-\delta})}$ . We refer to any hypergraph generated using Construction 23 as a planted hyperloop graph. This leads us to our first assumption.

► **Assumption 24.** For a sufficiently small constant  $\delta$ ,  $\ell = 0.36 \log n$ , and  $t = n^{0.75-\delta}$ ,  $P$  and  $Q$  are  $o(1)$ -indistinguishable in  $n^{O(1)}$  time.

Assumption 24 is the planted hyperloop assumption of [6] with the exception that it assumes  $o(1)$ -indistinguishability rather than  $(1 - \Omega(1))$ -indistinguishability.

Hypergraphs with certain parameters (including planted hyperloop hypergraphs) can be used as PRGs. To show how, we review Goldreich's PRG. Fix the predicate  $P_5(x_1, \dots, x_5) = x_1 \oplus x_2 \oplus x_3 \oplus x_4 x_5$ . For an  $n$  vertex,  $m$  hyperedge, 5-hypergraph  $H$ , we define the PRG  $F_H : \{0, 1\}^n \rightarrow \{0, 1\}^m$  as follows. On an input  $x$ , the bits of  $x$  are projected onto the vertices of  $H$ , and bit  $i$  of  $F_H(x)$  is given by applying  $P_5$  to the labeling of the vertices of hyperedge  $i$ .

Figure 1 gives a way to visualize Goldreich's PRG. We interpret our hypergraph  $H$  as a bipartite graph  $B$  where the input vertices of  $B$  represent vertices of  $H$ , the output vertices of  $B$  represent the hyperedges of  $H$ , and edge  $(a, b) \in B$  if and only if vertex  $a$  is contained in hyperedge  $b$  in  $H$ . See Figure 1 for the bipartite graph visualization with an example of the computation of  $F_H$ . Our second assumption is the security of Goldreich's PRG instantiated with the  $P_5$  predicate.

► **Assumption 25.** For every  $\delta$ ,  $m = n^{1.5-\delta}$ , and  $s = \text{poly}(n)$ , random  $Q$  belonging to set of 5-hypergraphs on  $n$  vertices with  $m$  hyperedges, and random  $x_1, \dots, x_s \in \{0, 1\}^n$ ,  $y_1, \dots, y_s \in \{0, 1\}^m$ ,  $(Q, F_Q(x_1), \dots, F_Q(x_s))$  and  $(Q, y_1, \dots, y_s)$  are  $o(1)$ -indistinguishable in  $n^{O(1)}$  time.

We now justify Assumption 25. Notice that it is too much to hope for  $\text{negl}(n)$ -indistinguishability in Assumption 25. To see why, notice that there is a  $\Omega(1/n^5)$  chance that the first two hyperedges in  $Q$  contain the exact same vertices in the same order, which would cause the first bit of the output of  $F_Q$  to always equal the second bit of the output. Such a function  $F_Q$  is clearly not a PRG.

Assumption 25 is in line with standard assumption about Goldreich's PRG [22, 11].

## 4.2 The construction

► **Construction 26** (Hyperloop Construction, Hyperloop $[\delta, m, \ell, t]$ ). Let  $\delta, m, \ell, t$  be efficiently computable functions of the security parameter  $n$ .

- **KeyGen**( $1^n$ ): Sample  $H$  and  $S$  as in Construction 23 conditioned on the last two vertices of each hyperedge in  $S_1$  being pairwise disjoint. output  $(sk = S, pk = H)$ .
- **Encode**( $1^n, H, 1$ ): Sample  $u \leftarrow \{0, 1\}^n$  and output  $F_H(u)$ .
- **Decode**( $1^n, S_1, x$ ): Compute  $w = \bigoplus_{j \in S_1} x_j$ . If  $w = 0$ , output 1, otherwise output  $\perp$ .

## 4.3 Robustness

We first review a basic fact about decoding in planted hyperloop graphs when the output is not subjected to errors and then show that this decoding mechanism is robust to errors.

► **Lemma 27** (Claim 1 in [6]). If  $H, S$  come from Construction 23 where the hyperedges of  $S_1$  are disjoint, if we sample  $x$  uniformly at random from  $\{0, 1\}^m$  and let  $y = F_H(x)$ , then  $w = \bigoplus_{j \in S_1} y_j$  has bias  $2^{-\ell}$  towards being 0.

**Proof.** See full version. ◀

We now use this to prove that the output of Goldreich's PRG instantiated with planted hyperloop graphs is indeed a robust to errors.

► **Lemma 28.** Let  $\delta, m, \ell, t$  be the parameters specified in Assumption 24 and  $p$  be any constant in  $[0, 1/2)$ . There exists some polynomial  $p(n)$  such that for Construction 26, for all  $d \leq pm$ , for all keys  $(sk, pk) \leftarrow \text{KeyGen}(1^n)$ ,

$$\Pr_{\mathcal{E}}[\text{Decode}(1^n, sk, \mathcal{E}(x)) = 1 : x \leftarrow \text{Encode}(1^n, pk, 1)] \geq \frac{1}{2} + \frac{1}{p(n)}.$$

Here  $\mathcal{E}$  is the  $d$ -hypergeometric channel and the randomness is over the randomness of the encoding algorithm and the errors of  $\mathcal{E}$ .

**Proof.** See full version. ◀

## 4.4 A form of soundness

► **Lemma 29.** Let  $\delta, m, \ell, t$  be the parameters specified in Assumption 24. For Construction 26, for any key pair  $(pk, sk) \leftarrow \text{KeyGen}(1^n)$ ,

$$\Pr_{x \leftarrow \{0, 1\}^n}[\text{Decode}(1^n, sk, x) = 1] = 1/2.$$

**Proof.** See full version. ◀

## 4.5 Pseudorandomness

► **Lemma 30.** *Let  $\delta, m, \ell, t$  be the parameters specified in Assumption 24. Under Assumption 24,  $H$  is  $o(1)$ -indistinguishable from a random 5-hypergraph with  $n$  vertices and  $m = n^{1.5-\delta}$  edges.*

**Proof.** See full version. ◀

► **Lemma 31.** *Let  $\delta, m, \ell, t$  be the parameters specified in Assumption 24. Under Assumption 24 and Assumption 25, the outputs of Construction 26 are  $o(1)$ -indistinguishable from random by any PPT adversary.*

**Proof.** See full version. ◀

## 4.6 Putting it all together

**Proof of Theorem 21.** See full version. ◀

The  $o(1)$  pseudorandomness in Theorem 21 is not ideal, but for the purpose of watermarking rather than security, seems tolerable. There is only a  $o(1)$  probability that watermarking will ever have any noticeable effect (to someone who does not have the secret key).

## 5 The weak planted XOR construction

Christ and Gunn [9] gave a scheme which is secure if both the planted XOR assumption and polynomial hardness of LPN with constant noise rate hold.<sup>1</sup> While polynomial hardness of LPN with constant noise rate is a well believed assumption, the planted XOR assumption is a non-standard and relatively unstudied assumption. Therefore, the conjunction of these two assumptions is quite a strong assumption. Therefore, it seems plausible that a scheme based on a strengthened LPN assumption and a weakened planted XOR assumption (denoted  $\text{XOR}_{m,t,\varepsilon}$ ) is more secure than the one presented in [9], and this is what we show in this section.

► **Theorem 32.** *For efficiently computable  $m = \text{poly}(n), t = O(\log n), \eta = o(1), \varepsilon = O(\log(m)/(\eta m))$  which are functions of  $n$  and constant  $p \in [0, 1/2)$ , if  $\text{XOR}_{m,t,\varepsilon}$  holds and  $\text{LPN}[\eta]$  holds, then there exists a  $(1 - \text{negl}(n), 1 - \text{negl}(n), \text{negl}(n))$ -public-key PRC which is robust to all  $p$ -bounded channels and pseudorandom against all PPT adversaries.*

### 5.1 The assumption

Let us define  $\mathcal{D}_0(m, n)$  as the uniform distribution over  $\{0, 1\}^{n \times m}$  and we will now define the distribution  $\mathcal{D}_1(n, m, t, \varepsilon)$  which corresponds to the distribution of matrices where we strategically implant a low weight vector in the row space.

► **Construction 33** (Generalization of [2]). *We define the distribution  $\mathcal{D}_1(n, m, t, \varepsilon)$*

1. *Sample  $G \leftarrow \{0, 1\}^{n \times m}$ ,*
2. *Choose a random tuple  $(a_1, \dots, a_t) \subseteq [n]^t$  such that  $i \neq j$  implies  $a_i \neq a_j$ ,*
3. *Let  $u = G_{a_1} \oplus \dots \oplus G_{a_{t-1}}$ ,  $v \leftarrow \text{Ber}(m, \varepsilon)$ , and update  $G_{a_t}$  to  $u + v$*
4. *Output  $(G, s)$  where  $s \in \{0, 1\}^n$  is the  $t$  sparse indicator vector for  $(a_1, \dots, a_t)$ .*

---

<sup>1</sup> For a particular setting of parameters, it is also secure if LPN with constant noise rate is  $2^{O(\sqrt{n})}$  hard.



We are now ready to introduce the (weak) planted XOR assumption.

► **Assumption 34.** For  $m, t : \mathbb{N} \rightarrow \mathbb{N}$  and  $\varepsilon : \mathbb{N} \rightarrow [0, 1/2]$  which are efficiently computable functions of  $n$ , the  $\text{XOR}_{m,t,\varepsilon}$  assumption states that for every probabilistic polynomial-time adversary  $\mathcal{A}$ ,

$$\left| \Pr_{G \leftarrow \mathcal{D}_0(n,m)}[\mathcal{A}(G) = 1] - \Pr_{(G,s) \leftarrow \mathcal{D}_1(n,m,t,\varepsilon)}[\mathcal{A}(G) = 1] \right| = \text{negl}(n)$$

What is referred to as the planted XOR assumption in [9] is simply  $\text{XOR}_{m,O(\log n),0}$ . As one of their major contributions, the authors of [9] give a PRC scheme which is secure if (i) Assumption 34 with  $\varepsilon = 0$ , and  $m = n^{1-\Omega(1)}$ ,  $t = \Theta(\log n)$  is true, and (ii) constant noise rate LPN is hard. In this section, we show that such a scheme can be based on a more expansive set of assumptions. Informally, we will show that if for any  $m = \text{poly}(n)$ ,  $\text{XOR}_{m,\Theta(\log n),O(\log(m)/(m\eta))}$  holds and  $\text{LPN}[\eta]$  holds, then pseudorandom codes exist. For concreteness, one may wish to read this section with the parameter regime  $\eta = 1/\sqrt{n}$  in mind since  $\text{LPN}[1/\sqrt{n}]$  is a well believed assumption and the weakest LPN assumption known to imply public-key cryptography [3].

## 5.2 Evidence $\text{XOR}_{m,t,\varepsilon}$ is a weaker assumption than $\text{XOR}_{m,t,0}$

Before proceeding with our PRC construction, we give two pieces of evidence that  $\text{XOR}_{m,t,\varepsilon}$  is indeed a weaker assumption than  $\text{XOR}_{m,t,0}$ . The first is a reduction which shows that  $\text{XOR}_{m,t,0}$  implies  $\text{XOR}_{m,t,\varepsilon}$ .

► **Theorem 35.** For any  $m, t : \mathbb{N} \rightarrow \mathbb{N}$  and  $\varepsilon : \mathbb{N} \rightarrow [0, 1/2]$  which are efficiently computable functions of  $n$ , if the  $\text{XOR}_{m,t,0}$  assumption holds and  $\text{Ber}(n, \varepsilon)$  is efficiently sampable, then the  $\text{XOR}_{m,t,\varepsilon}$  assumption holds.

**Proof.** See full version. ◀

Our second piece of evidence that  $\text{XOR}_{m,t,\varepsilon}$  is a weaker assumption than  $\text{XOR}_{m,t,0}$  is that  $\text{XOR}_{m,t,\varepsilon}$  seems more robust to known attacks than  $\text{XOR}_{m,t,0}$ . The first version of [9] assumed  $\text{XOR}_{\Theta(n),t,0}$ . However, a subsequent version of [2] gave an attack showing  $\text{XOR}_{\Theta(n),O(\log n),0}$  is not true. The attack (Thm 4.26 of [2]) consists of sampling random  $m/2 \times m$  submatrices of the input matrix  $G$  and then using Gaussian elimination to determine the submatrix contains a sparse subset of rows which xor to zero. The newest version of [9] circumvents this problem by setting  $m = n^{1-\Omega(1)}$ . We note that while  $\text{XOR}_{\Theta(n),O(\log n),0}$  is susceptible to this type of attack,  $\text{XOR}_{\Theta(n),O(\log n),\varepsilon}$  is not for reasonable  $\varepsilon$  (say  $\varepsilon = 1/\sqrt{m}$ ). When attempting this attack against  $\text{XOR}_{\Theta(n),O(\log n),0}$ , we can use Gaussian elimination since we were looking for a zero vector in a  $m/2$  dimensional subspace. When attempting this attack against  $\text{XOR}_{\Theta(n),O(\log n),\varepsilon}$ , we must find a low weight vector in a  $m/2$  dimensional subspace. This problem is the average case version of the problem finding a planted low weight codeword  $v$  in a linear code, a problem which is generally believed to be intractable.

## 5.3 The construction

► **Construction 36** (Weak sparse xor construction,  $\text{weakXOR}[m, t, \varepsilon, \eta]$ ). Let  $m, t, \varepsilon, \eta$  be efficiently computable functions of the security parameter  $n$

- **KeyGen**( $1^n$ ): Sample  $(G, s)$  from  $\mathcal{D}_1(n, m, t, \varepsilon)$ . Output  $(sk = s, pk = G)$ .
- **Encode**( $1^n, G$ ): Sample  $u \leftarrow \text{Ber}(m, \eta)$ ,  $e \leftarrow \text{Ber}(n, \eta)$ . Output  $Gu + e$ .
- **Decode**( $1^n, s, x$ ): If  $s^T x = 0$ , output 1. Otherwise, output  $\perp$ .



## 5.4 Robustness

► **Lemma 37.** *Let  $m = \text{poly}(n)$ ,  $\eta = o(1)$ ,  $t = O(\log n)$ ,  $\varepsilon = O(\log(m)/(\eta m))$ , and  $p$  be any constant in  $[0, 1/2)$ . There exists a polynomial  $p(n)$  such that for Construction 36, for any  $d \leq pn$ , for a  $1 - \text{negl}(n)$  fraction of keys  $(\text{pk}, \text{sk}) \leftarrow \text{KeyGen}(1^n)$ ,*

$$\Pr_{\mathcal{E}}[\text{Decode}(1^n, \text{sk}, \mathcal{E}(x)) = 1 : x \leftarrow \text{Encode}(1^n, \text{pk}, 1)] \geq \frac{1}{2} + \frac{1}{p(n)}$$

where  $\mathcal{E}$  is the  $d$ -hypergeometric channel. the randomness is over the randomness of the encoding algorithm and the errors of  $\mathcal{E}$ .

**Proof.** See full version. ◀

## 5.5 A form of soundness

► **Lemma 38.** *For any  $m, t, \varepsilon, \eta$ ,  $k = \text{poly}(n)$ , for Construction 36, for all key pairs  $(\text{pk}, \text{sk}) \leftarrow \text{KeyGen}(1^n)$ , we have*

$$\Pr_{x \leftarrow \{0,1\}^n}[\text{Decode}(1^n, \text{sk}, x) = \perp] = \frac{1}{2}$$

**Proof.** See full version. ◀

## 5.6 Pseudorandomness

► **Lemma 39.** *For any efficiently computable  $m = \text{poly}(n)$ ,  $t, \varepsilon, \eta$ , if  $\text{XOR}_{m,t,\varepsilon}$  holds, and  $\text{LPN}[\eta]$  holds, then  $\text{weakXOR}[m, t, \varepsilon, \eta]$  is pseudorandom.*

**Proof.** See full version. ◀

► **Remark 40.** Technically, we require something weaker than  $\text{LPN}[\eta]$  to hold for our proof of pseudorandomness. We need only that  $\text{LPN}$  is secure when the distribution of the secret comes from  $\text{Ber}(m, \eta)$  and the error comes from a distribution  $\text{Ber}(n, p)$  for any constant  $p < 1/2$ . However, since the  $\text{LPN}$  assumption is typically stated solely in terms of the error rate and  $\text{LPN}[\eta]$  is sufficient for this construction, we choose to state our results as being based on the (possibly stronger than necessary)  $\text{LPN}[\eta]$  assumption.

## 5.7 Putting it all together

**Proof of Theorem 32.** See full version. ◀

## 6 PRCs for space-bounded adversaries

We now present a zero-bit PRC scheme based on the time-space hardness of the learning parity with noise problem which is robust  $\text{BSC}(p)$  for any constant  $p < 1/2$ . The pseudorandomness of this construction is *unconditional* and not based on cryptographic assumptions.

- **Theorem 41.** *Let  $0 \leq \delta \leq 1/100$  be a constant and  $p$  be a constant in  $[0, 1/2)$ . There exists a constant  $c > 0$  such that  $\text{SSR}[c \log(n), \varepsilon, k, k', \delta]$  is a zero-bit secret-key PRC which*
- *has output length  $O(n)$*
  - *is robust to  $\text{BSC}(p)$*
  - *has key size  $O(n)$*
  - *pseudorandom against probabilistic polynomial time,  $O(n^{1.5-2\delta}/\log^{0.01}(n))$  space adversaries.*

The celebrated work of [25] showed that the learning parity without noise problem requires either a superpolynomial number of samples or  $\Omega(n^2)$  memory. Follow-up work [21] and [14] expanded this work to the cases where the secret is sparse and the case where the samples are noisy. We will begin by reviewing the relevant definitions and results.

► **Definition 42.** *The learning sparse parities problem with density  $\ell$  and error rate  $\varepsilon$  is defined as follows: The secret vector  $s$  is sampled uniformly at random from  $\mathcal{S}_{\ell,n}$ . An algorithm  $\mathcal{A}$  is given samples  $(a, a \cdot s + e)$  where  $a \leftarrow \{0, 1\}^n$ ,  $e \leftarrow \text{Ber}(1/2 - \varepsilon)$ . We say  $\mathcal{A}$  succeeds if it successfully outputs  $s$ .*

► **Lemma 43.** *Let  $q = \text{poly}(n)$  and  $\varepsilon = o(1)$ . The distribution  $a_1, a_1 \cdot s + e_1, \dots, a_q, a_q \cdot s + e_q$  where  $s \leftarrow \mathcal{S}_{\Theta(\log n), n}$  and  $a_i \leftarrow \{0, 1\}^n$ ,  $e_i \leftarrow \text{Ber}(1/2 - \varepsilon)$  for all  $i \in [1, q]$  is next-bit unpredictable for PPT algorithms with  $O(n \log^{0.99}(n)/\varepsilon)$  space.*

## 6.1 Construction

► **Construction 44** (small space resilient construction,  $\text{SSR}[\ell, \varepsilon, k, \delta]$ ). *Let  $\ell, \varepsilon, k'$  be efficiently computable functions of the security parameter  $n$  and  $\delta$  be a constant.*

- **KeyGen**( $1^n$ ): *Sample  $s_1, \dots, s_{k'} \leftarrow \mathcal{S}_{\ell,n}$  and output  $sk = (s_1, \dots, s_{k'})$ .*
- **Encode**( $1^n, (s_1, \dots, s_{k'}), 1$ ): *Sample  $a \leftarrow \{0, 1\}^n$ ,  $e_1, \dots, e_{k'} \leftarrow \text{Ber}(1/2 - \varepsilon)$ , output*

$$a||a \cdot s_1 + e_1|| \dots ||a \cdot s_{k'} + e_{k'}.$$

- **Decode**( $1^n, (s_1, \dots, s_{k'}), x$ ): *Reinterpret  $x \in \{0, 1\}^{n+k'}$  as  $\tilde{a}||\tilde{b}_1|| \dots ||\tilde{b}_{k'}$  where  $\tilde{a} \in \{0, 1\}^n$  and  $\tilde{b}_i \in \{0, 1\}$  for all  $i \in [1, k']$ . If  $\tilde{a}$  is not  $1/(2n^{0.4})$  balanced, output  $\perp$ . Otherwise, let  $w_i$  be one if and only if  $\tilde{a} \cdot s_i = \tilde{b}_i$ . If  $\sum_{i=0}^{k'} w_i \geq k'/2 + n^\delta \sqrt{k'}$  output 1 and otherwise output  $\perp$ .*

## 6.2 Robustness

Say that the decoder receives a string  $x = \tilde{a}||\tilde{b}_1|| \dots ||\tilde{b}_{k'}$ . Intuitively, for every  $i$  such that  $\tilde{a} \cdot s_i = \tilde{b}_i$ , the decoder gains more confidence that  $x$  is a codeword. However, on first inspection, it seems plausible one could flip a just a few of the first  $n$  bits of a codeword (turn  $a$  into  $\tilde{a}$ ) to ensure there would exist very few  $i \in [k']$  such that  $\tilde{a} \cdot s_i = \tilde{b}_i$ . The existence of such an attack could potentially imply that the code of Construction 44 is not particularly robust to errors. We will show that such an attack does not affect robustness due to the sparsity of the  $s_i$ .

► **Definition 45** ([15]). *A family  $Y_1, \dots, Y_{k'}$  of random variables is read- $d$  if there exists a sequence  $X_1, \dots, X_n$  of independent variables, and a sequence  $S_1, \dots, S_{k'}$  of subsets of  $[n]$  such that*

1. *Each  $Y_i$  is a function of  $(X_j : j \in S_i)$ , and*
2. *No element of  $[n]$  appears in more than  $d$  of the  $S_i$ 's.*

► **Lemma 46** ([15]). *Let  $Y_1, \dots, Y_{k'}$  be a family of read- $d$  indicator random variables with  $\Pr[Y_i = 1] = p_i$  and let  $p$  be the average of  $p_1, \dots, p_{k'}$ . Then for any  $\varepsilon > 0$ , the probabilities*

$$\Pr[Y_1 + \dots + Y_{k'} \geq (p + \varepsilon)k'] \quad \text{and} \quad \Pr[Y_1 + \dots + Y_{k'} \leq (p - \varepsilon)k']$$

*are both at most  $e^{-2\varepsilon^2 k'/d}$*

► **Lemma 47.** *Let  $\ell \leq O(\log n)$ ,  $d = \omega(\log n)$ , and  $k' \leq n$ . Consider  $S = \{S_1, \dots, S_{k'}\}$  where each  $S_i$  is drawn uniformly at random from  $\binom{[n]}{\ell}$ . Some element  $t \in [n]$  occurs in  $d$  elements of  $S$  with probability  $\text{negl}(n)$ .*

**Proof.** See full version. ◀

► **Lemma 48.** *Let  $\varepsilon$  be some function of  $n$ ,  $p$  be a constant in  $[0, 1/2)$ ,  $\delta > 0$ , and  $k' = (2n^{2\delta}/\varepsilon)^2$ . There exists a constant  $c > 0$  such that for  $\ell = c \log(n)$ ,  $\text{SSR}[\ell, \varepsilon, k', \delta]$  is robust to  $\text{BSC}(p)$  with probability  $1 - \text{negl}(n)$ .*

**Proof.** See full version. ◀

### 6.3 Soundness

On first inspection, it may seem strange that we output  $\perp$  when trying to decode strings where  $\tilde{a}$  is not balanced. This is to ensure soundness. To see why this exit condition is necessary, consider what happens when the codeword is the string of all zeros. Construction 44 would certainly decode this codeword to 0 regardless of what  $\text{sk}$  is. The requirement that  $\tilde{a}$  eliminates the possibility of such edge cases. We will now show the soundness of our zero bit encryption scheme by showing that any fixed  $x \in \{0, 1\}^{n+k'}$  decodes to  $\perp$  with high probability.

► **Lemma 49.** *Let  $a \in \{0, 1\}^n$  be a  $1/n^{0.4}$ -biased string,  $b \in \{0, 1\}$ ,  $c$  be an arbitrary constant and  $s$  be drawn uniformly at random from  $\mathcal{S}_{c \log(n), n}$ .*

$$\Pr_s[a \cdot s = b] \leq 1/2 + \text{negl}(n)$$

**Proof.** See full version. ◀

► **Lemma 50.** *Let  $0 \leq \delta \leq 1/100$  be constant,  $\epsilon$  be any function of  $n$ ,  $k' \geq n^\delta$  be  $\text{poly}(n)$ , and  $\ell = O(\log n)$ . For any fixed  $x \in \{0, 1\}^{n+k'}$ , in the  $\text{SRR}[\ell, \varepsilon, k', \delta]$  scheme,*

$$\Pr_{\text{sk}}[\text{Decode}(\text{sk}, x) = \perp] \geq 1 - \text{negl}(n).$$

**Proof.** See full version. ◀

### 6.4 Pseudorandomness

We will show pseudorandomness of Construction 44 by first showing that any polynomial number of codewords is next-bit unpredictable for a polynomial time, space-bounded adversary. Lemma 43 shows that sparse parity learning examples  $a||a \cdot s + e$  are next bit unpredictable. In this case,  $a$  is random and one pseudorandom bit is output per freshly sampled  $a$ . However, in Construction 44, the samples are of the form  $a||a \cdot s_1 + e_1|| \dots ||a \cdot s_{k'} + e_{k'}$ . In this case,  $a$  is random and multiple pseudorandom bits are output per freshly sampled  $a$ . Fortunately, next-bit unpredictability of samples of the form  $a||a \cdot s + e$  implies next-bit unpredictability of samples of the form  $a||a \cdot s_1 + e_1|| \dots ||a \cdot s_{k'} + e_{k'}$ .

► **Lemma 51.** *Let  $0 \leq \delta \leq 1/100$  be a constant,  $\varepsilon = 1/\text{poly}(n)$ ,  $\log(\varepsilon) \in \mathbb{Z}$ ,  $q = \text{poly}(n)$ ,  $k' = \text{poly}(n)$ , and  $\ell = \Theta(\log n)$ . Let  $\text{Enc}$  be the encoding function of  $\text{SSR}[\ell, \varepsilon, k', \delta]$ . Consider the distribution induced by  $\text{sk} \leftarrow \text{KeyGen}(1^n)$  and  $X_1, \dots, X_q \leftarrow \text{Enc}(1^n, \text{sk}, 1)$  where  $X_i \in \{0, 1\}^{n+k'}$  for all  $i \in [1, q]$ . No PPT,  $O(n \log^{0.99}(n)/\varepsilon)$  space adversary acts as a next bit predictor for  $X_1, \dots, X_q$ .*

**Proof.** See full version. ◀

► **Lemma 52.** *Let  $0 \leq \delta \leq 1/100$  be a constant,  $\varepsilon = 1/\text{poly}(n)$ ,  $\log(\varepsilon) \in \mathbb{Z}$ ,  $q = \text{poly}(n)$ ,  $k' = \text{poly}(n)$ , and  $\ell = \Theta(\log n)$ . The scheme  $\text{SSR}[\ell, \varepsilon, k', \delta]$  is pseudorandom against  $O(n \log^{0.99}(n)/\varepsilon)$  space,  $\text{poly}(n)$  time adversaries.*

**Proof.** See full version. ◀

## 6.5 Putting it all together

► **Theorem 53.** *Let  $0 \leq \delta \leq 1/100$  be a constant,  $\varepsilon = 1/\text{poly}(n)$ ,  $k' = (2n^{2\delta}/\varepsilon)^2$ , and  $p$  be a constant in  $[0, 1/2)$ . There exists a constant  $c > 0$  such that  $\text{SSR}[c \log(n), \varepsilon, k, k', \delta]$  is a zero-bit secret-key PRC which*

- *has output length  $n + k'$*
- *is robust to  $\text{BSC}(p)$*
- *has key size  $O(k' \cdot \log^2(n))$*
- *pseudorandom against probabilistic polynomial time,  $O(n \log^{0.99}(n)/\varepsilon)$  space adversaries.*

**Proof.** See full version. ◀

Theorem 41 shows that Construction 44 can have quite small key sizes at the expense of being pseudorandom against adversaries with smaller space. Theorem 41 now follows by instantiating the parameter regime we believe to be the most useful.

**Proof of Theorem 41.** See full version. ◀

Therefore, we have shown zero bit PRCs with  $O(n)$  length which are *unconditionally* pseudorandom against  $\text{poly}(n)$  time,  $O(n^{1.5-\delta})$  space (for any constant  $\delta > 0$ ) adversaries. It is natural to ask if this leads to multi-bit PRCs. The construction of multi-bit PRCs with rate  $1/n$  (construction 3 of [9]) also works in the space-bounded setting but has codeword length  $O(kn)$  when encoding  $k$  bits. This would let us build  $k$ -bit PRCs with codeword length  $O(kn)$  which are pseudorandom against PPT,  $O(n^{1.5-\delta})$  space adversaries. However, that construction has the undesirable property that it narrows the gap between the space of the adversary and the space of the encoding algorithm, thereby making the scheme less secure. It would be interesting to build constant rate PRCs which are unconditionally pseudorandom against PPT, space-bounded adversaries.

## 7 Perspectives

Here we review some of the design decisions we have made in our constructions.

In Section 6, we prove robustness to the binary symmetric channel rather than  $p$ -bounded channels (we assume  $p$  is a constant in  $[0, 1/2)$ ). One may ask whether it is possible to prove robustness to all  $p$ -bounded channels rather than just the binary symmetric channel. To show robustness to  $p$ -bounded channels, one could choose to apply a similar type of reduction as given in Lemma 19 by including in the secret key a shift  $z$  and a permutation  $\pi$ . This reduces showing robustness against  $p$ -bounded channels to showing robustness against  $d$ -hypergeometric channels for all  $d \leq pn$ , which is very similar to the binary symmetric channel. Since the robustness probability only goes up as  $p$  goes down in Construction 44, there exists a function  $u(n) = \text{negl}(n)$  such that for all  $d \in [1, pn]$ , Construction 44 is robust to  $\text{BSC}(d/n)$  with probability  $1 - u(n)$ . This implies Construction 44 is robust to any  $d$ -hypergeometric channel for  $d \leq pn$  with probability  $O(n)u(n) = \text{negl}(n)$ . However, such a reduction incurs an additive  $O(n \log n)$  factor in the key size since  $\pi$  is  $O(n \log n)$  bits. In the space-bounded setting, having small keys is particularly important, so we have chosen to focus on the standard setting of the binary symmetric channel, which allows for remarkably small key sizes. However, it should not be hard to formalize the argument for  $p$ -bounded channels. Of course, one could use a pseudorandom function to generate  $\pi$  and avoid the additive  $O(n \log n)$  factor in the key size. We chose not to do this to keep our construction unconditional. Combining the results of Section 6 with other cryptographic objects (such as PRFs) remains an interesting open question.

This work focuses on the theoretical aspects of PRCs but one can also ask if Section 4 and Section 5 are practical for watermarking LLM text. Unfortunately, this seems unlikely. The problem is that if we set the security parameter  $n = 128$  (a reasonable security parameter), the application of the Lemma 19, which allows us to construct a PRC from a scheme where there is only a small advantage in distinguishing codewords from random words, requires us to concatenate many codewords together, which may result in a code with a length of  $\text{poly}(n)$  for some very large polynomial. This is too long to be practical. Fundamentally, Lemma 19 allows us to amplify robustness by concatenating  $t$  codewords of length  $n$  to form a string  $x$  of length  $tn$ . Every  $n$  bit block of  $y = \mathcal{E}(x)$  that is decoded to 1 rather than  $\perp$  gives us more certainty that  $y$  is a corrupted codeword.

There are, however, other ways to amplify our confidence. For example, each codeword of length  $n$  can contain multiple checks. In Construction 36, (for simplicity, consider the  $\epsilon = 0$  regime) we sample  $G$  uniformly at random subject to  $s^T G = 0^m$  and then check if  $y$  is a corrupted codeword by checking if  $s^T y = 0$ . This gives us low confidence that  $y$  is a corrupted codeword, so we apply Lemma 19. However, imagine we had  $s_1, \dots, s_\tau$  and sampled  $G$  uniformly at random subject to the constraints that  $s_i^T G = 0^m$  for all  $i \in [1, \tau]$ . Then to check if  $y$  is a corrupted codeword, we check how many  $i \in [1, \tau]$  there were such that  $s_i^T y = 0$ , and the more there were, the more confidence that we could have that  $y$  is a corrupted codeword. This is the approach advocated by [9].

Similarly, in Section 4, we implant  $\text{poly}(n)$  hyperloops but one use one for decoding (by checking if  $\bigoplus_{j \in S_1} y_j = 0$ ) and then amplify our success probability using Lemma 19. From a theoretical perspective, the polynomial size blowup in the length of the code incurred by Lemma 19 does not matter. However, from a practical perspective, the correct approach would check how many  $i \in [1, t]$  there are such that  $\bigoplus_{j \in S_i} y_j = 0$ . In both cases, adding more structure in the encoding/decoding stages means that the decoder knows with greater certainty if a word is a codeword, without incurring a large blowup in codeword length.

---

## References

- 1 Scott Aaronson. My AI safety lecture for UT effective altruism, 2022. URL: <https://scottaaronson.blog/?p=6823>. 11
- 2 Shweta Agrawal, Sagnik Saha, Nikolaj Ignatieff Schwartzbach, Akhil Vanukuri, and Prashant Nalini Vasudevan.  $k$ -SUM in the sparse regime. Cryptology ePrint Archive, Paper 2023/488, 2023. URL: <https://eprint.iacr.org/2023/488>. 4, 14, 15
- 3 Michael Alekhnovich. More on average case vs approximation complexity. In *FOCS*, pages 298–307, 2003. doi:10.1109/SFCS.2003.1238204. 15
- 4 Benny Applebaum, Boaz Barak, and Avi Wigderson. Public-key cryptography from different assumptions. In *Proceedings of the Forty-Second ACM Symposium on Theory of Computing*, STOC '10, pages 171–180, New York, NY, USA, 2010. Association for Computing Machinery. doi:10.1145/1806689.1806715. 6
- 5 Benny Applebaum, David Cash, Chris Peikert, and Amit Sahai. Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In Shai Halevi, editor, *Advances in Cryptology - CRYPTO 2009*, pages 595–618, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg. doi:10.1007/978-3-642-03356-8\_35. 8
- 6 Andrej Bogdanov, Pravesh Kothari, and Alon Rosen. Public-key encryption, local pseudorandom generators, and the low-degree method. Cryptology ePrint Archive, Paper 2023/1049, 2023. URL: <https://eprint.iacr.org/2023/1049>. 1, 4, 6, 11, 12, 13
- 7 Christian Cachin and Ueli Maurer. Unconditional security against memory-bounded adversaries. In Burton S. Kaliski, editor, *Advances in Cryptology — CRYPTO '97*, pages 292–306, Berlin, Heidelberg, 1997. Springer Berlin Heidelberg. doi:10.1007/BFB0052243. 6

- 8 Herman Chernoff. A Measure of Asymptotic Efficiency for Tests of a Hypothesis Based on the sum of Observations. *The Annals of Mathematical Statistics*, 23(4):493–507, 1952. doi:10.1214/aoms/1177729330. 7
- 9 Miranda Christ and Sam Gunn. Pseudorandom error-correcting codes. In Leonid Reyzin and Douglas Stebila, editors, *Advances in Cryptology – CRYPTO 2024*, pages 325–347, Cham, 2024. Springer Nature Switzerland. doi:10.1007/978-3-031-68391-6\_10. 1, 2, 3, 4, 5, 6, 7, 9, 10, 11, 14, 15, 19, 20
- 10 Miranda Christ, Sam Gunn, and Or Zamir. Undetectable watermarks for language models. In *The Thirty Seventh Annual Conference on Learning Theory*, pages 1125–1139. PMLR, 2024. URL: <https://proceedings.mlr.press/v247/christ24a.html>. 2, 11
- 11 Geoffroy Couteau, Aurélien Dupin, Pierrick Meaux, Mélissa Rossi, and Yann Rotella. *On the Concrete Security of Goldreich’s Pseudorandom Generator: 24th International Conference on the Theory and Application of Cryptology and Information Security, Brisbane, QLD, Australia, December 2–6, 2018, Proceedings, Part II*, pages 96–124. Springer, 2018. doi:10.1007/978-3-030-03329-3\_4. 13
- 12 Yan Zong Ding. Oblivious transfer in the bounded storage model. In Joe Kilian, editor, *Advances in Cryptology — CRYPTO 2001*, pages 155–170, Berlin, Heidelberg, 2001. Springer Berlin Heidelberg. doi:10.1007/3-540-44647-8\_9. 6
- 13 Yevgeniy Dodis, Willy Quach, and Daniel Wichs. Speak much, remember little: Cryptography in the bounded storage model, revisited. In Carmit Hazay and Martijn Stam, editors, *Advances in Cryptology – EUROCRYPT 2023*, pages 86–116, Cham, 2023. Springer Nature Switzerland. doi:10.1007/978-3-031-30545-0\_4. 6
- 14 Sumegha Garg, Pravesh K. Kothari, Pengda Liu, and Ran Raz. Memory-sample lower bounds for learning parity with noise, 2021. arXiv:2107.02320. 5, 6, 17
- 15 Dmitry Gavinsky, Shachar Lovett, Michael E. Saks, and Srikanth Srinivasan. A tail bound for read-k families of functions. *Random Structures & Algorithms*, 47, 2012. URL: <https://api.semanticscholar.org/CorpusID:14447567>. 17
- 16 Surendra Ghentiyala and Venkatesan Guruswami. New constructions of pseudorandom codes, 2024. doi:10.48550/arXiv.2409.07580. 1
- 17 Oded Goldreich. *Candidate One-Way Functions Based on Expander Graphs*, pages 76–87. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011. doi:10.1007/978-3-642-22670-0\_10. 4, 6
- 18 Noah Golowich and Ankur Moitra. Edit distance robust watermarks for language models, 2024. doi:10.48550/arXiv.2406.02633. 2, 6, 11
- 19 Wassily Hoeffding. *Probability Inequalities for sums of Bounded Random Variables*, pages 409–426. Springer New York, New York, NY, 1994. doi:10.1007/978-1-4612-0865-5\_26. 7
- 20 John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. A watermark for large language models. In *International Conference on Machine Learning*, pages 17061–17084. PMLR, 2023. URL: <https://proceedings.mlr.press/v202/kirchenbauer23a.html>. 11
- 21 Gillat Kol, Ran Raz, and Avishay Tal. Time-space hardness of learning sparse parities. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2017, pages 1067–1080, New York, NY, USA, 2017. Association for Computing Machinery. doi:10.1145/3055399.3055430. 5, 6, 17
- 22 Alex Lombardi and Vinod Vaikuntanathan. Minimizing the complexity of goldreich’s pseudorandom generator. Cryptology ePrint Archive, Paper 2017/277, 2017. URL: <https://eprint.iacr.org/2017/277>. 13
- 23 Ueli M. Maurer. Conditionally-perfect secrecy and a provably-secure randomized cipher. *Journal of Cryptology*, 5(1):53–66, 1992. doi:10.1007/BF00191321. 6
- 24 Igor C. Oliveira, Rahul Santhanam, and Roei Tell. Expander-based cryptography meets natural proofs. *computational complexity*, 31(1):4, 2022. doi:10.1007/s00037-022-00220-x. 6

- 25    Ran Raz. Fast learning requires good memory: A time-space lower bound for parity learning. *J. ACM*, 66(1), December 2018. doi:10.1145/3186563. 5, 6, 17
- 26    Umesh V. Vazirani and Vijay V. Vazirani. Trapdoor pseudo-random number generators, with applications to protocol design. In *24th Annual Symposium on Foundations of Computer Science (sfcs 1983)*, pages 23–30, 1983. doi:10.1109/SFCS.1983.78. 6