Lecture 8: Watermarking (Part 2)

Lecturer: Vinod Vaikuntanathan **Date:** March 3, 2026
Scribes: Hannah Gao, Lillian Wang

1. Review

Last class we defined soundness and completeness in watermarking schemes and began describing the Christ-Gunn-Zamir scheme. We begin by defining a watermarking scheme and properties of the scheme and then reviewing the construction of the CGZ scheme.

Remark

Watermarking Scheme:[2] Given a learning model M, a watermarking scheme consists of three polynomial time algorithms:

- $Setup(1^\lambda) \to sk$.
- $Mark_{sk}^{M}(\pi) \to x$ where x represents a piece of watermarked model output.
- $Detect_{sk}(x) \to \{true, false\}$ which outputs true iff x is watermarked.

The potential properties of a watermarking scheme are as follows:[3]

1. **Soundness:** For every $\lambda \in \mathbb{N}$ and sequence x with polynomial length with respect to λ,

$$Pr_{sk \leftarrow Setup(1^\lambda)}[Detect_{sk}(x) = true] \leq \mathsf{negl}(\lambda)$$

   In other words, there is a low probability of false positives.

2. b**-Completeness:** For every $\lambda \in \mathbb{N}$ and polynomial length prompt π,

$$Pr[\text{Detect}_{sk}(x) = \text{false} \& \ Entropy(M, \pi) \geq b(|x|)] \leq \mathsf{negl}(\lambda)$$

   where the probability is over $sk \leftarrow Setup(1^\lambda)$ and $x \leftarrow \text{Mark}_{sk}^{M}(\pi)$. This means that if the entropy of the model is above threshold $b(|x|)$, then the probability of a false negative is negligible.

3. b**-Robustness w.r.t modification channel (**$\mathcal{E}$**:** For all λ and polynomial length prompts π,

$$Pr\left[Detect_{sk}(\mathcal{E}(x)) = 0 \ \& \ Entropy(M, \pi) \geq b\right] \leq \mathsf{negl}(\lambda)$$

   This means that the model should still be able to detect the watermark after the specified modfications, $\mathcal{E}$, to the watermarked text. This is a stronger notion of correctness.

4. **Undetectability:** Given oracle access to model outputs versus watermarked model outputs, the distinguisher cannot distinguish between the two types of outputs. In other words, for every $\lambda \in \mathbb{N}$ and P.P.T. distinguisher D,

$$\left| Pr\left[D^{M, Model^M}(1^\lambda) \to 1 \right] - Pr\left[D^{M, Mark^M_{sk}}(1^\lambda) \to 1 \right] \right| \leq \mathsf{negl}(\lambda)$$

   where $Model^M$ is a text generator given from sampling from the distributions that M outputs.

   We note the motivation behind undetectability is that it ensures there is no noticeable change or drop in quality of watermarked text compared to non-watermarked model generated text.

5. **Distribution Freeness:** For all polynomial size prompts π and for all learning models M,

$$Mark^M_{sk} \equiv Model^M(\pi).$$

   This means that watermarked text and unwatermarked text would have the same distribution given only one sample with the given key sk. Note that this is a weaker notion of distribution freeness.

Christ-Gunn-Zamir (CGZ) 2024 Undetectable Watermarking Scheme

Given a prompt π and prefix $x_1, ..., x_{t-1}$, the learning model M defines a probability p_t such that $x_t \sim Ber(p_t)$, and key k = $(u_1, u_2, ..., u_T)$ where each $u_t \in [0, 1]$, the Mark Algorithm autoregressively generates tokens from the vocabulary T as follows:

$Mark^M_k(\pi)$: for each $i \in [T]$, output

$$x_t = \begin{cases} 1 & \text{if } u_t \leq p_t \\ 0 & \text{otherwise} \end{cases}$$

$Detect_k(x_1, ..., x_T)$:

1. For each $t \in [T]$, calculate the score $s(x_t, u_t) = \begin{cases} ln\left(\frac{1}{u_t}\right) & \text{if } x_t = 1 \\ ln\left(\frac{1}{1-u_t}\right) & \text{if } x_t = 0 \end{cases}$
2. The total score is
$$c(\overline{x}, \overline{u}) = \sum_{t=1}^{T} s(x_t, u_t)$$
   A note on notation, $\overline{x} = x_1 ... x_T$ and $\overline{u} = (u_1, ..., u_T)$.
3. if $c(\overline{x}, \overline{u})$ is large, output "watermarked", otherwise output "not watermarked"

1.1. Overview

In this class, we covered the following topics

- Proved undetectability, completeness, and soundness for the CGZ scheme

- Constructed a Bit Flip Scheme with robustness to a constant fraction of bitflips
- Discussed Pseudorandom Codes which provide the robustness needed for the Bit Flip Scheme.

2. CGZ Scheme

2.1. Watermarking Properties

We now briefly discuss undetectibility, soundness, and completeness for the scheme, in that order.

Claim 2.1. *The CGZ Scheme is undetectable given only one sample. This is also referred to as distribution-freeness.*

Proof. For a given x_t, we know $x_t \sim Bernoulli(p_t)$, meaning $P[x_t = 1] = p_t$. This is equivalent to the probability of outputting 1 iff u_t sampled uniformly from the unit interval is less than p_t. So if the key is not reused, the x_t generated by $Mark_k^M$ will have exactly the same distribution as the distribution of x_i that the original model M would generate. □

Claim 2.2. *For any string $\overline{x}$ generated independently of the secret key, the expected score of $\overline{x}$ in the Detect algorithm is T for random secret key.*

Proof. Given a randomly distributed choice of key $\overline{u}$ and any string $\overline{x}$, the x_t and u_t are completely uncorrelated, so if $x_t = 1$, then $E_u[s(1, u_t)] = E_u[-ln(u_t)]$. If $x_t = 0$, then $E_u[s(0, u_t)] = E_u[-ln(1 - u_t)] = E_u[-ln(u_t)]$ since $u_t \sim Unif([0, 1])$. Thus, we have:

$$E_u[s(x_t, u_t)] = E_u[-ln(u_t)] = -\int_0^1 ln(u)\, du = 1.$$

Taking the expectation over the total score across the T tokens, we get:

$$E_u[c(\overline{x}, \overline{u})] = \sum_{t=1}^{T} 1 = T$$

Thus, as long as we can show that a watermarked string will have a score sufficiently far from T (the expected score of an unwatermarked string), the detect algorithm is unlikely to predict regular text as watermarked. We will show this next. □

This means that as long as the detection threshold for what score is "large enough" compared to the score for a regular string $\overline{x}$, then our Detect algorithm should predict regular strings as unwatermarked with high probability. This reasoning intuitively is similar to soundness which the probability of false negatives.

As an aside, the threshold used in the CGZ scheme is

$$score \geq T + \lambda\sqrt{(T)}$$

For simplicity, the reasoning for this choice of threshold will not be described. The curious reader may be directed to Lemma 2 and Theorem in [2] for more detailed discussion.

Claim 2.3. *The expected score of watermarked text $\overline{x}$ is*

$$T + ln(2) \sum_{t=1}^{T} H(p_t)$$

where H is the Shannon Entropy and p_t is the probability $x_t = 1$

Proof. Assume $\overline{x}$ is watermarked text. Again, let's compute the expected score for token t. This time, x_t and u_t *are* correlated.

$$\begin{aligned} E_u[s(x_t, u_t)] &= \int_0^1 \mathbf{1}[u_t \leq p_t] \cdot (-\ln(u))\, du + \int_0^1 \mathbf{1}[u_t > p_t] \cdot (-\ln(1-u))\, du \\ &= \int_0^{p_t} -\ln(u)\, du + \int_{p_t}^1 -\ln(1-u)\, du \\ &= p_t + (1-p_t) - p_t \cdot ln(p_t) - (1-p_t) \cdot ln(1-p_t) \\ &= 1 + ln(2) \cdot H(p_t). \end{aligned}$$

where $H(p_t) := -p_t log_2(p_t) - (1-p_t) log_2(1-p_t)$ is the binary entropy function of p_t. As before, let us calculate the expectation over the total score across the T tokens:

$$\begin{aligned} E_u[c(\overline{x}, \overline{u})] &= \sum_{t=1}^{T} (E_u[s(x_t, u_t)]) \\ &= \sum_{t=1}^{T} (1 + ln(2) \cdot H(p_t)) \\ &= T + ln(2) \sum_{t=1}^{T} H(p_t) \end{aligned}$$

□

This means that when the entropy of the watermarked text is sufficiently high, the calculated score will be higher than the threshold, ensuring its watermark can be detected correctly with high probability.

This gives us completeness which states that the probability false negatives is low given sufficient entropy.

2.2. Problems and Discussions

We discussed four main problems with this scheme:

2.2.1 Problem 1:

Although we proved that the expected scores for watermarked and unwatermarked text is distinguishable when the model has high entropy, this does not immediately guarantee soundness and completeness. This is because completeness and soundness require the two distributions of text to be distinguishable with high probability rather than in expectation.

We would also need to consider, for example, the variance of the score. With high variance, the model is more prone to false positives or false negatives, both of which could break soundness or completeness.

Additionally, consider the case where the model has high entropy but outputs a fixed string $\overline{z}$ with 1/2 probability. In other words, the model is biased towards outputing $\overline{z}$. Then if the watermarking scheme's detect algorithm considers $\overline{z}$ to be watermarked for a non-negligible number of secret keys, then this breaks soundness as it implies that there exists a string that is classified as watermarked a non-negligible amount of time. On the other hand, if the string is detected as unwatermarked, then if we assume that undetectability still holds, this means that the watermarking scheme must also be biased towards outputing $\overline{z}$. This means that there is non-negligible probability that the watermarked output will be $\overline{z}$ on a prompt, and the detect algorithm incorrectly labels it as unwatermarked, breaking completeness.

Thus, one solution is to consider the empirical entropy of the watermarked outputs and only enforce completeness for outputs with high empirical entropy. In the example above, this ensures that the detect algorithm can label $\overline{z}$ as unwatermarked since the empirical entropy of the unwatermarked text is relatively low.

2.2.2 Problem 2:

The key length in this scheme is too long. Since we need generate a random number in the secret key for each output token, the length of the key scales linearly with the model output.

We can solve this by using a short Pseudorandom Function to generate the secret key $(u_1, ..., u_T)$. The details are as follows: the watermarking scheme first generates a secret key sk for the PRF, F. For each prompt, it generates a seed r which will be used to to generate the secret key: $u_i = F_{sk}(r, i)$.

Note we will refer to the PRF's secret key as the "PRF key" and the watermarking scheme's long secret key as the "secret key".

2.2.3 Problem 3:

The scheme is no longer undetectable when watermarking multiple outputs using the same secret key as the model.

In particular, an adversary can detect if an output is watermarked if they use the same prompt and PRF seed. This is because pseudorandom functions and thus the watermarking scheme is deterministic (given a seed and PRF key), whereas the generative model is non-deterministic in its output. Thus, if the seed remains the same, an adversary can sample twice using the same prompt and predict the model to be the watermarking model if and only if the two outputs are the same, distinguishing the two models correctly with relatively high probability.

In order to avoid repeating PRF seeds, one solution is to use the beginning of the model output as the PRF seed. Namely we first generate $(x_1, ..., x_i)$ using the Model and when the empirical entropy accumulated is high enough, we set $(x_1, ..., x_i)$ to be the PRF seed. For the remaining tokens in $\overline{x}$, we generate the secret key for index $j > i$ as $PRF_{sk}((x_1, ..., x_i), j)$. We require the empirical entropy to accumulate so that the probability of colliding on the seed between different samples is low, decreasing the likelihood of reusing secret keys and thus compromising undetectability.

In the Detection algorithm, the algorithm guesses the PRF seed by iterating through all options of $i \in [T]$ and uses this seed to calculate the secret keys used by the watermarker.

2.2.4 Problem 4:

The described algorithm which uses the prefix of a string as a PRF seed and watermarks the remaining tokens with this seed, is brittle to bit perturbation attacks on the first few bits of the watermarked output. This is because if the the prefix of the text is modified, the seed for the PRF likely unrecoverable, preventing the detect algorithm from generating the correct secret key.

We would like the watermarking scheme to have substring robustness, which means that given any contiguous substring of the watermarked text that is unchanged and has enough entropy, the detect algorithm should be able to detect the watermark. Notably, this means that even if the prefix of a string is modified, the watermarking scheme can still detect the watermark.

This is done by using a PRF seed to generate watermarked text until the accumulated entropy from this section of text is high enough to use as the PRF seed for the next section of output. In the depiction below, this would mean that the first section is generated normally using the Model and used as the PRF seed for section 1 of watermarked text. Once the entropy in [watermarked text 1] is high enough, it can be used as the seed to generate [watermarked 2]. We repeatedly use the output as the new PRF seed until reaching the end of the output.

[plain text][watermarked text 1][watermarked 2] [watermarked 3]...

The detection algorithm guesses the seed by iterating over each contiguous substring of text. Note, this increases the runtime of the detection algorithm by a factor of T, the length of the text.

This way, even with multiple errors, as long as one PRF seed and the output generated with that seed remains, the model can still detect the watermark embedded in the text.

3. A Watermarking Scheme Robust to Bounded Bit Flips

We will now introduce a different watermarking scheme that is robust to a bounded number of bit flips, following the work of Christ and Gunn on Pseudorandom Error-Correcting Codes (PRC) [1]. In the following sections, we first state the properties needed in a function that would allow such a watermarking scheme to be constructed, known as a PRC. We then construct and discuss soundness, completeness, and undetectability for a bit-flip robust watermarking scheme, assuming such a function exists. Finally, investigate the details of constructing a PRC.

3.1. PRC definition

Definition

Pseudorandom Code (PRC): The pair of efficient functions $(G_{pk}, PRCDetect_{sk})$ where $G_{pk} : \{0,1\}^m \to \{0,1\}^T$ mapping from a seed $s \in \{0,1\}^m$ to an output in $\{0,1\}^T$ such that for every security parameter $\lambda \in \mathbb{N}$:

1. Pseudorandomness: the outputs of G should be computationally indistinguishable from truly random strings

$$(pk, G(U_m))_{U_m} \approx_c (pk, U_n)_{U_n}$$

2. Soundness: For any fixed string $x \in \{0,1\}^T$, the probability of a false positive is negligible. In other words:

$$Pr_{sk}[PRCDetect_{sk}(x) = 1] \leq \mathsf{negl}(\lambda).$$

3. Robustness: $PRCDetect_{sk}$ should be able to detect strings close to the range of G_{pk}. In other words, for all $x \in Range(G_{pk}) + HamBall(cT)$, $PRCDetect_{sk}(x) = 1$ with high probability.

Note that $Range(G_{pk})$ is the set of all possible images of G_{pk}, $HamBall(cT)$ is the set of all strings that are at most Hamming distance cT from the string 0^T, and $c < 1$ is some constant.

Before continuing to the watermarking scheme construction, we first motivate the use of PRCs. The goal of the watermarking scheme, in addition to having completeness and soundness, is to be undetectable and robustness to a constant fraction of bit-flips.

If we only want undetectability and not robustness, then we can use an encryption algorithm for G_{pk}. Then we can detect watermarked text if we use a decryption algorithm. However, this provides no robustness as adding noise to an encrypted output would change its decryption result.

If we only want robustness, we can let G_{pk} choose a random codeword from an error correcting code. An **error correcting code** is a collection of points in space where the distance between the points is large (for this use case, the distance can refer to hamming distance between bit strings). Thus, the detect algorithm could still detect if a string with bounded additional noise or error is a codeword. This, of course, does not provide undetectability since the error correcting code is public.

The pseudorandom error-correcting code provides both robustness due to the error-correcting properties as well as undetectability as this code appears pseudorandom.

3.2. Watermarking Scheme Construction

Bit Flip Robust Watermarking Scheme

This scheme uses public key pk and secret key sk for PRC $G_{pk} : \{0,1\}^m \to \{0,1\}^T$ and $PRCDetect_{sk}$ that satisfies the properties in section 3.1. We further assume that $T >> m$.

$Mark^M_{pk}(\pi)$: for each $t \in [T]$: Generate a random string, $r = G_{pk}(s)$, from seed $s \sim U_m$.

- if $p_t \geq \frac{1}{2}$
  - if $r_t = 1$, $x_t \sim Ber(1)$
  - if $r_t = 0$, $x_t \sim Ber(2p_t - 1)$
- if $p_t < \frac{1}{2}$
  - if $r_t = 1$, $x_t \sim Ber(2p_t)$
  - if $r_t = 0$, $x_t \sim Ber(0)$

$Detect_{sk}(x_1, ..., x_T)$: Given $x = x_1, ..., x_T$, check if $x \in Range(G_{pk}) + HamBall(cT)$ for some constant c, using $PRCDetect_{sk}(x_1, ..., x_T)$.

We now prove distribution freeness, completeness, soundness, and robustness of the watermarking scheme.

Claim 3.1. *This scheme satisfies distribution freeness.*

Proof. We first show that if r is uniformly sampled, $x_t \sim Ber(p_t)$ for all $t \in [T]$, meaning the distribution of x exactly matches the distribution of the model generator. When $p_t \geq \frac{1}{2}$,

$$P[x_t = 1] = \frac{1}{2}(1) + \frac{1}{2}(2p_t - 1) = p_t.$$

Similarly, when $p_t < \frac{1}{2}$,

$$P[x_t = 1] = \frac{1}{2}(2p_t) + \frac{1}{2}(0) = p_t.$$

Next, since G_{pk} is pseudorandom, this means that $G_{pk}(U_m) \approx U_T$, ensuring that the distribution of r is computationally indistinguishable from truly random. Thus, the distribution freeness argument holds when r is sampled from G_{pk}. □

Next, we aim to show completeness. This relies on the following lemma:

Lemma 3.2. *If $H_\infty(Model^M(\pi)) \geq cT$, then there are constants $c' > 0, c'' < \frac{1}{2}$ such that for at least $c'T$ indices $t \in [T]$, $\left|p_t - \frac{1}{2}\right| \leq c''$.*

In other words, if the model has lots of entropy (specifically, linear fraction entropy, such as $T/10$), then at least a constant fraction of indices are not fixed. The details of the proof are omitted. However, this can be shown by using the concavity of the entropy function, elaborated on in Lemma 21 of [1].

Claim 3.3. *This scheme gives us completeness.*

Proof. If $\overline{x}$ is watermarked, then

$$P[x_i = r_i] = \begin{cases} \frac{3}{2} - p_i & \text{if } p_i \geq \frac{1}{2} \\ \frac{1}{2} + p_i & \text{if } p_i < \frac{1}{2} \end{cases}$$

which can be easily calculated by the previously defined conditional probabilities for our marking scheme.

This means that if p_i is closer to 1/2, then x_i is more correlated with r_i. As an extreme example, consider the case where $p_i = 1/2$ exactly. Then $P[x_i = r_i] = 1$.

By Lemma 3.2, given enough entropy, p_i is close to $1/2$ for at least a constant fraction of indices, meaning x and r are highly correlated. Thus, the number of indices that r and x disagree on is also at most a constant fraction amount. Thus, if this constant is within the constant fraction of error $PRCDetect_{sk}$ is tolerant to, then the Detect algorithm should be able to successfully detect x as approximately an image of G_{pk} and thus watermarked. □

Soundness of the watermarking scheme follows directly from the soundness of the $PRCDetect_{sk}$.

In terms of robustness, we know outputs of a PRC are still detectable within constant fraction of added noise. This additional noise may come from two sources. The first source is the noise from sampling watermarked outputs x from the watermarked model, and the second source is from an adversary's bitflips.

The amount of noise generated by the model is discussed in lemma 3.2 and claim 3.3 which discusses completeness of the model. Specifically, we know that higher entropy results in less noise (in particular, the case of perfect entropy ensures $p_t = 1/2$ for all indices, which means $x = r$, producing no error). Therefore, we can set the entropy threshold high enough so that the detect algorithm can still detect an additional constant fraction $c'T$ of bitflips to the watermarked text.

3.3. PRC Construction

We construct a PRC $G_{pk}(s, e)$ through the assumption that Learning Parity with Noise is hard. This follows section 2.2 in [1].

Definition 3.1. Learning Parity with Noise (LPN): LPN states that the following two distributions are computationally indistinguishable

$$(A_{n\times m}, A\mathbf{s} + \mathbf{e})_{e\sim Bernoulli(p)} \approx_c (A_{n\times m}, \mathbf{u})_{u\leftarrow_R\{0,1\}^n}$$

where the distributions are over $n \times m$ matrix $A \leftarrow_R \{0,1\}^{n\times m}$, $s \leftarrow_R \{0,1\}^m$, size-n vector e with entries in $Bernoulli(p)$, and $u \leftarrow_R \{0,1\}^n$.

As a corollary, decoding random linear codes is hard because we could otherwise construct a distinguisher for the LPN problem.

We also define the notion of Low-Density Parity-Check matrices.

Definition 3.2. Low-Density Parity-Check matrices (LDPC codes): with hyperparameters (n, m, t, r) are a pair of matrices (D, A) such that

- $D \leftarrow \{0,1\}^{r \times n}$ where every row is $t-$sparse. This means each row of D has at most t non-zero entries.
- $A \leftarrow \{0,1\}^{n \times m}$ is randomly sampled such that $DA = 0 \pmod 2$.

Note, the check $Pr[Dx = 0]$ or $Pr[d \cdot x = 0]$ where d is some row vector of D and x is some string, is called the parity check.

Thus, our construction is:

PRC construction

Setup: Let $pk = A, sk = D$ where $(D \in \{0,1\}^{r \times n}, A \in \{0,1\}^{n \times m})$ is an LDPC$[n, m, t, r]$. The choice of sparsity t is discussed later, and the construction or sampling of (D, A) can be explored further in Lemma 9 of [1].

$G_{pk}(\mathbf{s}, \mathbf{e}) := A\mathbf{s} + \mathbf{e}$

- where s is some randomness and e is some error, and $pk = A$.

$PRCDetect_{sk}(x = As + e)$:

- Output 1 if $Dx = 0 \pmod 2$. In other words, if the product is 0, then the output is likely an output of the PRC.

Claim 3.4. *The construction is a PRC.*

Proof. Pseudorandomness of G_{pk} is immediate from the *LPN* assumption.

For the soundness of the detect algorithm, $D(As+e) = (DA)s + De$. We know $DA = 0$ by definition of LDPC codes. Additionally, if both D and e are sparse (recall that e has a constant fraction of sparsity in the Bit Flip Watermarking scheme), then the probability $De = 0$ is fairly high. On the other hand, if x is any vector determined independently of the PRC, then the probability $Dx = 0$ is very low. Thus, the probability of a false positive is very low.

Robustness is also given by the LPN assumption, since if the error allowed by the LPN assumption is a constant fraction of the output vector length, then adding a constant fraction of additional bitflips to some PRC output x to get x' should maintain a low enough error so that the probability $Dx' = 0$ is high, assuming the margin of error in the detect algorithm is higher compared to the margin of error introduced by the LPN. □

Remark 3.5. The choice of sparsity is very important for the security of the construction.

Let d be a row vector of D. Suppose the error e had $\sqrt{n}$ sparsity, and we could construct d to also have $\sqrt{n}$ sparsity, then the intersection of nonzero elements in d and e is empty with reasonable and thus detectable probability, which can be shown by the Birthday Paradox. Specifically, if we assume both d and e are picked by randomly flipping $\sqrt{n}$ bits (potentially flipping a bit multiple times), then the probability there is no collision of non-empty entries is approximately the probability that each pair of bit flips in d and e respectively occurs in different indices. This is $(1 - 1/n)^{(\sqrt{n})^2} = (1 - 1/n)^n \approx 1/e \approx 0.36788$, so the probability $d \cdot e = 0$ is fairly likely.

However, the watermarking scheme has a constant fraction of noise, meaning we would like to consider the case where e has $cT-$sparsity where $m := T$ is the number of entries in e and $c < 1$ is a constant. Since e is so dense, d would need to be very sparse in order to ensure $d \cdot e = 0$ with high probability. If d has constant error (rather than a constant fraction of error), then the probability that $d \cdot e = 0$ is very high, but this would break the security of the PRC. Namely, the adversary can determine d in polynomial time by trying all possible sparse vectors, and use the reconstructed d to distinguish between PRC outputs and truly random values. On PRC output, with non-negligible likelihood, at least one possible vector d will ensure $d \cdot e = 0$, whereas the probability of there existing such a d is very low.

In the complete construction, we would achieve $O(log\ n)$ sparsity. This would not be susceptible to the previous attack since the number of potential d would be $O(n^{log\ n})$ which is quasipolynomial (or greater than polynomial).

In the last few minutes of class, Vinod described the main idea of the complete construction of D and the PRC. We again redirect readers to Lemma 9 in section 2.2 of [1] for specifics of the complete construction. The sketch is as follows: we allow d to have $O(log\ n)$ nonzero bits where each bit is biased by a constant amount (i.e. drawn from a Bernoulli distribution where the probability is a constant away from $\frac{1}{2}$).

When we xor or add (mod 2) the bits up, the total bias lessens. However, the parity check biaswould be

$$\frac{1}{2} - const^{O(log\ n)} = \frac{1}{2} - \frac{1}{O(poly(n))}$$

which happens to be enough bias to be detectable if we estimate the parity check probability by examining the r rows in D and counting the number of rows that cause $d \cdot e = 0$.

Remark 3.6. The above construction only provides quasipolynomial security. There is a construction with better security following from a subexponential LPN assumption. Namely, if A only has $log\ log(n)$ columns, this reduces to the standard LPN assumption except with $log\ log(n)$ size secret keys. This is susceptible to a subexponential attack since there are only $2^{O(log\ log(n))}$ secret keys which is subexponentially many.

However, according to the lecturer, constructing a PRC with both better security and more reasonable assumptions is still very much an active area of research.

Remark 3.7. The constructions in this lecture are still brittle to paraphrasing and translation, and it's unclear if it's even possible to construct a watermarking scheme that is robust to these channels.

Another final remark is that allegedly, a version of this scheme is used by SynthID, a watermarking technology made by Google.

References

[1] Miranda Christ and Sam Gunn. Pseudorandom error-correcting codes, 2024. URL https://arxiv.org/abs/2402.09370.

[2] Miranda Christ, Sam Gunn, and Or Zamir. Undetectable watermarks for language models, 2023. URL https://arxiv.org/abs/2306.09194.

[3] Vinod Vaikuntanathan. Lecture 7: Watermarking for generative models handwritten notes. MIT 6.S976/18.S996 Cryptography and Machine Learning: Foundations and Frontiers, March 2026. URL https://mlcrypto.mit.edu/course/slides/lec7-handwritten.pdf.