Separating Pseudorandom Codes from Local Oracles

Nico Döttling[1⋆], Anne Müller[1,2], and Mahesh Sreekumar Rajasree[1⋆]

[1] CISPA Helmholtz Center for Information Security, Saarbrücken, Germany
{doettling,anne.mueller}@cispa.de, srmahesh1994@gmail.com
[2] Graduate School of Computer Science, Saarland University, Germany

Abstract. Pseudorandom codes (PRCs) are error-correcting codes with the distinguishing feature that their codewords are computationally indistinguishable from random strings. Introduced by Christ and Gunn (CRYPTO 2024), PRCs have found applications in areas such as AI watermarking, where both robustness and pseudorandomness are essential. All known constructions of PRCs rely on coding-theoretic hardness assumptions. In this work, we study how inherent the use of coding-theoretic hardness is in the construction of pseudorandom codes.

We show that there is no black-box construction of PRCs with binary alphabets capable of decoding from a constant fraction of Bernoulli noise from a class of oracles we call *local* oracles. The class of local oracles includes random oracles and trapdoor permutation oracles, and can be interpreted as a meaningful notion of oracles that are not resilient against noise. Our separation result is cast in the Impagliazzo-Rudich framework and crucially relies on the Bonami-Beckner hypercontractivity theorem on the Boolean hypercube.

As a complementary result, we show that PRCs with large alphabets that can tolerate high error rates can indeed be constructed in a black-box manner from one-way functions.

Keywords. pseudorandom codes, black-box separation

1 Introduction

Both coding theory and cryptography are deeply rooted in information theory, but their objectives and principles differ in fundamental ways. Coding theory primarily addresses the challenge of reliable communication and storage of information in the presence of noise. In the problem of error correction, the goal is to encode data so that even if a portion of it is corrupted during transmission or storage, the original message can still be recovered accurately. This is achieved by introducing *structured redundancy* into the message, which allows errors to be detected and corrected by the receiver.

The most basic task of cryptography, on the other hand, is to ensure secrecy of data. A central concept in cryptography is pseudorandomness: the idea that one can *efficiently* generate strings that are computationally indistinguishable from truly random ones. Hence pseudorandomness hides *any discernible structure*. This property underlies many cryptographic primitives, such as pseudorandom generators (PRGs), which expand short random seeds into long pseudorandom outputs. PRGs are a cornerstone of modern cryptography and are known to be constructible from the minimal assumption of one-way functions (OWFs) [Yao82, BM84, HILL99].

Pseudorandom Codes (PRCs) [CG24] are an exciting new notion that unites these two concepts which are seemingly at odds. In a nutshell, a pseudorandom code is an error correcting code in the sense that noisy codewords can be efficiently decoded from a bounded number of Hamming

⋆ Funded by the European Union (ERC, LACONIC, 101041207). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them.

errors *given a secret key*. At the same time, codewords are indistinguishable from uniformly random strings to any outsider who does not possess the secret key.

This notion was first formalized by Christ and Gunn [CG24], with the purpose of studying watermarking of AI models [ZDR19, KJGR21, Aar22, KGW+23, FGJ+25, ZALW24, KTHL24]. Their work introduces a model which allows to embed data (e.g., a watermark) into machine learning models in such a way that the watermark is resilient to small perturbations (errors), yet concealed to anyone without the appropriate secret.

Constructing PRCs. Building PRCs turns out to be a highly non-trivial task [CG24, GM24, GG24, AAC+24]. The naive approach of masking codewords of a standard error correcting code with a pseudorandom function is flawed; if encoder and decoder are not synchronized (i.e. do not share a common state) this does not yield a correct scheme. The basic premise to make this approach work is to let the encoder include an encryption of such a state into the codeword. But now this state must be protected against noise, hence we also need to encode it! It turns out that this is a hen-and-egg problem: In order to build a pseudorandom code in this way we need to start with a pseudorandom random code!

To break out of this loop, current constructions of PRCs rely on strong and sometimes non-standard cryptographic assumptions *strongly related to coding theory*, such as the McEliece problem [McE78], the Learning Parity with Noise (LPN) assumption [Kea93, BFKL93], the planted XOR or planted hyperloop assumptions [BKR23] or the existence of Local Weak Pseudorandom Functions (LWPRFs) [GM24], etc. These assumptions go significantly beyond the foundational concept of one-way functions and are typically viewed as strong assumptions in the hierarchy of cryptographic assumptions, and they are *explicitly linked to coding theory*. In other words, each of these assumptions more or less explicitly conjectures that some basic construction is, in essence, a weak form of a pseudorandom code.

Given this landscape, a fundamental and natural question arises:

Can we construct pseudorandom codes from (minimal) cryptographic standard assumptions that do not relate to coding theory?

The question of identifying the minimal assumptions required for constructing cryptographic primitives lies at the heart of cryptographic theory. Understanding the weakest possible assumption not only sheds light on the intrinsic complexity of a primitive but also helps delineate the boundaries of what is cryptographically feasible. A landmark result in this direction is the seminal work of Impagliazzo and Rudich [IR89], which showed that public-key encryption schemes cannot be constructed from one-way functions using black-box techniques.

In a black-box construction of a cryptographic primitive P from an assumption A, the primitive A is accessed solely as an oracle. Its internal structure is never exploited. Concretely, Impagliazzo and Rudich [IR89] model one-way functions as random oracles, i.e. functions with a uniformly random function table over some finite domain. This framework is attractive for its generality and modularity, often leading to simpler and more broadly applicable constructions. A *black-box separation* shows that no such construction is possible: that is, if a black-box construction of P from A did exist, then there always exists an unbounded adversary breaking the security of P which is efficient in the sense that it only makes a polynomial number of queries to A. The Impagliazzo–Rudich [IR89] framework laid the foundation for a rich body of work establishing black-box separations between various cryptographic primitives, illuminating the landscape of cryptographic assumptions and revealing the limitations of black-box constructions in building advanced cryptographic functionalities [Rud88, Rud91, Sim98, GKM+00, GMR01, HR04, BMG09, BKSY11, MM11, MP12, Mat14, FR23].

1.1 Our Results

We consider the most basic setting of symmetric key pseudorandom codes against a Bernoulli channel. In a symmetric key pseudorandom code, both the encoder Encode and the decoder Decode use the same secret key sk. In a Bernoulli channel with *bias-parameter* $\rho \in [0,1]$, each transmitted bit is independently flipped with probability $\frac{1}{2} - \frac{1}{2}\rho$. More generally, we say that an error $\mathbf{e} \in \{0,1\}^n$ follows the Bernoulli distribution [3] Ber_ρ^n, if each bit e_i of $\mathbf{e}$ is independently 1 with probability $\frac{1}{2} - \frac{1}{2}\rho$. Before we describe our technical main result, we will first discuss a consequence of this result.

Theorem 1 (Informal). *For any constant ρ, there is no black-box construction of symmetric key pseudorandom codes against the Bernoulli channel with parameter ρ from random oracles or even trapdoor permutations.*

That is, we can black-box-separate symmetric key pseudorandom codes from the most common *generic* assumptions used in both symmetric and public key cryptography, namely one-way functions and trapdoor permutations.

Our actual separation encompasses an even larger class of oracles, namely any oracle we call *local*. Consider the following experiment. An adversary $\mathcal{A}$ makes queries q_i to an oracle $\mathcal{O}$ and obtains responses y_i. Let Z denote the set of query-response pairs (q_i, y_i). Assume now that we want to simulate the output of $\mathcal{O}$ on a query q without actually querying $\mathcal{O}$. Under what conditions on q can we simulate the output consistently with the set Z without actually knowing Z?

In the case of random oracles this condition is simple: As long as q is distinct from all q_i in the set Z, we can just choose a fresh random output y_q to simulate $\mathcal{O}$, as in this case $\mathcal{O}(q)$ itself is uniformly random and independent of Z. That is, our simulation of the random oracle is faithful as long as q is not an *intersection query*. Further observe that if we are *given back* oracle access to the random oracle $\mathcal{O}$ after simulating the response for q, we can faithfully simulate any future query q', irrespective of whether q' is an intersection query or not: If $q' = q$ we answer with y_q, otherwise we query $\mathcal{O}$ on q'.

In general, we say that an oracle $\mathcal{O}$ is *local*, if for any (adversarially generated) set Z of query-response pairs of polynomial size, there exists a set Q_{rel} of *related queries* such that $|Q_{rel}|$ is of size polynomial in $|Z|$. We further require that for any $q \notin Q_{rel}$ there is a way to simulate the response of $\mathcal{O}(q)$ without knowledge of Z and without actually querying $\mathcal{O}$, such that the simulated response is consistent with Z. Finally, we require that all future queries q' can be simulated consistently with respect to both Z and q, given oracle access to $\mathcal{O}$ (but without knowledge of Z).

While by the above discussion it is immediately clear that random oracles are local, in Section 4 we show that the trapdoor permutation oracle is also local.

Our main result is a separation of pseudorandom codes from any local oracle.

Theorem 2 (Informal). *For any constant ρ, there is no black-box construction of symmetric key pseudorandom codes against the Bernoulli channel with parameter ρ from a local oracle $\mathcal{O}$.*

By the above discussion Theorem 1 is immediately implied by Theorem 2.

Pseudorandom Code Oracles are not local. It is instructive to realize *why* oracles $\mathcal{O}_{\mathsf{PRC}}$ that implement ideal pseudorandom codes are not local. Say that $\mathcal{O}_{\mathsf{PRC}}$ can decode from $r = \alpha n$ Hamming errors. If we make an encoding query m to $\mathcal{O}_{\mathsf{PRC}}$ and obtain a codeword $\mathbf{c}$, then by the error correction property of $\mathcal{O}_{\mathsf{PRC}}$ any decoding query $\mathbf{c} + \mathbf{e}$ must return m as long as the Hamming weight of $\mathbf{e}$

[3] Here we use the notation commonly used in the Boolean functions literature [O'D14], where the Bernoulli distribution is parametrized in terms of the bias rather than the bit-flip probability.

is smaller than r. However, there are $2^{H(\alpha)n}$ such errors $\mathbf{e}$ [4], which means there is an exponential number of related queries $\mathbf{c} + \mathbf{e}$!

Consequently, one way to interpret our result is that in order to construct a pseudorandom code in a black-box way, we need to start with a primitive that is *non-local* and therefore already exhibits strong pseudorandom code-like properties. This is consistent with our understanding of current constructions of pseudorandom codes, which essentially bootstrap pseudorandom codes with weak correctness and security properties into pseudorandom codes with strong correctness and security properties [CG24, GM24, GG24, AAC+24].

A complementary result. As a complementary result, we show that symmetric key pseudorandom codes over large alphabets, i.e. alphabets of size 2^λ *can* be constructed from minimal cryptographic assumptions.

Theorem 3. *Assuming one-way functions there exists a symmetric-key pseudorandom code construction over an alphabet of size 2^λ with code parameters approaching the Singleton bound.*

This demonstrates that the assumption of working over a small field is in fact critical for our separation.

2 Technical Overview

We will now provide a high level outline of the main ideas of our oracle separation. For concreteness assume that $\mathcal{O}$ is a random oracle and that we are given a candidate symmetric key PRC construction $\mathsf{PRC}^{\mathcal{O}} = (\mathsf{KeyGen}, \mathsf{Encode}^{\mathcal{O}}, \mathsf{Decode}^{\mathcal{O}})$ relative to $\mathcal{O}$. Without loss of generality, we assume that the key-generator KeyGen does not make any oracle queries; since we are in the symmetric key setting any $\mathcal{O}$-queries made by KeyGen can be deferred to both $\mathsf{Encode}^{\mathcal{O}}$ and $\mathsf{Decode}^{\mathcal{O}}$. Hence assume that KeyGen just outputs a uniformly random string $\mathsf{sk} \in \{0,1\}^\lambda$. $\mathsf{Encode}^{\mathcal{O}}$ and $\mathsf{Decode}^{\mathcal{O}}$ are efficient in the sense that they only make a $\mathsf{poly}(\lambda)$ number of $\mathcal{O}$-queries.

Security of such a construction holds with respect to a distinguisher $\mathcal{D}^{\mathcal{O},\mathcal{F}}$ which has access to the oracle $\mathcal{O}$ and an additional oracle $\mathcal{F}$. $\mathcal{D}$ is allowed to make a polynomial number of queries to its oracles, but is otherwise computationally unbounded, so as to model that $\mathcal{O}$ *is the only source of computational hardness* [IR89].

The goal of the distinguisher is to distinguish the following two cases:

- The oracle $\mathcal{F}$ implements $\mathsf{Encode}^{\mathcal{O}}(sk, \cdot)$ for $\mathsf{sk} \leftarrow \mathsf{KeyGen}$. I.e. given a query m it outputs $\mathbf{c} = \mathsf{Encode}^{\mathcal{O}}(sk, m) \in \{0,1\}^n$, choosing fresh random coins for Encode at each query.
- The oracle $\mathcal{F}$ outputs a fresh uniformly random string $\mathbf{u} \in \{0,1\}^n$ for each query. Note that $\mathcal{F}$ is not a random oracle as querying $\mathcal{F}$ twice on the same input will lead to two independent uniformly random outputs.

We say a pseudorandom code $\mathsf{PRC}^{\mathcal{O}}$ is secure (or satisfies the pseudorandomness property), if every distinguisher $\mathcal{D}$ that makes at most $\mathsf{poly}(\lambda)$ queries has at most negligible advantage in this experiment.

We will now outline why no such candidate construction $\mathsf{PRC}^{\mathcal{O}} = (\mathsf{KeyGen}, \mathsf{Encode}^{\mathcal{O}}, \mathsf{Decode}^{\mathcal{O}})$ can meet this security notion. That is, for every such candidate construction $\mathsf{PRC}^{\mathcal{O}}$ we will construct an efficient (i.e. poly-query) distinguisher which breaks the pseudorandomness property of $\mathsf{PRC}^{\mathcal{O}}$ with $1/\mathsf{poly}(\lambda)$ advantage.

[4] Here H is the binary entropy function.

Warmup: $\mathsf{Decode}^{\mathcal{O}}$ *makes no* $\mathcal{O}$*-queries.* As a starting point, we observe that if $\mathsf{PRC}^{\mathcal{O}} = (\mathsf{KeyGen}, \mathsf{Encode}^{\mathcal{O}}, \mathsf{Decode}^{\mathcal{O}})$ is such that Decode does not make any $\mathcal{O}$ queries, then unsurprisingly $\mathsf{PRC}^{\mathcal{O}}$ is insecure. Recall that $|\mathsf{sk}| = \lambda$. Our distinguisher $\mathcal{D}$ queries its oracle $\mathcal{F}$ on (say) 2λ uniformly random bits $b_1, \ldots, b_{2\lambda} \in \{0,1\}$ and obtains the outputs $\mathbf{x}_1, \ldots, \mathbf{x}_{2\lambda}$. First assume that $\mathbf{x}_1, \ldots, \mathbf{x}_{2\lambda}$ are distributed uniformly. Fix any $\mathsf{sk}^* \in \{0,1\}^\lambda$. Since the $b_i \in \{0,1\}$ are chosen uniformly and are independent of the $\mathbf{x}_i$, the probability that it holds for all i that $\mathsf{Decode}(\mathsf{sk}^*, \mathbf{x}_i) = b_i$ is at most $2^{-2\lambda}$. Hence, a union-bound shows that the probability that there exists and $\mathsf{sk} \in \{0,1\}^\lambda$ with this property is at most $2^{-\lambda}$.

On the other hand, if the $\mathbf{x}_i$ are codewords of $\mathsf{PRC}^{\mathcal{O}}$, i.e. $\mathbf{x}_i = \mathsf{Encode}^{\mathcal{O}}(\mathsf{sk}, b_i)$, then it holds by the correctness of $\mathsf{PRC}^{\mathcal{O}}$ that $\mathsf{Decode}(\mathsf{sk}, \mathbf{x}_i) = b_i$. Consequently, in this case we *know* that there exists an sk such that $\mathsf{Decode}(\mathsf{sk}^*, \mathbf{x}_i) = b_i$ for all $i \in [2\lambda]$.

This observation leads to a simple distinguishing strategy: Our distinguisher $\mathcal{D}$ just brute-forces over all $\mathsf{sk} \in \{0,1\}^\lambda$; if it finds an sk which decodes all $\mathbf{x}_i$ correctly it guesses 1 (with the meaning that the oracle $\mathcal{F}$ implements $\mathsf{Encode}^{\mathcal{O}}(\mathsf{sk}, \cdot)$), otherwise 0.

Decoders with correctness errors. The argument in the last paragraph assumed that if we use the correct key sk, $\mathsf{Decode}^{\mathcal{O}}(\mathsf{sk}, \mathbf{x})$ will always output the correct message. The attack does not work if Decode has a correctness error. However, even if the (honest) decoder has a correctness error, the distinguisher can still measure whether a decoder works *most of the time*. Specifically, the distinguisher can count the number of indices $i \in [2\lambda]$ for which $\mathsf{Decode}(\mathsf{sk}^*, \mathbf{x}_i) = b_i$ holds. Call this number Z.

If the $\mathbf{x}_i$ are chosen uniformly, we expect Z to be close to $2\lambda/2 = \lambda$. This can be made precise using a Chernoff bound. On the other hand, if the $\mathbf{x}_i$ are codewords, we would expect Z to be closer to 2λ *often enough*. We can make this concrete using a Markov bound.

Observe that at this point none of our distinguishers so far relies on the fact that Decode can correct errors. This will become crucial in the following paragraphs.

$\mathsf{Decode}^{\mathcal{O}}$ *makes one single* $\mathcal{O}$*-query.* Now consider the case that $\mathsf{Decode}^{\mathcal{O}}$ makes one single $\mathcal{O}$-query. In this case our previous argument no longer works, the distinguisher cannot brute force over all 2^λ many $\mathsf{sk} \in \{0,1\}^\lambda$ and test whether decoding works, as this would quickly exhaust its $\mathsf{poly}(\lambda)$ budget of $\mathcal{O}$-queries.

Hence, let's have a closer look at how $\mathsf{Decode}^{\mathcal{O}}$ may use its sole $\mathcal{O}$-query. The decoder gets as inputs a secret key sk and a word $\mathbf{x} \in \{0,1\}^n$ and formulates a query q for $\mathcal{O}$. Hence, we can describe this process by a function [5] $\mathsf{Quer}_{\mathsf{sk}} : \{0,1\}^n \to Q$, where Q is the input domain of $\mathcal{O}$, the *query space*. Recall that we are analyzing a decoder for the Bernoulli channel. So what happens if we obtain $\mathbf{x}$ by adding Bernoulli noise to a codeword $\mathbf{c}$? We have that $\mathsf{Decode}^{\mathcal{O}}(\mathsf{sk}, \mathbf{c} + \mathbf{e})$ queries $\mathcal{O}$ on $\mathsf{Quer}_{\mathsf{sk}}(\mathbf{c} + \mathbf{e})$ and obtains $\mathcal{O}(\mathsf{Quer}_{\mathsf{sk}}(\mathbf{c} + \mathbf{e}))$. Assume for now that $\mathcal{O}(\mathsf{Quer}_{\mathsf{sk}}(\mathbf{c} + \mathbf{e}))$ actually has an influence on the output of the decoder, if not we have essentially and oracle-free decoder and the attack discussed above applies.

Observe that the random oracle $\mathcal{O}$ itself is not noise-robust; for any error e the oracle outputs $\mathcal{O}(q)$ and $\mathcal{O}(q + e)$ are independently uniform. Now, as the Bernoulli error $\mathbf{e}$ has high entropy, we claim that in order to achieve correctness the query-function $\mathsf{Quer}_{\mathsf{sk}}$ must be noise robust! Otherwise, if $\mathsf{Quer}_{\mathsf{sk}}$ somehow allowed the error $\mathbf{e}$ to *pass through*, the response $\mathcal{O}(\mathsf{Quer}_{\mathsf{sk}}(\mathbf{c} + \mathbf{e}))$ would be completely unpredictable from the view of the encoder. In other words, as it would be very unlikely that the decoder ever queried $\mathcal{O}$ on $\mathsf{Quer}_{\mathsf{sk}}(\mathbf{c} + \mathbf{e})$, the codeword $\mathbf{c}$ is independent of this value and the decoder does not actually need to query $\mathcal{O}$, but could just locally simulate it! Hence, if this happens too frequently, the decoder does not actually need $\mathcal{O}$ and we are once again in the oracle-free decoder setting discussed above.

[5] Assume for now that this process is deterministic. However, our techniques readily generalize to the case where $\mathsf{Quer}_{\mathsf{sk}}(\mathbf{x})$ takes additional random coins.

Hence, we conclude that the function $\mathsf{Quer}_{\mathsf{sk}}$ must be noise robust *on codewords*! How can we use this fact to construct an attack against the one-query decoder?

Strategy Our basic high-strategy will leverage the ideas sketched in the last paragraph to construct an *equivalent decoder* that does not make use of the oracle! The discussion in the next paragraphs of this outline will be somewhat technical, so this is a good point to reflect what goals we want to achieve in each step.

- We will first establish that the only query functions $\mathsf{Quer}_{\mathsf{sk}}$ that are robust on noisy codewords are essentially *almost constant* functions, i.e. functions that produce only a small number of *heavy* queries. This argument will rely on both sophisticated Fourier-analytic properties of boolean functions (refer Section 5.1 and Section 5.2) and the pseudorandomness of $\mathsf{PRC}^{\mathcal{O}}$ (refer Lemma 4).
- Once we have established this, we can argue that *heavy queries* computed by $\mathsf{Quer}_{\mathsf{sk}}$ can be learned from the function $\mathsf{Quer}_{\mathsf{sk}}$ alone by an *offline learner*. This allows us to include query-response pairs for such heavy queries into an augmented secret key. That is, we can effectively *compile out* the single oracle query made by $\mathsf{Decode}^{\mathcal{O}}$ and the previous attack against oracle-free decoders applies. See Lemma 5 in the main section.
- If the decoder $\mathsf{Decode}^{\mathcal{O}}$ makes more $\mathcal{O}$-queries, we can compile out the oracle queries one at a time. Some care is needed to ensure that successive queries to $\mathcal{O}$ are answered consistently with the query that has been compiled out. See Theorem 9 in the main section.

Noise-robustness of Boolean Functions We will now bound the amount of noise robustness we can expect from a query function. Towards this goal, we will make a small detour through the realm of Fourier analysis of boolean functions. We will first introduce some terminology. Fix *any* function [6] $\mathsf{Quer} : \{0,1\}^n \to Q$. We will define two quantities with respect to Quer.

- For a query $q \in Q$, we define the *global weight* of q by $\mathsf{w}(q) = \Pr_{\mathbf{u}}[\mathsf{Quer}(\mathbf{u}) = q]$, i.e. the probability that Quer outputs q when queried on a uniformly random $\mathbf{u} \leftarrow \{0,1\}^n$.
- For an $\mathbf{x} \in \{0,1\}^n$ and a query $q \in Q$ we define the *local weight* of q by $\ell_{\mathbf{x}}(q) = \Pr_{\mathbf{e}}[\mathsf{Quer}(\mathbf{x}+\mathbf{e}) = q]$, where $\mathbf{e} \leftarrow \mathsf{Ber}_{\rho}^{n}$ follows a Bernoulli distribution. Intuitively, $\ell_{\mathbf{x}}(q)$ measures how heavy q is around $\mathbf{x}$.

How are the global and the local weight of a query q related? Intuitively, if we have a query q which has *small* global weight ϵ but *large* local weight δ around a word $\mathbf{x}$, then we can think of the function Quer as exhibiting *error correcting* properties around $\mathbf{x}$; we can think of $\mathbf{x}$ as an error-correcting encoding of q which can be decoded even under noise by evaluating $\mathsf{Quer}(\mathbf{x})$.

How frequent are such *exceptional* words $\mathbf{x}$? Basic coding bounds such as the Hamming bound (aka sphere-packing bound) suggest that such $\mathbf{x}$ are extremely rare, as each of them must come with a Hamming ball around it that occupies a considerable portion of the boolean hypercube $\{0,1\}^n$.

To make this intuition concrete, we will rely on the Bonami-Beckner hypercontractivity theorem [Bon70, Bec75], a celebrated result in Fourier-analysis of Boolean functions. In broad terms the result enables us to bound higher order moments of a noisy version of a Boolean function in terms of lower order moments of the function itself. In our concrete case, the hypercontractivity theorem allows us to bound the second moment of the local weight $\ell_{\mathbf{u}}(q)$ (for a uniformly random $\mathbf{u} \in \{0,1\}^n$) in terms of the global weight $\mathsf{w}(q)$. Specifically we get for every $q \in Q$ that

$$\mathop{\mathsf{E}}_{\mathbf{u} \leftarrow \{0,1\}^n}[\ell_{\mathbf{u}}(q)^2] \leq \mathsf{w}(q)^{1+c}$$

[6] In this paragraph we will ignore that there is a secret key involved.

for some constant $c \in (0,1)$ that only depends on the Bernoulli noise-rate ρ, but is independent of the query function Quer!

Now, we call an $\mathbf{x} \in \{0,1\}^n$ (ϵ,δ)-exceptional (just exceptional for short) *if there exists* a $q \in Q$ with $\mathsf{w}(q) < \epsilon$ and $\ell_{\mathbf{x}}(q) > \delta$. We can now bound the probability that a uniformly random $\mathbf{u} \leftarrow \{0,1\}^n$ is exceptional by

$$\begin{aligned}\Pr[\mathbf{u} \text{ exceptional }] &= \Pr[\exists q \in Q \text{ with } \mathsf{w}(q) < \epsilon \text{ s.t. } \ell_{\mathbf{u}}(q) > \delta] \\ &\leq \sum_{\substack{q \in Q \\ \mathsf{w}(q)<\epsilon}} \Pr[\ell_{\mathbf{u}}(q) > \delta] \\ &= \sum_{\substack{q \in Q \\ \mathsf{w}(q)<\epsilon}} \Pr[\ell_{\mathbf{u}}(q)^2 > \delta^2] \\ &\leq \sum_{\substack{q \in Q \\ \mathsf{w}(q)<\epsilon}} \frac{\mathsf{w}(q)^{1+c}}{\delta^2},\end{aligned}$$

where we have just used the union bound and the Markov inequality. Using the fact that $\sum_{q\in Q} \mathsf{w}(q) = \sum_{q\in Q} \Pr[\mathsf{Quer}(\mathbf{u}) = q] = 1$ we can use a basic norm-interpolation inequality to conclude that

$$\Pr[\mathbf{u} \text{ exceptional}] \leq \epsilon^c/\delta^2.$$

For the details, refer to Section 5.1.

Exceptionality of codewords. So far we have established a bound that a uniformly random string $\mathbf{u} \leftarrow \{0,1\}^n$ is exceptional. However, returning to our prior discussion, what we are actually interested in is the noise-robustness of the query function $\mathsf{Quer}_{\mathsf{sk}}$ when we evaluate it on a noisy codeword $\mathbf{c} + \mathbf{e}$. As by the discussion in the last paragraph, this immediately relates to the exceptionality of $\mathbf{c}$.

We also gained the insight that exceptional words are rare under the uniform measure on $\{0,1\}^n$.

We will now translate these insights to codewords of $\mathsf{PRC}^{\mathcal{O}}$ by relying on the pseudorandomness of $\mathsf{PRC}^{\mathcal{O}}$. Specifically, we can bound the probability that a codeword $\mathbf{c}$ is exceptional with respect to the query function $\mathsf{Quer}_{\mathsf{sk}}$ (using the correct secret key sk) by

$$\Pr[\mathbf{c} \text{ exceptional}] \leq 3\epsilon^c/\delta^2.$$

In the following, let $p^* = \epsilon^c/\delta^2$. Assume towards contradiction that the probability $\Pr[\mathbf{c} \text{ exceptional}] > 3p^*$. Recall that it holds $\Pr[\mathbf{u} \text{ exceptional}] \leq p^*$. Note that if we are given a key sk we can determine whether a word $\mathbf{x}$ is exceptional *without using the oracle* $\mathcal{O}$, i.e. just check whether there is a $q \in Q$ with $\mathsf{w}(q) < \epsilon$ and $\ell_{\mathbf{x}}(q) > \delta$. The distinguisher $\mathcal{D}$ queries its challenge oracle $N \approx (|\mathsf{sk}| + \lambda)/p^* = \mathsf{poly}(\lambda)$ times, obtaining words $\mathbf{x}_1, \ldots, \mathbf{x}_N$. Now, for each *candidate key* $\mathsf{sk} \in \{0,1\}^\lambda$ it determines the number of $\mathbf{x}_i$ that are exceptional with respect to $\mathsf{Quer}_{\mathsf{sk}}$. If there exists an sk such that this number is greater than $2p^*N$ it outputs 1, otherwise it outputs a uniformly random bit. Using a Chernoff and a union-bound we can routinely show that if the $\mathbf{x}_i \in \{0,1\}^n$ are distributed uniformly, then no such sk exists, except with negligible probability $2^{-\lambda}$. On the other hand, if the $\mathbf{x}_i$ are valid codewords with respect to some key sk, then the event that we count more than $2p^*N$ exceptional words for this key sk is at least p^*. Hence we obtain a distinguisher with advantage $p^*/2 = 1/\mathsf{poly}(\lambda)$.

We conclude that the query function $\mathsf{Quer}_{\mathsf{sk}}$ behaves essentially the same on noisy codewords as it does on uniformly random words.

Compiling-out $\mathcal{O}$-queries. Now we have all the tools in place to provide and analyze our main technical step: Compiling out $\mathcal{O}$-queries from the decoder.

Fix a secret key sk and hence a query function $\mathsf{Quer}_{\mathsf{sk}}$. We can we compile out the $\mathcal{O}$-query from the decoder as follows:

- We will introduce an offline-learner which queries $\mathcal{O}$ on all globally ϵ-heavy queries. Then all of the corresponding query-response pairs are included in into an *augmented secret key* (together with sk). Observe that there can be at most $1/\epsilon = \mathsf{poly}(\lambda)$ many queries of global weight $> \epsilon$, hence this learner is efficient (in terms of its number of $\mathcal{O}$-queries)
- The augmented key is given to an *augmented decoder*, which proceeds as follows. Given an input $\mathbf{x}$, it first computes $q = \mathsf{Quer}_{\mathsf{sk}}(\mathbf{x})$ to determine whether q is in the list of queries that have been learned by the offline learner. If so it takes the corresponding response from the list provided in the augmented secret key. If not it *locally simulates* $\mathcal{O}$ by choosing a uniformly random output.

We will now analyze the correctness of the augmented decoder. Fix a secret key sk and thus a query function $\mathsf{Quer}_{\mathsf{sk}}$. Further fix a codeword $\mathbf{c} = \mathsf{Encode}^{\mathcal{O}}(\mathsf{sk}, b)$. Let Q_{Encode} be the set of $\mathcal{O}$-queries made by the encoder $\mathsf{Encode}^{\mathcal{O}}$ during the encoding process for $\mathbf{c}$.

Further run the offline-learner on sk to obtain an augmented secret key. When we run the augmented decoder on $\mathbf{c} + \mathbf{e}$ (for a Bernoulli error $\mathbf{e}$) the decoder formulates a query $q = \mathsf{Quer}_{\mathsf{sk}}(\mathbf{c} + \mathbf{e})$. We now distinguish 3 different cases.

1. q is globally heavy, i.e. $\mathsf{w}(q) > \epsilon$. In this case the offline-learner has made this query and the augmented decoder will find the corresponding response in the augmented secret key. Thus the simulation is perfect, i.e. the oracle-free augmented decoder and the decoder $\mathsf{Decode}^{\mathcal{O}}$ behave identically.
2. q is globally light, i.e, $\mathsf{w}(q) \leq \epsilon$ but $q \notin Q_{\mathsf{Encode}}$, i.e. q is not an intersection query between encoder and decoder. In this case the augmented decoder will simulate the oracle output locally. As the query q was not made by the encoder, this is also a perfect simulation.
3. q is globally light, i.e, $\mathsf{w}(q) \leq \epsilon$ and $q \in Q_{\mathsf{Encode}}$, i.e. q is an intersection query between encoder and decoder. From the view of the decoder, this case is indistinguishable from the second case as it does not have access to Q_{Encode}. Hence, it will locally simulate the oracle on the query q, which will however be inconsistent with the query made by the encoder. We cannot provide any guarantees in this case and need to assume that a decoding error happens.

Hence, to bound the probability of a decoding error we need to bound the probability that case 3 happens. First assume that $\mathbf{c}$ is *not* exceptional. Then it holds for any $q \in Q_{\mathsf{Encode}}$ with $\mathsf{w}(q) \leq \epsilon$ that $\ell_{\mathbf{c}}(q) < \delta$. Hence we can bound the probability that $q = \mathsf{Quer}_{\mathsf{sk}}(\mathbf{c} + \mathbf{e}) \in Q_{\mathsf{Encode}}$ via a union bound by $|Q_{\mathsf{Encode}}| \cdot \delta$. Now observe that the probability that $\mathbf{c}$ is exceptional is bounded by $3\epsilon^c/\delta^2$. Therefore, we can bound the overall probability that case 3 happens by $|Q_{\mathsf{Encode}}| \cdot \delta + \epsilon^c/\delta^2$. Since $|Q_{\mathsf{Encode}}|$ is a fixed polynomial in λ, we can control this term by choosing $\delta = 1/(|Q_{\mathsf{Encode}}|\mathsf{poly}(\lambda))$ and $\epsilon = \delta^{2/c}/\mathsf{poly}(\lambda)$ as sufficiently small inverse polynomials to achieve any desired inverse polynomial error bound.

Note that by making ϵ smaller we increase the runtime of the offline learner, which is $1/\epsilon$. This trade-off allows us to buy a smaller correctness error by increasing the runtime of the offline learner and thus the size of the augmented secret key. Finally, note that the correctness analysis provided in this paragraph crucially relies on the fact that we add noise! Hence we cannot provide a correctness guarantee for our augmented decoder for noise-free codewords.

Attacking the Augmented Decoder. We have thus shown that a decoder $\mathsf{Decode}^{\mathcal{O}}$ that makes a single oracle query can be compiled into an augmented decoder that uses an augmented (and thus longer) secret key, but does not make any oracle queries. Furthermore, this augmented decoder has a small

correctness error which can, however, be controlled by increasing the size of the augmented secret key. We can thus implement the distinguisher against oracle free decoders sketched above, with the small modification that the distinguisher now needs to add noise to the $\mathbf{x}_i$ to ensure correctness of the augmented decoder.

Compiling out multiple $\mathcal{O}$-queries. The final step in our argument is to deal with decoders that make multiple $\mathcal{O}$-queries. However, as suggested above, the argument for compiling out a single $\mathcal{O}$ query can be applied recursively, thus eliminating oracle queries one by one. The only slightly complicating factor in this case is that we need to ensure that *future* oracle queries made by the decoder are answered consistently. In essence, the augmented decoder has to check for each future query if it is identical to the query that has been compiled out. If so it has to respond consistently, either with a response from the augmented secret key, or with the same locally simulated response. Any fresh query will be answered by an $\mathcal{O}$-query.

This concludes the technical overview.

2.1 Related Work

Black-box separations. The seminal result by Impagliazzo and Rudich [IR89] (IR) famously demonstrated the impossibility of a black-box construction of key-agreement (KA) protocols from one-way permutations (OWPs). This work highlighted fundamental barriers in achieving public-key cryptography from weaker primitives.

Building on this, Rudich, in his thesis [Rud88], established a black-box separation between one-way functions (OWFs) and OWPs. This result, combined with the IR separation, implies that OWFs are also insufficient for black-box constructions of KA and other public-key primitives. Later, Rudich [Rud91] specifically showed a black-box separation between trapdoor functions (a stronger primitive often used for public-key encryption) and KA protocols.

Further delineating the cryptographic hierarchy, Simon [Sim98] demonstrated a black-box separation between OWPs and collision-resistant hash functions (CRHFs). The original IR separation itself has been subject to further investigation: Barak and Mahmoody [BMG09] presented a stronger separation (a more efficient attack in the black-box model), while Brakerski et al. [BKSY11] provided a simpler proof technique for the IR result.

Matsuda and Matsuura [MM11] showed that there is no fully black-box construction of OWPs from length-increasing injective OWFs, even if the latter are only 1-bit-increasing. Mahmoody and Pass [MP12] established that non-interactive commitment schemes cannot be constructed from OWFs in a black-box manner. Döttling and Mour [DM24] showed that any fully-black-box construction of correlation intractable hash for any t-wise independent class of relations from collision-resistant hash, or one-way permutations, must make at least $O(t)$ calls to the underlying base primitive(s).

In the quantum setting, considerable effort has focused on understanding the implications of quantum computation for these classical separations. Several works have investigated the separation between post-quantum one-way functions (PQ-OWFs) and post-quantum key-agreement (PQ-KA) protocols, including [ACC+22, BGV+23, LLLL24, LLLL25b, LLLL25a].

Watermarking. An important application of pseudorandom codes is the watermarking of large language models (LLMs). A large language model takes as input a prompt and a random seed $x \leftarrow \{0,1\}^n$ and creates a response. The model samples output tokens based on the prompt, the seed and the previously sampled tokens. A watermarking scheme for an LLM needs to fulfill the following properties:

– **Undetectability:** The presence of a watermark in the output of the LLM cannot be detected even if the model is queried multiple times adaptively.

– **Robustness:** The watermarked text should be detected as such even if a limited number of bits are adversarially altered.
– **Soundness:** The watermark scheme should not create false positives, i.e., an independently generated text should not be accepted by the detection algorithm.

The approach of [CG24] follows a line of work [CGZ24, KGW+23, ZALW24, KTHL24, FGJ+25] that embeds watermarks in the output of LLMs by biasing the sampling of each token of the output. When the output of the LLM has high entropy the tokens can be biased towards a codeword of a pseudorandom code which can then be detected by the decoder of the PRC. Due to the pseudorandomness of the PRC, a codeword can be use instead of the random seed. Then, the first condition is fulfilled, which ensures that the quality of the output of the language model is not degraded.

This watermarking technique has also been extended to image-generating models [GZS25]. A shortcoming of this watermarking technique is that the detection of the watermark requires a secret key. The work of [FGJ+25] constructs a watermarking scheme with public detectability, but they only achieve a weaker form of robustness. A strong form of robustness was ruled out by [ZEF+24] for secret-key as well as public-key detectable watermarks. For a review of watermarking techniques, see [ZGC+24].

Constructions of Pseudorandom Codes. In their original work [CG24] defined the error-correction capabilities of the code with respect to random errors and in a follow-up work they showed that the constructions are secure with respect to adversarially sampled errors [AAC+24]. Both works consider the notion of secret-key as well as public-key PRCs. In a public-key PRC the encoding algorithm is a public-key algorithm and the decoder is a secret-key algorithm, while in a secret-key PRC both Encode and Decode rely on the secret key.

A zero-bit PRC[7] is constructed in [GM24] under the assumption of local weak PRFs which are implied by the hardness of learning $\log(n)$-juntas over the uniform distribution of n-bit strings. A recent work shows that the problem of learning k-junta distributions is computationally equivalent to the problem of learning k-parity functions with noise [Ber25].

Further constructions of pseudorandom codes are explored in [GG24]. They construct public-key pseudorandom codes from the planted hyperloop assumption [BKR23] and PRGs and they also construct statistically secure pseudorandom codes assuming a space bounded adversary. Additionally, they provide a construction assuming only one-way functions but the scheme tolerates only $o(1)$ errors introduced by a non-adaptive error channel.

In an earlier work [KKRT16] define a notion of pseudorandom codes that differs from the work of [CG24]. Their notion of pseudorandom codes does not require efficient decoding and they do not require actual pseudorandomness. Their codewords only need to be far apart. Such a notion can be constructed from one-way functions.

3 Preliminaries

Notations: Let PPT denote probabilistic polynomial-time. We denote the set of all positive integers upto n as $[n] := \{1, \ldots, n\}$. For any set S, we denote $\mathcal{P}(S)$ as the power set of S. We denote vectors/binary strings using bold font. For any two binary strings $\mathbf{x}, \mathbf{y}$, we use the notation $\mathbf{x}\|\mathbf{y}$ to denote $\mathbf{x}$ concatenated with $\mathbf{y}$. Here, x_i denotes the i^{th} bit of $\mathbf{x}$. We denote $\mathsf{Ham}(\mathbf{x})$ as the Hamming weight of $\mathbf{x}$, i.e., the number of '1' in the binary string $\mathbf{x}$. And, $\mathsf{Ham}(\mathbf{x}, \mathbf{y})$ denotes the Hamming distance between the binary strings $\mathbf{x}$ and $\mathbf{y}$, i.e., the number of indices where $\mathbf{x}$ and $\mathbf{y}$ differ. We denote $\mathcal{B}_r^n(\mathbf{x})$ as the set $\{\mathbf{x}' \in \{0,1\}^n \| \mathsf{Ham}(\mathbf{x}, \mathbf{x}') \leq r\}$, i.e., the set of all strings with Hamming distance of r from $\mathbf{x}$. For any finite set S, $x \leftarrow S$ denotes a uniformly random element x from the set S. Similarly, for any distribution $\mathcal{D}$, $x \leftarrow \mathcal{D}$ denotes an element x drawn from distribution $\mathcal{D}$.

[7] A zero-bit PRC can only encode a single message $m = 0$.

3.1 Probability and Linear Algebra over the Reals

Definition 1 (Bernoulli distribution). *A binary random variable $x \in \{0,1\}$ is said to follow a **Bernoulli distribution** with bias parameter $\rho \in [0,1]$, denoted as Ber_ρ, if $\Pr[x = 1] = \frac{1-\rho}{2}$. We say that $\mathbf{x} \in \{0,1\}^n$ follows Ber_ρ^n, if each component x_i of $\mathbf{x}$ independently follows Ber_ρ.*

Theorem 4 (Markov Inequality). *Let $X \in \mathbb{R}_{\geq 0}$ be a non-negative random variable with expectation $\mathsf{E}[X]$. Then it holds for any $\delta > 0$ that*

$$\Pr[X \geq \delta] \leq \mathsf{E}[X]/\delta$$

We will also use the following simple Markov-like lemma.

Lemma 1. *Let $Z \in [0,1]$ be a (discrete) random variable. Then it holds for all $t \in [0,1]$ that $\Pr[Z > t] \geq \mathsf{E}[Z] - t$.*

Proof. It holds that

$$\begin{aligned}
\mathsf{E}[Z] &= \sum_z z \cdot \Pr[Z = z] \\
&= \sum_{z \leq t} z \cdot \Pr[Z = z] + \sum_{z > t} \underbrace{z}_{\leq 1} \cdot \Pr[Z = z] \\
&\leq t \underbrace{\Pr[Z \leq t]}_{\leq 1} + \Pr[Z > t] \\
&\leq t + \Pr[Z > t],
\end{aligned}$$

It follows that $\Pr[Z > t] \geq \mathsf{E}[Z] - t$.

Theorem 5 (Chernoff bound). *Suppose $X_1, \ldots, X_n$ are independent random variables taking values in $\{0,1\}$. Let $X = \sum_{i=1}^n X_i$ denote their sum and let $\mu = \mathbb{E}[X]$ denote the sum's expected value. Then, for any $\delta \geq 0$,*

$$\Pr(X \geq (1+\delta)\mu) \leq e^{-\frac{\delta^2 \mu}{2+\delta}}$$

Claim 1. Let $r, n \in \mathbb{N}$ such that $r \leq n/2$. Then, for $\rho = 1 - \frac{r}{n}$,

$$\Pr_{\mathbf{x} \leftarrow \mathsf{Ber}_\rho^n}[\mathsf{Ham}(\mathbf{x}) \geq r] \leq e^{-r/3}$$

Proof. We have $\Pr_{\mathbf{x} \leftarrow \mathsf{Ber}_\rho^n}[x_i = 1] = \frac{r}{2n}$. Now, using Chernoff bound, we have $\mu = r/2$, and setting $\delta = 1$, we get the required bound.

Recall that over a vector space $\mathbb{R}^n$, we define the L_p norm $\|\cdot\|_p$ via

$$\|\mathbf{x}\|_p = (\sum_{i=1}^n |x_i|^p)^{1/p}$$

for $p \geq 1$ and $\|\mathbf{x}\|_\infty = \max_{i \in [n]} |x_i|$. We will make use of the following very basic version of the Riesz-Thorin Theorem [Tho48] that bounds $\|\mathbf{x}\|_p$ (for $1 \leq p < \infty$) in terms of $\|\mathbf{x}\|_1$ and $\|\mathbf{x}\|_\infty$.

Lemma 2 (Norm-Interpolation). *It holds for all $1 \leq p < \infty$ and all $\mathbf{x} \in \mathbb{R}^n$ that*

$$\|\mathbf{x}\|_p \leq \|\mathbf{x}\|_\infty^{1-1/p} \cdot \|\mathbf{x}\|_1^{1/p}.$$

Proof. Fix $1 \leq p < \infty$. It holds for all $\mathbf{x} \in \mathbb{R}^n$ that

$$\|\mathbf{x}\|_p^p = \sum_{i=1}^{n} |x_i|^p = \sum_{i=1}^{n} (\underbrace{|x_i|}_{\substack{\leq \max_i |x_i| \\ = \|\mathbf{x}\|_\infty}})^{p-1} \cdot |x_i| \leq \|\mathbf{x}\|_\infty^{p-1} \cdot \sum_{i=1}^{n} |x_i| = \|\mathbf{x}\|_\infty^{p-1} \cdot \|\mathbf{x}\|_1.$$

The statement follows by raising both sides to the power of $1/p$.

3.2 Secret-key Pseudorandom Codes for Bernoulli Channel

A secret-key PRC with κ-correctness ρ-robustness is described by three algorithms $(\mathsf{KeyGen}, \mathsf{Encode}, \mathsf{Decode})$ parameterized by a secret key sk, satisfying the following criteria:

- **(Syntax)** There exist functions $s, n, d, r, r', r'' \in \mathbb{N} \to \mathbb{N}$ such that for all $\lambda \in \mathbb{N}$, $\mathsf{KeyGen} : 1^\lambda \times \{0,1\}^{r''(\lambda)} \to \{0,1\}^{s(\lambda)}$, $\mathsf{Encode} : \{0,1\}^{s(\lambda)} \times \{0,1\}^{d(\lambda)} \times \{0,1\}^{r'(\lambda)} \to \{0,1\}^{n(\lambda)}$, and $\mathsf{Decode} : \{0,1\}^{s(\lambda)} \times \{0,1\}^{n(\lambda)} \times \{0,1\}^{r(\lambda)} \to \{0,1\}^{d(\lambda)} \cup \perp$.
- **(Error correction, or robustness)** For any $\lambda \in \mathbb{N}$, message $m \in \{0,1\}^{d(\lambda)}$,

$$\Pr\left[\mathsf{Decode}(\mathsf{sk}, \mathbf{c} + \mathbf{e}) \neq m \,\middle|\, \begin{array}{r} \mathsf{sk} \leftarrow \mathsf{KeyGen}(1^\lambda) \\ \mathbf{c} \leftarrow \mathsf{Encode}(\mathsf{sk}, m) \\ \mathbf{e} \leftarrow \mathsf{Ber}_\rho^n \end{array}\right] \leq \kappa(\lambda).$$

  where the probability is over the randomness of KeyGen, Encode and error e.
- **(Pseudorandomness)** For any polynomial-time adversary $\mathcal{A}$,

$$\Pr_{\mathsf{sk} \leftarrow \mathsf{KeyGen}(1^\lambda)}\left[\mathcal{A}^{\mathsf{O}_{\mathsf{sk}}(\cdot)}(1^\lambda) = 1\right] - \Pr_{U}\left[\mathcal{A}^{U(1^\lambda, \cdot)}(1^\lambda) = 1\right] \leq \mathrm{negl}(\lambda),$$

  where O_{sk} on any (even previously queried) input $\mathbf{m}$ samples a random $r \in \{0,1\}^{r'(\lambda)}$ and outputs $\mathsf{Encode}(\mathsf{sk}, \mathbf{m}; r)$ and U denotes an oracle that on any (even previously queried) input returns a fresh uniform string in $\{0,1\}^n$.

Remark 1. Note that robustness is defined with respect to the Bernoulli distribution. In the context of error-correcting codes, one typically specifies a decoding radius α such that any word of the form $\mathbf{c} + \mathbf{e}$, where $\mathsf{Ham}(\mathbf{e}) < \alpha n$, can be successfully decoded. By setting $\rho = 1 - \alpha$, we ensure that an error vector $\mathbf{e} \leftarrow \mathsf{Ber}_\rho^n$ satisfies $\mathsf{Ham}(\mathbf{e}) < \alpha n$ with high probability, provided α is a constant. This follows from Claim 1 and the fact that for pseudorandomness, $n = O(\lambda)$.

Remark 2. In [CG24], the authors also define a property called soundness which is not significant in our work – **(Soundness)** For any fixed $\mathbf{c} \in \Sigma^{n(\lambda)}$,

$$\Pr_{\mathsf{sk} \leftarrow \mathsf{KeyGen}(1^\lambda)}\left[\mathsf{Decode}(\mathsf{sk}, \mathbf{c}) = \perp\right] \geq 1 - \mathrm{negl}(\lambda).$$

If the scheme can only encode a singular message (i.e., $k = 0$), it is called a *zero-bit PRC*. Christ and Gunn [CG24] required the soundness condition to ensure that the construction of zero-bit PRCs is non-trivial.

3.3 Secret key encryption

A secret key encryption scheme $\mathsf{SKE} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ for unbounded length messages consists of the following PPT algorithms.

- $\mathsf{KeyGen}(1^\lambda)$: The setup algorithm is a randomized algorithm that takes as input the security parameter λ and outputs a secret key sk.
- $\mathsf{Enc}(\mathsf{sk}, \mathbf{m} \in \{0,1\}^*)$: The encryption algorithm is a randomized algorithm that takes as input a secret key sk and a message $\mathbf{m}$ and outputs a ciphertext ct.
- $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct})$: The decryption algorithm takes as input a secret key sk and a ciphertext ct and outputs either a message $\mathbf{m} \in \{0,1\}^*$ or $\perp$.

Correctness. For correctness, we require that for all $\lambda \in \mathbb{N}, \mathbf{m} \in \{0,1\}^*$ and $\mathsf{sk} \leftarrow \mathsf{KeyGen}(1^\lambda)$,

$$\Pr[\mathsf{Dec}(\mathsf{sk}, \mathsf{Enc}(\mathsf{sk}, \mathbf{m})) = \mathbf{m}] = 1$$

where the probability is over the random bits used in the encryption algorithm.

$-CPA-SKE Security. Consider the following experiment with an adversary $\mathcal{A}$ that takes 1^λ and 1^n as input.

- **Initialization Phase:** The challenger runs $\mathsf{sk} \leftarrow \mathsf{KeyGen}(1^\lambda)$.
- **Pre-Challenge Query Phase:** $\mathcal{A}$ is allowed to make polynomially many encryption queries. For each query $\mathbf{m}$, the challenger computes $\mathsf{ct} \leftarrow \mathsf{SKE.Enc}(\mathsf{sk}, \mathbf{m})$ and returns ct to $\mathcal{A}$.
- **Challenge Phase:** The challenger randomly chooses $b \in \{0,1\}$. $\mathcal{A}$ sends a message $\mathbf{m} \in \{0,1\}^n$ to the challenger. If $b = 0$, it samples a truly random string ct^* from the ciphertext space. Else, it computes a ciphertext $\mathsf{ct}^* = \mathsf{Enc}(\mathsf{sk}, \mathbf{m})$ and sends it to $\mathcal{A}$.[8]
- **Post-Challenge Query Phase:** $\mathcal{A}$ is allowed to make polynomially many encryption queries. For each query $\mathbf{m}$, the challenger computes $\mathsf{ct} \leftarrow \mathsf{SKE.Enc}(\mathsf{sk}, \mathbf{m})$ and returns ct to $\mathcal{A}$.
- **Response Phase:** $\mathcal{A}$ outputs $b' \in \{0,1\}$ and wins the experiment if $b = b'$.

Definition 2. *An SKE scheme is said to be $-CPA secure if for all PPT adversaries $\mathcal{A}$, there exists a negligible function* $\mathrm{negl}(\cdot)$ *such that for all $\lambda \in \mathbb{N}, n \in \mathbb{N}$,*

$$\Pr[\mathcal{A} \textit{ wins in the above experiment}] \leq \frac{1}{2} + \mathrm{negl}(\lambda)$$

3.4 Pseudorandom Permutations

A pseudorandom permutation $\mathsf{PRP} = (\mathsf{KeyGen}, \mathsf{Eval})$ consists of PPT algorithms $\mathsf{KeyGen} : \{0,1\}^\lambda \to \{0,1\}^{s(\lambda)}$ and $\mathsf{Eval} : \{0,1\}^{s(\lambda)} \times \{0,1\}^{n(\lambda)} \to \{0,1\}^{n(\lambda)}$ such that for any $\mathsf{sk} \leftarrow \mathsf{KeyGen}(1^\lambda)$, $\mathsf{Eval}(\mathsf{sk}, \cdot) : \{0,1\}^n \to \{0,1\}^n$ is a permutation. Also, for any PPT adversary $\mathcal{A}$, there exists a negligible function $\mathrm{negl}(\cdot)$ such that for any $\lambda \in \mathbb{N}$, the following advantage function $\mathrm{Adv}_f^{\mathrm{PRP}}(\lambda, \mathcal{A})$ is $\mathrm{negl}(\lambda)$:

$$\mathrm{Adv}^{\mathrm{PRP}}(\lambda, \mathcal{A}) = \Pr\left[x' = x^* \middle| \begin{array}{l} \mathsf{sk} \leftarrow \mathsf{KeyGen}(1^\lambda) \\ y^* \leftarrow \mathsf{Eval}(\mathsf{sk}, x^*) \\ x' \leftarrow \mathcal{A}(1^\lambda, y^*) \end{array}\right]$$

where the probability is over the randomness of $\mathcal{A}$, KeyGen and the random choice of x^*.

[8] Note that this security notion implies the standard indistinguishability security notion where the adversary sends two messages m_0, m_1 and receives an encryption of one of the messages.

3.5 Linear Error-Correcting Codes

Definition 3 (Linear Error-Correcting Code). *Let q be a power of 2 and $\mathbb{F}_q$ be the finite field of order q. A* linear $[n, k, d]$ code *C over $\mathbb{F}_q$ is a k-dimensional subspace of $\mathbb{F}_q^n$ where n is the codeword length, k is the message length and d is the minimum distance. The minimum distance d is defined as*

$$d \;=\; \min_{\substack{\mathbf{u},\mathbf{v}\in C \\ \mathbf{u}\neq\mathbf{v}}} \mathsf{Ham}(\mathbf{u},\mathbf{v})$$

The code is defined by two efficient algorithms $\mathsf{Encode}, \mathsf{Decode}$ *such that the following properties hold:*

- ***Correctness:*** *A code C is correct for all $\mathbf{m} \in \mathbb{F}_q^k$, $\mathsf{Decode}(\mathsf{Encode}(\mathbf{m})) = \mathbf{m}$*
- ***Robustness:*** *A code C is robust for up to t errors if for all $\mathbf{m} \in \mathbb{F}_q^k$ and $\mathbf{e} \in \mathbb{F}_q^n$ such that the number of non-zero symbols in $\mathbf{e}$ is less than t, it holds that $\mathsf{Decode}(\mathsf{Encode}(\mathbf{m}) + \mathbf{e}) = \mathbf{m}$*

Reed-Solomon codes are $[n, k, n-k+1]$ linear error-correcting codes [RS60].

3.6 Black-Box Constructions and Separations

Definition 4 (Black-box Construction). *A construction of a Pseudorandom Code* $\mathsf{PRC} = (\mathsf{KeyGen}, \mathsf{Encode}, \mathsf{Decode})$ *from a family of oracle-sets $\{\overline{\mathcal{O}}_\lambda\}$ is* black-box *if the following properties hold for any $\mathcal{O}$ from the family $\overline{\mathcal{O}}_\lambda$*

- ***Construction Efficiency****: The algorithms $\mathsf{KeyGen}^{\mathcal{O}}$, $\mathsf{Encode}^{\mathcal{O}}$, and $\mathsf{Decode}^{\mathcal{O}}$ run in time polynomial in $\lambda \in \mathbb{N}$ (and input lengths) and make at most $\mathsf{poly}(\lambda)$ queries to $\mathcal{O}$.*
- ***Correctness****: For any $\lambda \in \mathbb{N}$, message $m \in \{0,1\}^{d(\lambda)}$,*

$$\Pr\left[\mathsf{Decode}^{\mathcal{O}}(\mathsf{sk}, \mathbf{c}+\mathbf{e}) \neq m \;\middle|\; \begin{array}{r} \mathsf{sk} \leftarrow \mathsf{KeyGen}^{\mathcal{O}}(1^\lambda) \\ \mathbf{c} \leftarrow \mathsf{Encode}^{\mathcal{O}}(\mathsf{sk}, m) \\ \mathbf{e} \leftarrow \mathsf{Ber}_\rho^n \end{array}\right] \;\leq\; \kappa(\lambda).$$

  where the probability is over the randomness of $\mathsf{KeyGen}, \mathsf{Encode}$ *and error e.*
- ***Security****: For any (unbounded) adversary $\mathcal{A}$ that makes only polynomially many queries, there exists a negligible function* $\mathrm{negl}(\cdot)$ *such that for all $\lambda \in \mathbb{N}$,*

$$\Pr_{\substack{\mathcal{O}\leftarrow\overline{\mathcal{O}}_\lambda \\ \mathsf{sk}\leftarrow\mathsf{KeyGen}^{\mathcal{O}}(1^\lambda)}}\left[\mathcal{A}^{\mathcal{O}(\cdot),\mathsf{O}_{\mathsf{sk}}(\cdot)}(1^\lambda)=1\right] \;-\; \Pr_{U}\left[\mathcal{A}^{\mathcal{O}(\cdot),U(1^\lambda,\cdot)}(1^\lambda)=1\right] \;\leq\; \mathrm{negl}(\lambda),$$

  where O_{sk} on any (even previously queried) input m, samples a random $r \in \{0,1\}^{r'(\lambda)}$ and outputs $\mathsf{Encode}^{\mathcal{O}}(\mathsf{sk}, m; r)$ and U denotes an oracle that on any (even previously queried) input returns a fresh uniform string in Σ^n.

Definition 5 (Black-box Separation). *We say a PRC is black-box separated from a family of oracle sets if for any black-box construction of a PRC from $\{\overline{\mathcal{O}}_\lambda\}$ there exists an (unbounded) adversary $\mathcal{A}$ which makes only polynomially many oracle queries, breaks the pseudorandomness of the PRC.*

4 Local Oracles

In this section, we introduce a new class of oracles, called **local oracles**. A query q is said to be *related* to a set of query-response pairs $\{(q_i, \mathcal{O}(q_i))\}_i$ if the oracle's response to q can be computed or meaningfully constrained using the information in the set. Intuitively, this captures semantic

dependency among queries. An oracle is said to be local if, for any set Z of query-response pairs, the number of queries q such that q is related to Z is bounded by a polynomial in $|Z|$.

Local oracles also possess a simulation strategy for two-stage algorithms, described as follows. In the first stage, the algorithm issues a bounded number of queries, each of which is answered by the oracle. It then outputs a new query q, which is required to be unrelated to any of the previous queries. The simulator responds to q with a value r, without accessing the oracle. In the second stage, the algorithm continues interacting with the oracle. These subsequent queries are simulated by a stateful simulator that has access to the original oracle and the pair (q, r).

Because the simulator's response r may not match the oracle's actual output on q, it must carefully handle new queries. If a new query is related to (q, r), the simulator generates a consistent answer. Otherwise, it forwards the query to the oracle to ensure consistency with the responses provided in the first phase.

Definition 6. *A family of local oracles is a collection of oracle-relation pairs $\{(\mathcal{O}_{i,\lambda}, \mathcal{R}_{i,\lambda})\}_{i,\lambda}$ such that for each i, the oracle $\mathcal{O}_{i,\lambda}$ has domain $\mathcal{X}_\lambda$ and range $\mathcal{Y}_\lambda$, and $\mathcal{R}_{i,\lambda}$ is a relation between a query and a set of query-response pairs: $\mathcal{R}_{i,\lambda} \subseteq \mathcal{X}_\lambda \times \mathcal{P}(\mathcal{Z}_{i,\lambda})$, where $\mathcal{Z}_{i,\lambda} = \{(x, \mathcal{O}_{i,\lambda}(x)) \mid x \in \mathcal{X}_\lambda\}$. The following conditions must also be satisfied:*

1. ***Locality:*** *There exists a polynomial* poly *such that for any $Z \in \mathcal{P}(\mathcal{Z}_{i,\lambda})$, the number of queries $q \in \mathcal{X}_\lambda$ for which $(q, Z) \in \mathcal{R}_{i,\lambda}$ is at most* $\mathsf{poly}(|Z|)$.
2. ***Simulatability:*** *There exist an algorithm S_1, a* ***stateful*** *oracle-aided algorithm S_2 and a negligible function* $\mathrm{negl}(\cdot)$ *such that for any pair of oracle-aided algorithms $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, the output distributions of the following two experiments are statistically close:*
   - *Sample $(\mathcal{O}, \mathcal{R})$ from the distribution $\{(\mathcal{O}_{i,\lambda}, \mathcal{R}_{i,\lambda})\}_i$.*
   - *The adversary $\mathcal{A}_1^{\mathcal{O}}(1^\lambda, \mathcal{R})$ makes $poly(\lambda)$ queries to the oracle and outputs a query q and internal state $\mathsf{st}_{\mathcal{A}}$. Let T be the set of all query-response pairs collected during $\mathcal{A}_1$'s execution. If $(q, T) \in \mathcal{R}$, abort the experiment.*
     - ***Real world:*** *Compute $r := \mathcal{O}(q)$, and run $\mathcal{A}_2$ with oracle access to $\mathcal{O}(\cdot)$ and input $(1^\lambda, q, r, \mathsf{st}_{\mathcal{A}})$.*
     - ***Simulated world:*** *Run S_1 on input q to obtain $(r, \mathsf{st}_{\mathsf{S}}) := \mathsf{S}_1(1^\lambda, q)$. Then, run $\mathcal{A}_2$ with oracle access to $\mathsf{S}_2^{\mathcal{O}}(1^\lambda, \mathsf{st}_{\mathsf{S}}, \cdot)$ and input $(1^\lambda, q, r, \mathsf{st}_{\mathcal{A}})$.*
   - *The second adversary $\mathcal{A}_2$ outputs a bit in $\{0, 1\}$ after making* $\mathsf{poly}(\lambda)$ *many oracle queries.*

Let Exp^{Real} denote a run of the experiment where $\mathcal{A}$ interacts with the real world and let Exp^{Sim} denote a run of the experiment where $\mathcal{A}$ interacts with the simulated world, then

$$|\Pr[\mathsf{Exp}^{Real}(1^\lambda, \mathcal{A}) = 1] - \Pr[\mathsf{Exp}^{Sim}(1^\lambda, \mathcal{A}) = 1]| < \mathrm{negl}(\lambda)$$

This definition encompasses a wide class of oracles. Here, we explain how some common functionalities achieve the definition.

Random oracles are local. Consider a random oracle which is typically used to model a OWF or a OWP. Here, the relation $\mathcal{R}$ simply checks whether q is a query present as a query in Z. For an unrelated query q, the simulator $\mathsf{S}_1(q)$ returns a random $r \in \mathcal{Y}$ and sets the state as $\mathsf{st}_{\mathsf{S}} = (q, r)$. The simulator $\mathsf{S}_2^{\mathcal{O}}(\mathsf{st}, t)$ then responds with $\mathcal{O}(t)$ for $t \neq q$ whereas returns r for $t = q$. This simulation is exact in the case of OWFs. However, In the case of OWPs, this simulation is imperfect with negligible probability, since S_1 might choose a value r already present in the image of some previous query in Z, which could violate injectivity.

Trapdoor permutation oracles are local. A more intricate example of a local oracle arises in the context of trapdoor permutations. Here, the oracle is defined as a tuple $(\mathsf{KeyGen}, \mathsf{Eval}, \mathsf{Inv})$ with the following components:

$$\mathsf{KeyGen} : \mathcal{SK} \to \mathcal{PK}, \quad \mathsf{Eval} : \mathcal{PK} \times \mathcal{M} \to \mathcal{N}, \quad \mathsf{Inv} : \mathcal{SK} \times \mathcal{N} \to \mathcal{M},$$

where KeyGen maps a secret key $\mathsf{sk} \in \mathcal{SK}$ to a public key $\mathsf{pk} \in \mathcal{PK}$, Eval evaluates the permutation under the public key, and Inv inverts it using the secret key.

The oracle $\mathcal{O}$ accepts queries of the form (mode, q), where $\mathsf{mode} \in \{\mathsf{KeyGen}, \mathsf{Eval}, \mathsf{Inv}\}$ and q belongs to the domain of the corresponding function. Suppose an adversary observes the set of query-response pairs

$$Z = \{((\mathsf{KeyGen}, \mathsf{sk}), \mathsf{pk}),\ ((\mathsf{Eval}, \mathsf{pk}, x), y)\}.$$

Then the query $(\mathsf{Inv}, \mathsf{sk}, y)$ is *related* to Z, because given both $(\mathsf{KeyGen}, \mathsf{sk}) = \mathsf{pk}$ and $(\mathsf{Eval}, \mathsf{pk}, x) = y$, one can deduce that $(\mathsf{Inv}, \mathsf{sk}, y) = x$. Thus, a simulator without oracle access cannot correctly respond to $(\mathsf{Inv}, \mathsf{sk}, y)$ unless this information is already known.

In contrast, queries that are *unrelated* to Z do not pose this issue. For example, for a fresh secret key $\mathsf{sk}' \neq \mathsf{sk}$, the simulator S_1 can simulate $(\mathsf{KeyGen}, \mathsf{sk}')$ by generating a fresh public key pk'. However, to ensure consistency across the simulation, this key pair must be retained and passed to the stateful simulator S_2, since the second-stage adversary $\mathcal{A}_2$ may issue follow-up queries involving $(\mathsf{pk}', \mathsf{sk}')$ (e.g., $(\mathsf{Eval}, \mathsf{pk}', x)$ or $(\mathsf{Inv}, \mathsf{sk}', y)$) and expect coherent answers.

In general, a query $(\mathsf{Inv}, \mathsf{sk}, y)$ is related to Z if it is explicitly present in Z, or if both $((\mathsf{KeyGen}, \mathsf{sk}), \mathsf{pk})$ and $((\mathsf{Eval}, \mathsf{pk}, x), y)$ appear in Z. A similar statement holds for queries of the form $(\mathsf{Eval}, \mathsf{pk}, x)$. Similarly, a query $(\mathsf{KeyGen}, \mathsf{sk})$ is related to Z if it appears explicitly, or if there exist queries $((\mathsf{Eval}, \mathsf{pk}, x), y)$ and $((\mathsf{Inv}, \mathsf{sk}, y), x)$ such that $\mathsf{KeyGen}(\mathsf{sk}) = \mathsf{pk}$.

Since each pair of query-response elements in Z can relate to at most a single query, the total number of queries related to Z is bounded by $O(|Z|^2)$. This quadratic bound implies that the oracle satisfies the locality property.

Pseudorandom code oracles are not local. It is important to observe that high-error-tolerant PRC oracles are *not* local. The reason is that, for a single Encode query producing a codeword $\mathbf{c}$, all Decode queries corresponding to codewords within the Hamming sphere centered at $\mathbf{c}$ are considered related. Since the number of such queries is exponential in the dimension of the codeword, the locality condition which requires only a polynomial number of related queries, cannot be satisfied.

5 Black-box Impossibility of Pseudorandom Codes

In this section, we present the proof of Theorem 2, i.e., there does not exist a black-box construction of PRC from any local oracle $\mathcal{O}$.

For any oracle $\mathcal{O}$, let $\mathsf{PRC}^{\mathcal{O}} = (\mathsf{KeyGen}^{\mathcal{O}}, \mathsf{Encode}^{\mathcal{O}}, \mathsf{Decode}^{\mathcal{O}})$ denote a pseudorandom error-correcting code. Without loss of generality, we can assume that KeyGen does not make any oracle queries. Concretely, one simply treats its internal randomness as part of the secret key, and modifies Encode and Decode to internally run KeyGen at the start, using this randomness to generate the required key.

Theorem 6. *Let $\mathcal{O}$ be a local oracle and $\mathsf{PRC}^{\mathcal{O}} = (\mathsf{KeyGen}, \mathsf{Encode}^{\mathcal{O}}, \mathsf{Decode}^{\mathcal{O}})$ be a single-bit ρ-robust PRC where ρ is any constant between 0 and $1/2$. Then, there exists an (unbounded) distinguisher $\mathcal{D}$ and a polynomial poly such that for all $\lambda \in \mathbb{N}$,*

- *$\mathcal{D}$ makes $\mathsf{poly}(\lambda)$ many oracle queries.*
- *$\mathcal{D}$ breaks the pseudorandomness property of $\mathsf{PRC}^{\mathcal{O}}$ with inverse polynomial advantage.*

The proof proceeds in two main steps. In the first part, we show that for any given secret key sk, there exists a corresponding augmented secret key sk' such that, for a randomly chosen (errorless) codeword c, the adversary $\mathcal{A}$ can correctly decode c using sk' **without** querying the random oracle. In the second part, we prove that no such augmented secret key sk' can successfully decode many

uniformly random strings with non-negligible probability, thus allowing the adversary to distinguish between codewords and uniformly random strings. This contradicts the pseudorandomness property of the PRC.

For the first part, we need to show that every query can either be learned or simulated. To this end, we divide the queries made by the decoder into global and local queries and prove key properties in the following section.

5.1 Global and Local Heavy Queries

Let Q be a finite set and let $\mathsf{Quer} : \{0,1\}^n \times \{0,1\}^r \to Q$ be an arbitrary function which maps an input word $\mathbf{x} \in \{0,1\}^n$ and a string $r \in \{0,1\}^r$ to a *query* $\mathsf{Quer}(\mathbf{x}, r) \in Q$. We think of $r \in \{0,1\}^r$ as random coins provided to Quer. In the following, if not specified, assume that $\mathbf{u} \leftarrow \{0,1\}^n$ is chosen uniformly random, $r \leftarrow \{0,1\}^r$ are uniformly random coins and that $\mathbf{e} \leftarrow \mathsf{Ber}_\rho^n$ follows a Bernoulli distribution.

We will now define two weight functions for queries, namely a global and local weight function which assign weights to queries.

- The *global weight function* $\mathsf{w}(q)$ measures how likely it is that a uniformly random input word leads to the query q. It is defined via
$$\mathsf{w}(q) = \Pr_{\mathbf{u},r}[\mathsf{Quer}(\mathbf{u}, r) = q],$$
where the probability is taken over the uniform choice of $\mathbf{u} \leftarrow \{0,1\}^n$.
- The *local weight function* $\ell_{\mathbf{x}}(q)$ measures how likely it is that we get the query q, if we start from a fixed word $\mathbf{x}$ and add Bernoulli noise $\mathbf{e}$. It is defined by
$$\ell_{\mathbf{x}}(q) = \Pr_{\mathbf{e},r}[\mathsf{Quer}(\mathbf{x} + \mathbf{e}, r) = q],$$
where $\mathbf{e} \leftarrow \mathsf{Ber}_\rho^n$ follows a Bernoulli distribution.

We will make critical use of the Bonami-Beckner hypercontractivity inequality [Bon70, Bec75]. We will first need some terminology concerning boolean functions.

Definition 7 (L_p-Norm and Smoothing of Boolean Functions). *Let $f : \{0,1\}^n \to \mathbb{R}$ be a boolean function. For a $1 \leq p < \infty$, we define the L_p-norm of f via*
$$\|f\|_{(p)} = (\mathsf{E}_{\mathbf{u}}[|f(\mathbf{u})|^p])^{1/p}$$
and
$$\|f\|_\infty = \max_{\mathbf{x} \in \{0,1\}^n} [|f(\mathbf{x})|].$$
Furthermore, for a $0 < \rho < 1/2$, the noise operator $\mathbf{T}_\rho$ *maps f to a boolean function $\mathbf{T}_\rho f : \{0,1\}^n \to \mathbb{R}$ given by*
$$(\mathbf{T}_\rho f)(\mathbf{x}) = \mathsf{E}_{\mathbf{e} \leftarrow \mathsf{Ber}_\rho^n}[f(\mathbf{x} + \mathbf{e})].$$

Note that while very similar to the standard L_p norm on $\mathbb{R}^n$, $\| f\|_{(p)}$ norm is not the same as the L_p norm on the truth table of f. The difference is a normalization factor $2^{-n/p}$, i.e. it holds for all f that
$$\|f\|_{(p)} = 2^{-n/p} \cdot \|f\|_p.$$
One effect of this normalization is that the relationship of the L_p and L_q norms reverses, i.e. while it holds for all $\mathbf{x}$ that $\|\mathbf{x}\|_q \leq \|\mathbf{x}\|_p$ for $1 \leq p < q$, it holds that $\|f\|_{(p)} \leq \|f\|_{(q)}$ for $1 \leq p < q$ [9]. Hence, to avoid confusion, we opted to use a slightly different notation for $\| \cdot \|_{(p)}$.

[9] This fact is easily established via the Hölder inequality

Theorem 7 (Hypercontractivity Theorem [Bon70, Bec75, O'D14]). *Let $f : \{0,1\}^n \to \mathbb{R}$ be a boolean function, let $0 \leq s \leq t < \infty$ and let $0 \leq \rho \leq \sqrt{\frac{s}{t}}$. Then it holds that*

$$\|\mathbf{T}_\rho f\|_{(1+t)} \leq \|f\|_{(1+s)}.$$

We will now use the hypercontractivity theorem to relate the global and local weights defined above. The following lemma allows us to bound the r-th moment of $\ell_{\mathbf{u}}(q)$ (for a uniform $\mathbf{u} \leftarrow \{0,1\}^n$) by a low degree term in the global weight $\mathsf{w}(q)$.

Lemma 3. *Let $0 < \rho < 1$ and let $t \geq 0$. Fix a query $q \in Q$. Then it holds that*

$$\mathsf{E}_{\mathbf{u}}[\ell_{\mathbf{u}}(q)^{1+t}] \leq (\mathsf{w}(q))^{(1+t)/(1+\rho^2 t)}.$$

Furthermore, for $0 < \delta \leq 1$, we have the tail-bound

$$\Pr_{\mathbf{u}}[\ell_{\mathbf{u}}(q) \geq \delta] \leq \frac{(\mathsf{w}(q))^{(1+t)/(1+\rho^2 t)}}{\delta^{1+t}}.$$

Proof. Define the function $p_q : \{0,1\}^n \to \mathbb{R}$ by $p_q(\mathbf{x}) = \Pr_r[\mathsf{Quer}(\mathbf{x}, r) = q]$. First note that

$$(\mathbf{T}_\rho p_q)(\mathbf{x}) = \mathsf{E}_{\mathbf{e} \leftarrow \mathsf{Ber}_\rho^n}[p_q(\mathbf{x} + \mathbf{e})] = \Pr_{\mathbf{e}, r}[\mathsf{Quer}(\mathbf{x} + \mathbf{e}, r) = q] = \ell_{\mathbf{x}}(q).$$

Hence it holds by hypercontractivity (Theorem 7) with $s = \rho^2 \cdot t$ (which satisfies the requirements of the theorem) that

$$\mathsf{E}_{\mathbf{u}}[\ell_{\mathbf{u}}(q)^r] = \|\mathbf{T}_\rho p_q\|_{(1+t)}^{1+t} \leq \|p_q\|_{(1+s)}^{1+t}. \tag{1}$$

It further holds that

$$\begin{aligned}
\|p_q\|_{(1+s)}^{1+s} &= \mathsf{E}_{\mathbf{u}}[p_q(\mathbf{u})^{1+s}] = \mathsf{E}_{\mathbf{u}}[\Pr_r[\mathsf{Quer}(\mathbf{u}, r) = q]^{1+s}] \\
&\leq \mathsf{E}_{\mathbf{u}}[\Pr_r[\mathsf{Quer}(\mathbf{u}, r) = q]] = \Pr_{\mathbf{u}, r}[\mathsf{Quer}(\mathbf{u}, r) = q] \\
&= \mathsf{w}(q),
\end{aligned}$$

where the inequality $\Pr_r[\mathsf{Quer}(\mathbf{u}, r) = q]^{1+s} \leq \Pr_r[\mathsf{Quer}(\mathbf{u}, r) = q]$ holds as $1 + s > 1$ and $\Pr_r[\mathsf{Quer}(\mathbf{u}, r) = q] \leq 1$. Hence, combining with (1) we obtain that

$$\mathsf{E}_{\mathbf{u}}[\ell_{\mathbf{u}}(q)^{1+t}] \leq (\mathsf{w}(q))^{(1+t)/(1+s)} = (\mathsf{w}(q))^{(1+t)/(1+\rho^2 t)}.$$

The *furthermore* part of the lemma-statement now follows routinely from the Markov-inequality (Theorem 4): As $\ell_{\mathbf{x}}(q) = \Pr_{\mathbf{e}, r}[\mathsf{Quer}(\mathbf{x} + \mathbf{e}, r) = q] \geq 0$, it holds that

$$\begin{aligned}
&\Pr_{\mathbf{u}}[\ell_{\mathbf{u}}(q) \geq \delta] = \Pr_{\mathbf{u}}[(\ell_{\mathbf{u}}(q))^{1+t} \geq \delta^{1+t}] \\
&\leq \mathsf{E}_{\mathbf{u}}[(\ell_{\mathbf{u}}(q))^{1+t}] \leq (\mathsf{w}(q))^{(1+t)/(1+\rho^2 t)} / \delta^{1+t}.
\end{aligned}$$

5.2 Exceptionality

We will now define a notion of *exceptionality* for words $\mathbf{x} \in \{0,1\}^n$. In a nutshell, we call an $\mathbf{x}$ exceptional if it exhibits code-word like properties, in the sense that for such an $\mathbf{x}$ there exists a query $q \in Q$ which has low global weight but high local weight with respect to $\mathbf{x}$. Intuitively, this can be interpreted as the set $C_q = \{\mathbf{y} \in \{0,1\}^n \mid \mathsf{Quer}(\mathbf{y}) = q\}$ having a large intersection with the hamming ball around $\mathbf{x}$.

Definition 8. *Fix $0 < \epsilon < \delta < 1$. We say that a word $\mathbf{x} \in \{0,1\}^n$ is (ϵ, δ)-exceptional with respect to a query function $\mathsf{Quer} : \{0,1\}^n \times \{0,1\}^r \to Q$, in short we say the event $\mathsf{EXC}_{\mathsf{Quer},\epsilon,\delta}(\mathbf{x})$ happens, if there exists a query $q \in Q$ such that $\mathsf{w}(q) < \epsilon$ and $\ell_{\mathbf{x}}(q) \geq \delta$ (with respect to the query function Quer).*

Looking ahead, we will use the notion of exceptionality at the core-part of our separation, namely to argue that the probability that there are *globally non-heavy* intersection queries between encoder and decoder is necessarily low. Hence, it will suffice for our learner to learn the globally heavy queries, which are independent of the word to be decoded.

We will now use Lemma 3 to establish our main result of this section, which shows that when we choose $\mathbf{u} \leftarrow \{0,1\}^n$ uniformly random, then the probability that $\mathbf{u}$ is (ϵ, δ)-exceptional is at most $\frac{\epsilon^{(1+t)/(1+\rho^2 t)-1}}{\delta^{1+t}}$. Note that this bound is independent of $|Q|$ and even the query function Quer.

Theorem 8. *Fix an integer $n > 0$ and fix a $\rho > 0$. Fix an integer $r > 0$, any finite set Q, and any query function $\mathsf{Quer} : \{0,1\}^n \times \{0,1\}^r \to Q$. Let w, ℓ be defined as above.Fix any $0 < \epsilon < \delta < 1$ and let $\mathsf{EXC}_{\mathsf{Quer},\epsilon,\delta}(\cdot)$ be as defined in Definition 8. Let $\mathbf{u} \leftarrow \{0,1\}^n$. Then it holds that*

$$\Pr_{\mathbf{u}}[\mathsf{EXC}_{\mathsf{Quer},\epsilon,\delta}(\mathbf{u})] \leq \frac{\epsilon^{(1+t)/(1+\rho^2 t)-1}}{\delta^{1+t}}$$

Note that n and r are only stated for syntactical reasons. Before we go into the proof of the theorem, notice that the exponent $(1+t)/(1+\rho^2 t) - 1$ is greater than 0 for any choice of $\rho \in (0,1)$ and any $t > 0$. Hence the choice of the concrete t will not be critical when we use Theorem 8, hence for convenience we can choose $t = 1$. Note that we can control this bound by making ϵ small, which will effectively translate to making our learner learn more queries.

Proof. For notational convenience, denote $\gamma = (1+t)/(1+\rho^2 t) > 0$ in this proof. By a union-bound and Lemma 3 it holds that

$$\begin{aligned}
\Pr_{\mathbf{u}}[\mathsf{EXC}_{\mathsf{Quer},\epsilon,\delta}(\mathbf{u})] &= \Pr_{\mathbf{u}}[\exists q \in Q \text{ s.t. } \mathsf{w}(q) < \epsilon \text{ and } \ell_{\mathbf{u}}(q) \geq \delta] \\
&\leq \sum_{\substack{q \in Q \\ \mathsf{w}(q) < \epsilon}} \Pr_{\mathbf{u}}[\ell_{\mathbf{u}}(q) \geq \delta] \\
&\leq \sum_{\substack{q \in Q \\ \mathsf{w}(q) < \epsilon}} \mathsf{w}(q)^{\gamma} / \delta^{1+t} \\
&= \frac{1}{\delta^{1+t}} \cdot \sum_{\substack{q \in Q \\ \mathsf{w}(q) < \epsilon}} \mathsf{w}(q)^{\gamma}.
\end{aligned}$$

Now define a vector $\mathbf{w} \in \mathbb{R}^Q$ such that $w_q = \mathsf{w}(q)$ if $\mathsf{w}(q) < \epsilon$ and otherwise $w_q = 0$. By the above it holds that

$$\Pr_{\mathbf{u}}[\mathsf{EXC}_{\mathsf{Quer},\epsilon,\delta}(\mathbf{u})] \leq \frac{1}{\delta^{1+t}} \cdot \|\mathbf{w}\|_{\gamma}^{\gamma}.$$

Furthermore, since every component of $\mathbf{w}$ is ϵ-bounded (and greater equal 0) we have that

$$\|\mathbf{w}\|_{\infty} = \max_{q \in Q} |w_q| < \epsilon.$$

Finally, it holds that

$$\|\mathbf{w}\|_1 \leq \sum_{q \in Q} \mathsf{w}(q) = \sum_{q \in Q} \Pr_{\mathbf{u},r}[\mathsf{Quer}(\mathbf{u}, r) = q] = 1.$$

Hence, we have that $\|\mathbf{w}\|_1 \leq 1$ and $\|\mathbf{w}\|_{\infty} < \epsilon$. By the norm-interpolation lemma (Lemma 2) we get that $\|\mathbf{w}\|_{\gamma}^{\gamma} < \epsilon^{\gamma - 1}$. We conclude that

$$\Pr_{\mathbf{u}}[\mathsf{EXC}_{\mathsf{Quer},\epsilon,\delta}(\mathbf{u})] \leq \frac{\epsilon^{\gamma-1}}{\delta^{1+t}} = \frac{\epsilon^{(1+t)/(1+\rho^2 t)-1}}{\delta^{1+t}}$$

5.3 Compiling-Out Oracle Queries

We now describe how to eliminate oracle queries from the decoding algorithm, thereby obtaining an oracle-free decoder. We start by defining PRCs with an augmented decoding procedure.

Definition 9 (Augmented Decoding). *A secret-key PRC* $\mathsf{PRC}^{\mathcal{O}} = (\mathsf{KeyGen}, \mathsf{Encode}^{\mathcal{O}}, \mathsf{Decode}^{\mathcal{O}})$ *has* κ-***augmented correctness*** *with respect to augmented decoding algorithms* $(\mathsf{LearnQuery}^{\mathcal{O}}, \mathsf{ADecode}^{\mathcal{O}})$ *if the following holds.* $\mathsf{ADecode}$ *is a PPT oracle algorithm and for all* $\lambda \in \mathbb{N}$*,*

$$\Pr\left[\mathsf{ADecode}^{\mathcal{O}}(\mathsf{sk}, \mathsf{aux}, \mathbf{c}+\mathbf{e}) \neq m \;\middle|\; \begin{array}{r} \mathsf{sk} \leftarrow \mathsf{KeyGen}(1^\lambda) \\ m \leftarrow \{0,1\}^d, \mathbf{e} \leftarrow \mathsf{Ber}_\rho^n \\ \mathbf{c} \leftarrow \mathsf{Encode}^{\mathcal{O}}(\mathsf{sk}, m) \\ \mathsf{aux} \leftarrow \mathsf{LearnQuery}^{\mathcal{O}}(\mathsf{sk}, \epsilon) \end{array}\right] \leq \kappa$$

Before we can prove our lemma for this section, we will prove a technical lemma which establishes that exceptional queries are almost as rare for codewords of pseudorandom codes as they are for uniformly random words. The basic idea is that, since we can determine exceptionality of a word *without using the oracle*, if exceptionality of codewords was noticeably different than it is for random words, we would obtain a distinguisher (without using the oracle), contradicting pseudorandomness. In the following, we will use $\mathbf{EXC}_{\mathbf{sk},\epsilon,\delta}(\mathbf{x})$ as a shorthand for $\mathsf{EXC}_{\mathsf{Quer}_{\mathsf{sk}},\epsilon,\delta}(\mathbf{x})$

Lemma 4. *Let* $\mathsf{PRC}^{\mathcal{O}} = (\mathsf{KeyGen}, \mathsf{Encode}^{\mathcal{O}}, \mathsf{Decode}^{\mathcal{O}}, \mathsf{LearnQuery}^{\mathcal{O}}, \mathsf{ADecode}^{\mathcal{O}})$ *be an augmented pseudorandom code. Fix a query function* $\mathsf{Quer}_{\mathsf{sk}'} : \{0,1\}^n \times \{0,1\}^r \to Q$ *which depends on an augmented key* $\mathsf{sk}' = (\mathsf{sk}, \mathsf{aux})$*. Let* $p^* = p^*(\epsilon, \delta) = \frac{\epsilon^{(1-\rho^2)/(1+\rho^2)}}{\delta^2}$*. Assume that* $\mathsf{sk}' = (\mathsf{sk}, \mathsf{aux})$*, where* $\mathsf{sk}' \in \{0,1\}^k$ *and* $\mathbf{c}$ *are generated as in the experiment described in Definition 9. If* $\Pr[\mathsf{EXC}_{\mathsf{sk}',\epsilon,\delta}(\mathbf{c})] > 3p^*$*, then there exists a distinguisher* $\mathcal{D}$ *distinguishing codewords of* $\mathsf{PRC}^{\mathcal{O}}$ *from the uniform distribution on* $\{0,1\}^n$ *with advantage* $\frac{1}{2}p^* - 2^{-\lambda}$ *given* $N = (k+\lambda)/p^*$ *samples, but neither* sk *nor access to* $\mathcal{O}$*. Equivalently, if* $\mathsf{PRC}^{\mathcal{O}}$ *is pseudorandom, then* $\Pr[\mathsf{EXC}_{\mathsf{sk}',\epsilon,\delta}(\mathbf{c})] \leq 3p^*$*.*

Note that the distinguisher $\mathcal{D}$ does not know the secret key sk. Further note that due to the oracle $\mathcal{O}$, codewords of $\mathsf{PRC}^{\mathcal{O}}$ may be arbitrarily correlated.

Proof. The distinguisher $\mathcal{D}$ proceeds as follows, given N samples $\mathbf{x}_1, \ldots, \mathbf{x}_N$ obtained from its encoding oracle. If there exists an $\mathsf{sk} \in \{0,1\}^k$ such that $\frac{1}{N}\sum_{i=1}^N \mathsf{EXC}_{\mathsf{sk},\epsilon,\delta}(\mathbf{x}_i) > 2p^*$ guess 1 with the meaning that the samples $\mathbf{x}_1, \ldots, \mathbf{x}_N$ are codewords of $\mathsf{PRC}^{\mathcal{O}}$. Otherwise output a uniformly random bit.

We will first show that if the $\mathbf{x}_1, \ldots, \mathbf{x}_N$ are chosen uniformly random, then $\mathcal{D}$ outputs a uniformly random bit, except with negligible probability $2^{-\lambda}$. Fix an $\mathsf{sk}^* \in \{0,1\}^k$ and let $\mathbf{u}_1, \ldots, \mathbf{u}_N \leftarrow \{0,1\}^n$ be chosen i.i.d. uniformly random. Then it holds that

$$\begin{aligned}
\mathsf{E}_{\mathbf{u}_1,\ldots,\mathbf{u}_N}\left[\sum_{i=1}^N \mathsf{EXC}_{\mathsf{sk}^*,\epsilon,\delta}(\mathbf{u}_i)\right] &= \sum_{i=1}^N \mathsf{E}_{\mathbf{u}_i}[\mathsf{EXC}_{\mathsf{sk}^*,\epsilon,\delta}(\mathbf{u}_i)] \\
&= \sum_{i=1}^N \underbrace{\Pr_{\mathbf{u}_i}[\mathsf{EXC}_{\mathsf{sk}^*,\epsilon,\delta}(\mathbf{u}_i)]}_{=p^*} \\
&= N \cdot p^*.
\end{aligned}$$

Hence, by the Chernoff bound we obtain that

$$\begin{aligned}\Pr_{\mathbf{u}_1,\ldots,\mathbf{u}_N}[\frac{1}{N}\sum_{i=1}^{N}\mathsf{EXC}_{\mathsf{sk}^*,\epsilon,\delta}(\mathbf{u}_i) > 2\cdot p^*] &= \Pr_{\mathbf{u}_1,\ldots,\mathbf{u}_N}[\sum_{i=1}^{N}\mathsf{EXC}_{\mathsf{sk}^*,\epsilon,\delta}(\mathbf{u}_i) > 2\cdot p^*\cdot N]\\ &\leq e^{-4/3\cdot p^*\cdot N}\\ &\leq 2^{-p^*\cdot N}\end{aligned}$$

Hence, via a union-bound we obtain

$$\begin{aligned}&\Pr[\exists \mathsf{sk}^* \in \{0,1\}^k : \frac{1}{N}\sum_{i=1}^{N}\mathsf{EXC}_{\mathsf{sk}^*,\epsilon,\delta}(\mathbf{u}_i) > 2\cdot p^*]\\ &\leq \sum_{\mathsf{sk}^*\in\{0,1\}^k}\Pr_{\mathbf{u}_1,\ldots,\mathbf{u}_N}[\frac{1}{N}\sum_{i=1}^{N}\mathsf{EXC}_{\mathsf{sk}^*,\epsilon,\delta}(\mathbf{u}_i) > 2\cdot p^*]\\ &\leq 2^k\cdot 2^{-p^*\cdot N} = 2^{k-p^*\cdot N}.\end{aligned}$$

Hence, choosing $N \geq (k+\lambda)/p^*$ we get that on input $\mathbf{u}_1,\ldots,\mathbf{u}_N$, $\mathcal{D}$ outputs a uniformly random bit, except with negligible probability $2^{-\lambda}$.

Now assume that $\mathbf{c}_1,\ldots,\mathbf{c}_N$ were generated by $\mathsf{PRC}^{\mathcal{O}}$. Define that random variable

$$Z = \frac{1}{N}\sum_{i=1}^{N}\mathsf{EXC}_{\mathsf{sk}',\epsilon,\delta}(\mathbf{c}_i).$$

First note that $Z \in [0,1]$ as for all $i \in [N]$ we have $\mathsf{EXC}_{\mathsf{sk}',\epsilon,\delta}(\mathbf{c}_i) \in \{0,1\}$. Furthermore, it holds that

$$\mathsf{E}_{\mathsf{sk}',\mathcal{O},\mathbf{c}_1,\ldots,\mathbf{c}_n}[Z] = \frac{1}{N}\sum_{i=1}^{N}\mathsf{E}_{\mathsf{sk}',\mathcal{O},\mathbf{c}_i}[\mathsf{EXC}_{\mathsf{sk}',\epsilon,\delta}(\mathbf{c}_i)] = \frac{1}{N}\sum_{i=1}^{N}\underbrace{\Pr_{\mathsf{sk}',\mathcal{O},\mathbf{c}_i}[\mathsf{EXC}_{\mathsf{sk}',\epsilon,\delta}(\mathbf{c}_i)]}_{>3p^*} > 3p^*.$$

Hence it holds by Lemma 1 that $\Pr[Z > 2p^*] \geq \mathsf{E}[Z] - 2p^* > p^*$. It follows that

$$\Pr[Z > 2p^*] > p^*.$$

Hence we have that

$$\begin{aligned}&\Pr[\mathcal{D}((\mathbf{c}_i)_i) = 1]\\ &= \underbrace{\Pr[\mathcal{D}((\mathbf{c}_i)_i) = 1|Z > 2p^*]}_{=1}\Pr[Z > 2p^*] + \underbrace{\Pr[\mathcal{D}((\mathbf{c}_i)_i) = 1|Z \leq 2p^*]}_{=1/2}\Pr[Z \leq 2p^*]\\ &= \Pr[Z > 2p^*] + \frac{1}{2}(1 - \Pr[Z > 2p^*])\\ &= \frac{1}{2} + \frac{1}{2}\Pr[Z > 2p^*]\\ &> \frac{1}{2} + \frac{1}{2}p^*.\end{aligned}$$

We conclude that our distinguisher has advantage $\frac{1}{2}p^* - 2^{-\lambda}$.

Theorem 9. *Fix the oracle $\mathcal{O} : \{0,1\}^* \to \{0,1\}^*$. Let $\mathsf{PRC}^{\mathcal{O}} = (\mathsf{KeyGen}, \mathsf{Encode}^{\mathcal{O}}, \mathsf{Decode}^{\mathcal{O}})$ be a pseudorandom code with κ-correctness and ρ-robustness. Then, there exists an (unbounded) oracle algorithm $\mathsf{LearnQuery}^{\mathcal{O}}$ and an efficient deterministic decoder Decode' such that for all $\lambda \in \mathbb{N}$, $(\mathsf{KeyGen}, \mathsf{LearnQuery}^{\mathcal{O}}, \mathsf{Encode}^{\mathcal{O}}, \mathsf{Decode}')$ is a PRC with $\kappa + 1/p(\lambda)$-augmented correctness and ρ-robustness for any polynomial $p(\cdot)$.*

We will prove Theorem 9 via the following Lemma, which enables us to remove oracle queries iteratively.

Lemma 5. *Let* $\mathsf{PRC}^{\mathcal{O}} = (\mathsf{KeyGen}, \mathsf{Encode}^{\mathcal{O}}, \mathsf{Decode}^{\mathcal{O}})$ *be a PRC which has an augmented decoder* $(\mathsf{LearnQuery}^{\mathcal{O}}, \mathsf{ADecode}^{\mathcal{O}})$ *with correctness error* κ *and* ρ *robustness relative to an oracle* $\mathcal{O}$*. Let* $\mathsf{S} = (\mathsf{S}_1, \mathsf{S}_2^{\mathcal{O}})$ *be a simulator relative to the oracle* $\mathcal{O}$*. Assume that* $\mathsf{ADecode}$ *makes at most* ℓ $\mathcal{O}$*-queries and* KeyGen *makes no oracle queries. Then for every* $\kappa' = 1/\mathsf{poly}(\lambda)$*, there exists an augmented decoder* $(\mathsf{LearnQuery}'^{\mathcal{O}}, \mathsf{ADecode}'^{\mathcal{O}})$ *for* $\mathsf{PRC}^{\mathcal{O}}$ *with the following properties.*

1. $\mathsf{ADecode}'$ *makes* $(\ell - 1)$ $\mathcal{O}$*-queries.*
2. *For all* $\lambda \in \mathbb{N}$, $(\mathsf{KeyGen}, \mathsf{LearnQuery}'^{\mathcal{O}}, \mathsf{Encode}^{\mathcal{O}}, \mathsf{ADecode}'^{\mathcal{O}})$ *is a PRC with* $\kappa + \kappa'$*-augmented correctness and* ρ*-robustness.*
3. *The number of* $\mathcal{O}$*-queries made by* $\mathsf{LearnQuery}'$ *is at most the sum of the number of* $\mathcal{O}$*-queries made by* $\mathsf{LearnQuery}$ *and* $1/\epsilon$ *where* ϵ *only depends on* κ' *and* ρ*.*
4. *The size of the output of* $\mathsf{LearnQuery}'$ *is atmost* $\mathsf{poly}(\lambda)/\epsilon$*.*

Proof. We start by describing the $\mathsf{LearnQuery}'$ and $\mathsf{ADecode}'$ algorithms.

Description of $\mathsf{LearnQuery}'$*:* On input (sk, ϵ), $\mathsf{LearnQuery}'$ proceeds as follows:

1. Run $\mathsf{aux} \leftarrow \mathsf{LearnQuery}^{\mathcal{O}}(\mathsf{sk}, \epsilon)$.
2. Set $\mathsf{sk}' = (\mathsf{sk}, \mathsf{aux})$ and let $\mathsf{Quer}_{\mathsf{sk}'} : \{0,1\}^n \times \{0,1\}^r \to Q$ be the function that computes the *first* $\mathcal{O}$-query of $\mathsf{ADecode}^{\mathcal{O}}(\mathsf{sk}, \mathsf{aux}, \cdot)$. Define the global weight function $\mathsf{w}(\cdot)$ with respect to $\mathsf{Quer}_{\mathsf{sk}'}$.
3. Let $\tilde{Q} = \{q \in Q : \mathsf{w}(q) \geq \epsilon\}$ be the set of all globally ϵ-heavy queries. Query the oracle $\mathcal{O}$ on all $q \in \tilde{Q}$ and store all query-response pairs in a list $\mathcal{L}$.
4. Define $\mathsf{aux}' = \mathsf{aux} \| \mathcal{L}$ and output aux'.

Description of $\mathsf{ADecode}'$*:* On input $(\mathsf{sk}, \mathsf{aux}', \mathbf{c})$, where aux is a list of query-response pairs. $\mathsf{ADecode}'$ does the following

1. Parse $\mathsf{aux}' = \mathsf{aux} \| \mathcal{L}$.
2. Run $\mathsf{ADecode}^{\mathcal{O}}(\mathsf{sk}, \mathsf{aux}, \mathbf{c})$ until it makes the first query q, check if the query q is in $\mathcal{L}$. If so reply with the corresponding stored response, if not then run the simulator $(y, \mathsf{st}) \leftarrow \mathsf{S}_1(q)$ and reply y.
3. If the response to the query q was taken from $\mathcal{L}$, answer the remaining $\mathcal{O}$-queries of $\mathsf{ADecode}$ normally using $\mathcal{O}$. Otherwise, if the response to q was simulated by the first simulator, i.e., $(y, \mathsf{st}) \leftarrow \mathsf{S}_1(q)$, answer the remaining queries using $\mathsf{S}_2^{\mathcal{O}}(\mathsf{st})$.
4. Output the final output of $\mathsf{ADecode}$.

Correctness of $\mathsf{ADecode}'$ *- From Exceptionality to Intersection Queries:* We will now analyze the correctness error of the augmented decoder $(\mathsf{LearnQuery}', \mathsf{ADecode}')$. First, note that the size of the set $\tilde{Q}$ is at most $1/\epsilon$. As all $q \in \tilde{Q}$ are ϵ-heavy it holds that

$$|\tilde{Q}| \cdot \epsilon \leq \sum_{q \in \tilde{Q}} \mathsf{w}(q) = \sum_{q \in \tilde{Q}} \Pr[\mathsf{Quer}_{\mathsf{sk}'}(\mathbf{u}, r) = q] = \Pr[\mathsf{Quer}_{\mathsf{sk}'}(\mathbf{u}, r) \in \tilde{Q}] \leq 1.$$

It follows that $|\tilde{Q}| \leq 1/\epsilon$. Hence, the learner $\mathsf{LearnQuery}'$ makes at most $1/\epsilon$ oracle queries and thus satisfies the efficiency requirement.

Now consider the correctness experiments of $(\mathsf{LearnQuery}, \mathsf{ADecode})$ and $(\mathsf{LearnQuery}', \mathsf{ADecode}')$. In both experiments:

– Sample an oracle $\mathcal{O}$

– Run $\mathsf{sk} \leftarrow \mathsf{KeyGen}(1^\lambda)$
– $m \leftarrow \{0,1\}^d$
– $\mathbf{c} \leftarrow \mathsf{Encode}^{\mathcal{O}}(\mathsf{sk}, m)$
– $\mathbf{e} \leftarrow \mathsf{Ber}_\rho^n$

In $(\mathsf{LearnQuery}, \mathsf{ADecode})$ we then compute

– $\mathsf{aux} \leftarrow \mathsf{LearnQuery}^{\mathcal{O}}(\mathsf{sk}, \epsilon)$
– If $\mathsf{ADecode}^{\mathcal{O}}(\mathsf{sk}, \mathsf{aux}, \mathbf{c}+\mathbf{e}) = m$ output 1, otherwise 0

Whereas in $(\mathsf{LearnQuery}', \mathsf{ADecode}')$ we continue by

– $\mathsf{aux}' \leftarrow \mathsf{LearnQuery}'^{\mathcal{O}}(\mathsf{sk}, \epsilon)$
– If $\mathsf{ADecode}'^{\mathcal{O}}(\mathsf{sk}, \mathsf{aux}, \mathbf{c}+\mathbf{e}) = m$ output 1, otherwise 0

Now let Z_{Encode} be the set of query-response pairs (q_i, y_i) made and obtained by $\mathsf{Encode}^{\mathcal{O}}(\mathsf{sk}, m)$, i.e. $\mathsf{Encode}^{\mathcal{O}}(\mathsf{sk}, m)$ has made queries q_i to $\mathcal{O}$ and obtained the responses $y_i = \mathcal{O}(q_i)$. By the locality property of $\mathcal{O}$, the size of the set $Q_{rel} \subset Q$ of queries *related* to Z_{Encode} is at most $|Q_{rel}| \leq f(|Z_{\mathsf{Encode}}|)$ for some fixed polynomial f which only depends on $\mathcal{O}$. Consequently, as $|Z_{\mathsf{Encode}}| = \mathsf{poly}(\lambda)$ we obtain that $M = |Q_{rel}| = \mathsf{poly}(\lambda)$.

Now let $q = \mathbf{Quer}_{sk'}(\mathbf{c}+\mathbf{e})$ and consider the following 3 cases:

1. q has weight $\mathsf{w}(q) \geq \epsilon$. In this case $\mathsf{LearnQuery}'^{\mathcal{O}}$ queries q, consequently a corresponding query-response pair for q is in the list $\mathcal{L}$. All further queries are answered using $\mathcal{O}$. Hence $\mathsf{ADecode}'$ simulates $\mathsf{ADecode}$ perfectly, i.e. their outputs are identically distributed.
2. q has weight $\mathsf{w}(q) < \epsilon$ but $q \notin Q_{rel}$. In this case $\mathsf{ADecode}'$ simulates the response y to query q via $(y, \mathsf{st}) \leftarrow \mathsf{S}_1(q)$. All further queries are answered via $\mathsf{S}_2^{\mathcal{O}}(\mathsf{st})$. Hence, as $q \notin Q_{rel}$ by the simulatability property of the local oracle $\mathcal{O}$ it holds that the output distribution of $\mathsf{ADecode}'$ is statistically close to that of $\mathsf{ADecode}$.
3. q has weight $\mathsf{w}(q) < \epsilon$ and $q \in Q_{rel}$. Such queries are problematic and cannot be answered consistently by $\mathsf{ADecode}'$. Call this event BAD.

By the above discussion, conditioned on $\neg\mathsf{BAD}$ the outputs of the correctness experiment of $(\mathsf{LearnQuery}', \mathsf{ADecode}')$ is identically distributed to the output of the correctness experiment of $(\mathsf{LearnQuery}, \mathsf{ADecode})$.

Thus we need to show that case 3, i.e. the event BAD only occurs with small probability. For this purpose recall the notion of exceptional queries (Definition 8). $\mathsf{EXC}_{\mathsf{Quer}_{\mathsf{sk}'},\epsilon,\delta}(\mathbf{c})$ holds, if and only if there exists a query $q \in Q$ such that $\mathsf{w}(q) < \epsilon$ and $\ell_{\mathbf{c}}(q) \geq \delta$ (both with respect to the query function $\mathsf{Quer}_{\mathsf{sk}'}$). Lemma 4 and Theorem 8 (with $t = 1$) establish that if $\mathbf{c} \leftarrow \mathsf{Encode}^{\mathcal{O}}(\mathsf{sk}', m)$ as above, then

$$\Pr[\mathsf{EXC}_{\mathsf{Quer}_{\mathsf{sk}'},\epsilon,\delta}(\mathbf{c})] \leq 3p^*(\epsilon,\delta) = 3 \cdot \frac{\epsilon^{(1-\rho^2)/(1+\rho^2)}}{\delta^2}.$$

$$\begin{aligned}\Pr[\mathsf{BAD}] &= \Pr[\mathsf{BAD}|\neg\mathsf{EXC}_{\mathsf{sk}'}(\mathbf{c})] \underbrace{\Pr[\neg\mathsf{EXC}_{\mathsf{sk}'}(\mathbf{c})]}_{\leq 1} + \underbrace{\Pr[\mathsf{BAD}|\mathsf{EXC}_{\mathsf{sk}'}(\mathbf{c})]}_{\leq 1} \underbrace{\Pr[\mathsf{EXC}_{\mathsf{sk}'}(\mathbf{c})]}_{\leq 3p^*} \\ &\leq \Pr[\mathsf{BAD}|\neg\mathsf{EXC}_{\mathsf{sk}'}(\mathbf{c})] + 3p^*.\end{aligned}$$

It remains to bound $\Pr[\mathsf{BAD}|\neg\mathsf{EXC}_{\mathsf{sk}'}(\mathbf{c})]$. Hence fix any sk' and $\mathbf{c}$ with $\mathsf{EXC}_{\mathsf{sk}',\epsilon,\delta}(\mathbf{c}) = 0$, i.e. it holds for all $q \in Q$ with $\mathsf{w}(q) < \epsilon$ that $\ell_{\mathbf{c}}(q) < \delta$, that is $\Pr_{\mathbf{e}}[\mathsf{Quer}_{\mathsf{sk}'}(\mathbf{c}+\mathbf{e}) = q] < \delta$. Note that this last probability only depends on $\mathbf{e}$. In the following fix any set Q_{rel} with $|Q_{rel}| \leq M$.

Then it holds that

$$\begin{aligned}\Pr[\mathsf{BAD}] &= \Pr[\exists q \in Q_{rel} : \mathsf{Quer}_{\mathsf{sk}'}(\mathbf{c}+\mathbf{e}) = q \text{ and } \mathsf{w}(q) < \epsilon] \\ &\leq \sum_{\substack{q \in Q_{rel} \\ \mathsf{w}(g) < \epsilon}} \underbrace{\Pr[\mathsf{Quer}_{\mathsf{sk}'}(\mathbf{c}+\mathbf{e}) = q]}_{=\ell_{\mathbf{c}}(q) < \delta} \\ &< M \cdot \delta.\end{aligned}$$

We conclude that

$$\Pr[\mathsf{BAD}] < M \cdot \delta + 3p^* = M \cdot \delta + 3 \cdot \frac{\epsilon^{(1-\rho^2)/(1+\rho^2)}}{\delta^2}.$$

Hence, to limit the additional correctness error of $(\mathsf{LearnQuery}'^{\mathcal{O}}, \mathsf{ADecode})$ (which occurs exactly when BAD happens) to a value κ^* we can choose

- $\delta = \frac{\kappa^*}{2 \cdot M}$
- $\epsilon = \left(\frac{\kappa^* \cdot \delta^2}{6}\right)^{(1+\rho^2)/(1-\rho^2)}$

which routinely yields $\Pr[\mathsf{BAD}] < \kappa^*$. Note that both δ and ϵ are polynomial in κ^* and inverse polynomial in M. Finally, notice that we can absorb the negligible correctness error resulting from Case 2 into κ' by e.g. choosing $\kappa' = \kappa^*/2$. This concludes the proof.

Proof (*Proof of Theorem 9*). Let the number of oracle queries made by $\mathsf{Decode}^{\mathcal{O}}(\mathsf{sk}, \cdot)$ be bounded by ℓ. We begin by applying Lemma 5 to the original decoding algorithm $\mathsf{Decode}^{\mathcal{O}}$. This yields a new decoding algorithm $\mathsf{ADecode}^{\mathcal{O}}$ that makes one fewer oracle query than $\mathsf{Decode}^{\mathcal{O}}$, while maintaining correctness up to an additive error of κ'. Additionally, the size of the auxiliary information required is at most $\mathsf{poly}(\lambda)/\epsilon$ because there are at most ϵ^{-1} many ϵ-global heavy queries and each query-response pair is of size $\mathsf{poly}(\lambda)$.

We apply Lemma 5 iteratively ℓ times, eventually obtaining a decoder Decode' that makes no oracle queries. Instead, all oracle responses are simulated using the auxiliary information aux.

The total size of aux is at most $\ell \cdot \mathsf{poly}(\lambda)/\epsilon$. Since Decode' simulates Decode by answering oracle queries from aux, its running time increases by at most a $\mathsf{poly}(\epsilon^{-1}, \lambda)$ factor over that of Decode. The total degradation in correctness is additive, resulting in an overall correctness error of at most $\kappa + (\ell \cdot \kappa')$.

By setting $\kappa' = 1/(\ell \cdot p(\lambda))$ where p is polynomial in the theorem statement, we obtain the necessary efficiency and probability bound. This is because ℓ and M are polynomials in λ and ϵ depends on κ', κ and ρ.

5.4 Breaking Pseudorandomness with the Oracle-Free Decoder

Theorem 10. *Let* $\mathsf{PRC}^{\mathcal{O}} = (\mathsf{KeyGen}, \mathsf{Encode}^{\mathcal{O}}, \mathsf{Decode}^{\mathcal{O}})$ *be a PRC relative to a local oracle* $\mathcal{O}$. *Let* $\mathsf{LearnQuery}^{\mathcal{O}}$ *be a unbounded algorithm and* $\mathsf{ADecode}$ *be a decoder such that the correctness error with respect to the augmented scheme* $(\mathsf{KeyGen}, \mathsf{LearnQuery}^{\mathcal{O}}, \mathsf{Encode}^{\mathcal{O}}, \mathsf{ADecode})$ *is* κ *where* $\kappa < 1/8$. *Then, there exists an unbounded distinguisher* $\mathcal{D}$ *that breaks the pseudorandomness property of* $\mathsf{PRC}^{\mathcal{O}}$ *with advantage* $\frac{1}{16} - 2^{-\lambda}$.

Proof. Assume in the following the augmented secret key sk' for $\mathsf{ADecode}$ has bitlength k. Let $M = 8(k + \lambda)$.

Consider the following distinguisher $\mathcal{D}$, which has access to an encoding oracle but does not require additional access to the oracle $\mathcal{O}$:

– Query the encoding oracle on M uniformly random bits $b_1, \ldots, b_M \leftarrow \{0,1\}$ and obtain words $\mathbf{x}_1, \ldots, \mathbf{x}_M$.
– Choose Bernoulli errors $\mathbf{e}_1, \ldots, \mathbf{e}_M \leftarrow \mathsf{Ber}_\rho^n$
– If there exists a string $\mathsf{sk}' \in \{0,1\}^k$ such that $Z(\mathsf{sk}') = \frac{1}{M}\sum_{i=1}^M z_i(\mathsf{sk}') > 3/4$ output 1, otherwise output a uniformly random bit. Here, we define the binary random variables $z_i(\mathsf{sk}')$ to be 1 if $\mathsf{ADecode}(\mathsf{sk}', \mathbf{x}_i + \mathbf{e}_i) = b_i$ and otherwise 0.

We will now analyze the advantage of this distinguisher. Assume first that the oracle provided to $\mathcal{D}$ produces uniformly random strings, i.e. for all $i \in [M]$ we have $\mathbf{x}_i = \mathbf{u}_i$ for uniformly random $\mathbf{u}_i \leftarrow \{0,1\}^n$.

Fix a secret key sk'. Since the $\mathbf{u}_i$ are independent of the b_i, and therefore, for each $i \in [M]$ that $\Pr[z_i(\mathsf{sk}') = 1] = \Pr[\mathsf{ADecode}(\mathsf{sk}', \mathbf{x}_i + \mathbf{e}_i) = b_i] = 1/2$, as the b_i are chosen uniformly random from $\{0,1\}$. Furthermore, as the b_i are independent, so are the $z_i(\mathsf{sk}')$. Consequently, it holds that

$$\mathsf{E}[Z(\mathsf{sk}')] = \mathsf{E}\left[\frac{1}{M}\sum_{i=1}^M z_i(\mathsf{sk}')\right] = \frac{1}{M}\sum_{i=1}^M \underbrace{\mathsf{E}[z_i(\mathsf{sk}')]}_{=\Pr[z_i(\mathsf{sk}')=1]} = \frac{1}{M}\cdot\frac{1}{2}\cdot M = \frac{1}{2}.$$

Consequently, it holds by a Chernoff bound that

$$\Pr[Z(\mathsf{sk}') > 3/4] \leq e^{-\frac{1}{8}M} \leq 2^{-M/8} = 2^{-k-\lambda}.$$

By a union-bound, it holds that

$$\Pr[\exists \mathsf{sk}' \in \{0,1\}^k \text{ s.t. } Z(\mathsf{sk}') > 3/4] \leq 2^k \cdot 2^{-k-\lambda} = 2^{-\lambda}.$$

Thus, we conclude that in this case $\mathcal{D}$ outputs a uniformly random bit, except with negligible probability $2^{-\lambda}$.

On the other hand, assume that the oracle provided to $\mathcal{D}$ outputs well-formed codewords $\mathbf{c}_1, \ldots, \mathbf{c}_M$ encoding the bits $b_1, \ldots, b_M$. In this case it holds that

$$\mathsf{E}[Z(\mathsf{sk})] = \frac{1}{M}\sum_{i=1}^M \mathsf{E}[z_i(\mathsf{sk})] = \frac{1}{M}\sum_{i=1}^M \underbrace{\Pr[\mathsf{ADecode}(\mathsf{sk}, \mathbf{c}_i + \mathbf{e}_i) = b_i]}_{\geq 1-\kappa} = 1 - \kappa > 7/8.$$

over all random choices, including sk. Noting that $Z \in [0,1]$ it follows from Lemma 1 that $\Pr[Z > 3/4] \geq \mathsf{E}[Z] - 3/4 \geq 1/8$. Note that if there exists an sk such that with $Z(\mathsf{sk}) > 3/4$, then $\mathcal{D}$ will always output 1. Thus we get

$$\begin{aligned}
&\Pr[\mathcal{D}^{\mathsf{Encode}} = 1] \\
&= \underbrace{\Pr[\mathcal{D}^{\mathsf{Encode}} = 1 | Z > 3/4]}_{=1}\Pr[Z > 3/4] + \underbrace{\Pr[\mathcal{D}^{\mathsf{Encode}} = 1 | Z \leq 3/4]}_{=1/2}\Pr[Z \leq 3/4] \\
&= \Pr[Z > 3/4] + \frac{1}{2}(1 - \Pr[Z > 3/4]) \\
&= \frac{1}{2} + \frac{1}{2}\Pr[Z > 3/4] \\
&> \frac{1}{2} + \frac{1}{16}.
\end{aligned}$$

Hence we conclude that our distinguisher has advantage $\frac{1}{16} - 2^{-\lambda}$.

6 Construction of Large-Alphabet Pseudorandom Codes from Minimal Assumptions

In this section, we present a construction for error-correcting codes from a PRP and a SKE, which in turn can be based on the minimal assumption of one-way functions. The construction avoids the impossibility result by working over a large alphabet. For error-correcting codes that work over a large alphabet, errors are introduced per symbol[10] instead of per bit.

6.1 Construction

Consider the following parameters and primitives.

- Let $q = 2^\lambda$.
- Let $\mathsf{SKE} = (\mathsf{SKE.KeyGen}, \mathsf{SKE.Enc}, \mathsf{SKE.Dec})$ be a secret-key encryption scheme that encrypts messages of any length ℓ and generates pseudorandom ciphertexts of size $\mathbb{F}_q^k$ where $k = \mathsf{poly}(\lambda, \ell)$.
- Let $\mathsf{RS} = (\mathsf{Encode}, \mathsf{Decode})$ be a $[n, k, n-k+1]$ Reed-Solomon Code.
- Let $\mathsf{PRP} = (\mathsf{PRP.KeyGen}, \mathsf{PRP.Eval})$ be a pseudorandom permutation over the space $\mathbb{F}_q^n$.

Our construction is as follows.

$\mathsf{KeyGen}(1^\lambda) \to \mathsf{sk}$: Run $\mathsf{SKE.KeyGen}(1^\lambda) \to \mathsf{sk}_1$ and $\mathsf{PRP.KeyGen}(1^\lambda) = \mathsf{sk}_2$. Output $\mathsf{sk} = (\mathsf{sk}_1, \mathsf{sk}_2)$.

$\mathsf{Encode}(m, \mathsf{sk} = (\mathsf{sk}_1, \mathsf{sk}_2)) \to \mathsf{c}$: Encrypt the message $\mathsf{SKE.Enc}(\mathsf{sk}_1, m) \to \mathsf{ct}$. Encode ct using the Reed-Solomon Code to produce $\mathsf{Encode}(\mathsf{ct}) \to \tilde{\mathsf{c}}$. Parse $\tilde{\mathsf{c}}$ as n symbols in $\mathbb{F}_q$ and apply the pseudorandom permutation to each $i \in [n]$, i.e., $\mathsf{PRP.Eval}(\mathsf{sk}_2, \tilde{\mathsf{c}}_i) = \mathsf{c}_i$. Output $\mathsf{c} = (\mathsf{c}_1, \ldots, \mathsf{c}_n)$.

$\mathsf{Decode}(\mathsf{c}, \mathsf{sk} = (\mathsf{sk}_1, \mathsf{sk}_2)) \to m/\perp$: Parse c as n symbols in $\mathbb{F}_q$ and apply the inverse pseudorandom permutation to each element: $\mathsf{PRP.Eval}(\mathsf{sk}_2, \mathsf{c}_i) = \tilde{\mathsf{c}}_i$. Decode $\tilde{\mathsf{c}} = (\tilde{\mathsf{c}}_1, \ldots, \tilde{\mathsf{c}}_n)$ using the Reed-Solomon Decoder $\mathsf{Decode}(\tilde{\mathsf{c}}) \to \mathsf{ct}$. If the decoding was unsuccessful, output $\perp$. Otherwise decrypt the ciphertext $\mathsf{SKE.Dec}(\mathsf{ct}, \mathsf{sk}_1) \to m$ and output m.

Perfect Correctness. Since a $[n, k, n-k+1]$-Reed-Solomon code achieves perfect robustness against errors below the error threshold $t = (n-k)/(2n)$, our construction achieves perfect robustness up to tn errors. The construction inherits correctness from the correctness of the SKE scheme, the Reed-Solomon code and the pseudorandom permutation. Assuming that the message is ℓ' elements of $\mathbb{F}_q$, the transmission rate of the scheme is $r = \frac{\ell'}{n}$.

Theorem 11. *Let* SKE *be a* \$*-CPA secure secret-key encryption scheme and let* PRP *be a secure pseudorandom permutation and let* RS *be a Reed-Solomon Code, then the above construction is pseudorandom.*

Proof. We show pseudorandomness by a series of hybrid arguments.

- **Hybrid 0**: This is the original construction.
- **Hybrid 1**: In this hybrid, the SKE ciphertexts generated in Encode are replaced with truly random strings.
- **Hybrid 2**: In this hybrid, the outputs of the PRP are replaced by uniformly random values. This is equivalent to sampling the codeword uniformly at random.

[10] An alphabet is a finite set of symbols

Analysis: Let $p_{\mathcal{A},i}$ denote probability of $\mathcal{A}$ outputting 1 in Game G_i. We will show that this probability is almost the same in every hybrid.

Lemma 6. *Assuming the pseudorandomness property of the* SKE *scheme, for all PPT adversary* $\mathcal{A}$*, there exists a negligible function* $\mathrm{negl}(\cdot)$ *such that for all* $\lambda \in \mathbb{N}$*,*

$$|p_{\mathcal{A},1} - p_{\mathcal{A},0}| \leq \mathrm{negl}(\lambda)$$

Proof. Let ℓ be the number of queries made by the adversary Adv. We consider $\ell + 1$ intermediate hybrids $\{H_{0,i}\}_{i\in[\ell]}$ where in the hybrid $H_{0,i}$, for all queries before the i^{th} query made by $\mathcal{A}$, the challenger replaces the SKE ciphertexts with truly random strings. It is easy to see that if $\mathcal{A}$ can distinguish $H_{0,i}$ from $H_{0,i-1}$, it can be turned into an attacker Adv^* against the underlying SKE scheme. Until the $(i-1)^{th}$ query, $\mathcal{A}^*$ uses truly random string while responding to the queries. In the i^{th} query, it forwards the query m to the SKE challenger. Upon receiving a challenge ciphertext c^*, the $\mathcal{A}^*$ encodes c^* with a Reed-Solomon code and applies a PRP. Then he forwards the message to Adv and outputs whatever Adv outputs. For the remaining queries, it queries the \$-CPA oracle to obtain the ciphertexts. Observe that Adv^* wins with the same probability as Adv.

Lemma 7. *Assuming the security of the pseudorandom permutation, for all PPT adversary* $\mathcal{A}$*, there exists a negligible function* $\mathrm{negl}(\cdot)$ *such that for all* $\lambda \in \mathbb{N}$*,*

$$|p_{\mathcal{A},2} - p_{\mathcal{A},1}| \leq \mathrm{negl}(\lambda)$$

Proof. If all the inputs to the PRP are distinct then, we can replace the outputs by uniformly random strings due the security of the PRP. Therefore, we need to bound the probability that any symbol in the Reed-Solomon codeword appears twice. The encoding procedure of a Reed-Solomon code takes the message and parses it as k symbols from the alphabet $\mathbb{F}_q$. Then, a polynomial is defined by using the k symbols as coefficients for a $(k-1)$-degree polynomial. The polynomial is evaluated at $n > k$ predetermined positions.

It is a well-known fact [CW79] that the outputs of a random polynomial f evaluated at k positions have k-wise independence, i.e.,

$$\Pr[f(x_1) = \ldots = f(x_k)] \leq \frac{1}{|\mathbb{F}_q|^k}$$

where f is chosen at random from all $(k-1)$-degree polynomials over $\mathbb{F}_q$. A $(k-1)$-degree polynomial can be chosen uniformly at random by choosing each coefficient uniformly at random. Since, all the SKE ciphertexts are replaced with truly random string, this is satisfied.

Observe that any k-wise independent function f for $k \geq 2$, is also a 2-wise independent function. We know that for such a function, for all x, y,

$$\Pr[f(x) = y] = \frac{1}{|\mathbb{F}_q|}$$

where the probability is over all 2-wise independent functions. Therefore, for any $x_i \neq x_j$, we get

$$\Pr[f(x_i) = f(x_j)] = \frac{1}{|\mathbb{F}_q|}$$

Now, we can use a union bound to show that over randomly chosen 2-wise independent functions $\{f_i\}_{i\in[\ell]}$ and distinct evaluation points $x_1, \ldots, x_n$,

$$\Pr[f_a(x_i) = f_b(x_j) \mid a \neq b, i \neq j] \leq \sum_{a\neq b, i\neq j} \Pr[f_a(x_i) = f_b(x_j)] = \frac{\ell^2 n^2}{|\mathbb{F}_q|}$$

This is the bound for the probability that a collision occurs between any of the symbols output by the Reed-Solomon code over all ℓ queries of the adversary. Since $|\mathbb{F}_q| = 2^\lambda$, $\ell = O(\lambda)$ and $n = O(\lambda)$ the probability of a collision is negligible.

Using the above lemmas and triangular inequality, we get $|p_{\mathcal{A},0} - p_{\mathcal{A},2}| \leq \text{negl}(\lambda)$.

References

AAC+24. Omar Alrabiah, Prabhanjan Ananth, Miranda Christ, Yevgeniy Dodis, and Sam Gunn. Ideal pseudorandom codes. *arXiv preprint arXiv:2411.05947*, 2024. 2, 4, 10

Aar22. Scott Aaronson. My ai safety lecture for ut effective altruism, 2022. *URL https://scottaaronson. blog*, 2022. 2

ACC+22. Per Austrin, Hao Chung, Kai-Min Chung, Shiuan Fu, Yao-Ting Lin, and Mohammad Mahmoody. On the impossibility of key agreements from quantum random oracles. In *Annual International Cryptology Conference*, pages 165–194. Springer, 2022. 9

Bec75. William Beckner. Inequalities in fourier analysis. *Annals of Mathematics*, 102(1):159–182, 1975. 6, 17, 18

Ber25. Lorenzo Beretta. New statistical and computational results for learning junta distributions, 2025. 10

BFKL93. Avrim Blum, Merrick Furst, Michael Kearns, and Richard J Lipton. Cryptographic primitives based on hard learning problems. In *Annual international cryptology conference*, pages 278–291. Springer, 1993. 2

BGV+23. Samuel Bouaziz, Alex B Grilo, Damien Vergnaud, Quoc-Huy Vu, et al. Towards the impossibility of quantum public key encryption with classical keys from one-way functions. *Cryptology ePrint Archive*, 2023. 9

BKR23. Andrej Bogdanov, Pravesh K Kothari, and Alon Rosen. Public-key encryption, local pseudorandom generators, and the low-degree method. In *Theory of Cryptography Conference*, pages 268–285. Springer, 2023. 2, 10

BKSY11. Zvika Brakerski, Jonathan Katz, Gil Segev, and Arkady Yerukhimovich. Limits on the power of zero-knowledge proofs in cryptographic constructions. In *Theory of Cryptography: 8th Theory of Cryptography Conference, TCC 2011, Providence, RI, USA, March 28-30, 2011. Proceedings 8*, pages 559–578. Springer, 2011. 2, 9

BM84. Manuel Blum and Silvio Micali. How to generate cryptographically strong sequences of pseudo-random bits. *SIAM Journal on Computing*, 13(4):850, 1984. 1

BMG09. Boaz Barak and Mohammad Mahmoody-Ghidary. Merkle puzzles are optimal—an o (n 2)-query attack on any key exchange from a random oracle. In *Annual International Cryptology Conference*, pages 374–390. Springer, 2009. 2, 9

Bon70. Aline Bonami. Etude des coefficients de fourier des fonctions de $l^p(g)$. In *Annales de l'institut Fourier*, volume 20, pages 335–402, 1970. 6, 17, 18

CG24. Miranda Christ and Sam Gunn. Pseudorandom error-correcting codes. In Leonid Reyzin and Douglas Stebila, editors, *Advances in Cryptology – CRYPTO 2024, Part VI*, volume 14925 of *Lecture Notes in Computer Science*, pages 325–347, Santa Barbara, CA, USA, August 18–22, 2024. Springer, Cham, Switzerland. 1, 2, 4, 10, 12

CGZ24. 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. 10

CW79. J.Lawrence Carter and Mark N. Wegman. Universal classes of hash functions. *Journal of Computer and System Sciences*, 18(2):143–154, 1979. 27

DM24. Nico Döttling and Tamer Mour. On the black-box complexity of correlation intractability. In *15th Innovations in Theoretical Computer Science Conference (ITCS 2024)*, pages 40–1. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2024. 9

FGJ+25. Jaiden Fairoze, Sanjam Garg, Somesh Jha, Saeed Mahloujifar, Mohammad Mahmoody, and Mingyuan Wang. Publicly-detectable watermarking for language models. *IACR Communications in Cryptology*, 1(4), 2025. 2, 10

FR23. Marc Fischlin and Felix Rohrbach. Searching for elfs in the cryptographic forest. In *Theory of Cryptography Conference*, pages 207–236. Springer, 2023. 2

GG24. Surendra Ghentiyala and Venkatesan Guruswami. New constructions of pseudorandom codes. *arXiv preprint arXiv:2409.07580*, 2024. 2, 4, 10

GKM+00. Yael Gertner, Sampath Kannan, Tal Malkin, Omer Reingold, and Mahesh Viswanathan. The relationship between public key encryption and oblivious transfer. In *Proceedings 41st Annual Symposium on Foundations of Computer Science*, pages 325–335. IEEE, 2000. 2

GM24. Noah Golowich and Ankur Moitra. Edit distance robust watermarks for language models. *arXiv preprint arXiv:2406.02633*, 2024. 2, 4, 10

GMR01. Yael Gertner, Tal Malkin, and Omer Reingold. On the impossibility of basing trapdoor functions on trapdoor predicates. In *Proceedings 42nd IEEE Symposium on Foundations of Computer Science*, pages 126–135. IEEE, 2001. 2

GZS25. Sam Gunn, Xuandong Zhao, and Dawn Song. An undetectable watermark for generative image models, 2025. 10

HILL99. Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28(4):1364–1396, 1999. 1

HR04. Chun-Yuan Hsiao and Leonid Reyzin. Finding collisions on a public road, or do secure hash functions need secret coins? In *Advances in Cryptology–CRYPTO 2004: 24th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 2004. Proceedings 24*, pages 92–105. Springer, 2004. 2

IR89. Russell Impagliazzo and Steven Rudich. Limits on the provable consequences of one-way permutations. In *Proceedings of the twenty-first annual ACM symposium on Theory of computing*, pages 44–61, 1989. 2, 4, 9

Kea93. Michael Kearns. Efficient noise-tolerant learning from statistical queries. In *Proceedings of the Twenty-Fifth Annual ACM Symposium on Theory of Computing*, New York, NY, USA, 1993. Association for Computing Machinery. 2

KGW+23. John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. A watermark for large language models. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 17061–17084. PMLR, 23–29 Jul 2023. 2, 10

KJGR21. Gabriel Kaptchuk, Tushar M Jois, Matthew Green, and Aviel D Rubin. Meteor: Cryptographically secure steganography for realistic distributions. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 1529–1548, 2021. 2

KKRT16. Vladimir Kolesnikov, Ranjit Kumaresan, Mike Rosulek, and Ni Trieu. Efficient batched oblivious prf with applications to private set intersection. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, CCS '16, page 818–829, New York, NY, USA, 2016. Association for Computing Machinery. 10

KTHL24. Rohith Kuditipudi, John Thickstun, Tatsunori Hashimoto, and Percy Liang. Robust distortion-free watermarks for language models. *Transactions on Machine Learning Research*, 2024. 2, 10

LLLL24. Longcheng Li, Qian Li, Xingjian Li, and Qipeng Liu. How (not) to build quantum pke in minicrypt. In *Annual International Cryptology Conference*, pages 152–183. Springer, 2024. 9

LLLL25a. Longcheng Li, Qian Li, Xingjian Li, and Qipeng Liu. Cryptomania vs minicrypt in a quantum world. *arXiv preprint arXiv:2504.05710*, 2025. 9

LLLL25b. Longcheng Li, Qian Li, Xingjian Li, and Qipeng Liu. Toward the impossibility of perfect complete quantum pke from owfs. In *16th Innovations in Theoretical Computer Science Conference (ITCS 2025)*, pages 71–1. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2025. 9

Mat14. Takahiro Matsuda. On the impossibility of basing public-coin one-way permutations on trapdoor permutations. In *Theory of Cryptography: 11th Theory of Cryptography Conference, TCC 2014, San Diego, CA, USA, February 24-26, 2014. Proceedings 11*, pages 265–290. Springer, 2014. 2

McE78. Robert J McEliece. A public-key cryptosystem based on algebraic. *Coding Thv*, 4244(1978):114–116, 1978. 2

MM11. Takahiro Matsuda and Kanta Matsuura. On black-box separations among injective one-way functions. In *Theory of Cryptography: 8th Theory of Cryptography Conference, TCC 2011, Providence, RI, USA, March 28-30, 2011. Proceedings 8*, pages 597–614. Springer, 2011. 2, 9

MP12. Mohammad Mahmoody and Rafael Pass. The curious case of non-interactive commitments–on the power of black-box vs. non-black-box use of primitives. In *Annual Cryptology Conference*, pages 701–718. Springer, 2012. 2, 9

O'D14. Ryan O'Donnell. *Analysis of boolean functions*. Cambridge University Press, 2014. 3, 18

RS60. I. S. Reed and G. Solomon. Polynomial codes over certain finite fields. *Journal of the Society for Industrial and Applied Mathematics*, 8(2):300–304, 1960. 14

Rud88. Steven Rudich. *Limits on the provable consequences of one-way functions*. University of California at Berkeley, 1988. 2, 9

Rud91. Steven Rudich. The use of interaction in public cryptosystems. extended abstract. In *Annual International Cryptology Conference*, pages 242–251. Springer, 1991. 2, 9

Sim98. Daniel R Simon. Finding collisions on a one-way street: Can secure hash functions be based on general assumptions? In *Advances in Cryptology—EUROCRYPT'98: International Conference on the Theory and Application of Cryptographic Techniques Espoo, Finland, May 31–June 4, 1998 Proceedings 17*, pages 334–345. Springer, 1998. 2, 9

Tho48. G Olof Thorin. Convexity theorems generalizing those of m. riesz and hadamard with some applications. (*No Title*), 1948. 11

Yao82. Andrew C Yao. Theory and application of trapdoor functions. In *23rd Annual Symposium on Foundations of Computer Science (SFCS 1982)*, pages 80–91. IEEE, 1982. 1

ZALW24. Xuandong Zhao, Prabhanjan Vijendra Ananth, Lei Li, and Yu-Xiang Wang. Provable robust watermarking for ai-generated text. In *The Twelfth International Conference on Learning Representations*, 2024. 2, 10

ZDR19. Zachary M Ziegler, Yuntian Deng, and Alexander M Rush. Neural linguistic steganography. *arXiv preprint arXiv:1909.01496*, 2019. 2

ZEF+24. Hanlin Zhang, Benjamin L. Edelman, Danilo Francati, Daniele Venturi, Giuseppe Ateniese, and Boaz Barak. Watermarks in the sand: Impossibility of strong watermarking for language models. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 58851–58880. PMLR, 21–27 Jul 2024. 10

ZGC+24. Xuandong Zhao, Sam Gunn, Miranda Christ, Jaiden Fairoze, Andres Fabrega, Nicholas Carlini, Sanjam Garg, Sanghyun Hong, Milad Nasr, Florian Tramer, et al. Sok: Watermarking for ai-generated content. *arXiv preprint arXiv:2411.18479*, 2024. 10