Computational Fuzzy Extractors

Benjamin Fuller* Xianrui Meng† Leonid Reyzin‡
Boston University

June 24, 2013

Abstract

Fuzzy extractors derive strong keys from noisy sources. Their security is defined information-theoretically, which limits the length of the derived key, sometimes making it too short to be useful. We ask whether it is possible to obtain longer keys by considering computational security, and show the following.

- **Negative Result:** Noise tolerance in fuzzy extractors is usually achieved using an information reconciliation component called a "secure sketch." The security of this component, which directly affects the length of the resulting key, is subject to lower bounds from coding theory. We show that, even when defined computationally, secure sketches are still subject to lower bounds from coding theory. Specifically, we consider two computational relaxations of the information-theoretic security requirement of secure sketches, using conditional HILL entropy and unpredictability entropy. For both cases we show that computational secure sketches cannot outperform the best information-theoretic secure sketches in the case of high-entropy Hamming metric sources.
- **Positive Result:** We show that the negative result can be overcome by analyzing computational fuzzy extractors directly. Namely, we show how to build a computational fuzzy extractor whose output key length equals the entropy of the source (this is impossible in the information-theoretic setting). Our construction is based on the hardness of the Learning with Errors (LWE) problem, and is secure when the noisy source is uniform or symbol-fixing (that is, each dimension is either uniform or fixed). As part of the security proof, we show a result of independent interest, namely that the decision version of LWE is secure even when a small number of dimensions has no error.

Keywords Fuzzy extractors, secure sketches, key derivation, Learning with Errors, error-correcting codes, computational entropy, randomness extractors.

1 Introduction

Authentication generally requires a secret drawn from some high-entropy source. One of the primary building blocks for authentication is reliable key derivation. Unfortunately, many sources that contain sufficient entropy to derive a key are noisy, and provide similar, but not identical secret values at each reading (examples of such sources include biometrics [Dau04], human memory [ZH93], pictorial passwords [BS00], measurements of capacitance [TSv+06], timing [SD07], motion [CM05], quantum information [BBR88] etc.).

*Email: bfuller@cs.bu.edu. Work done in part while the author was at MIT Lincoln Laboratory.
†Email: xmeng@cs.bu.edu.
‡Email: reyzin@cs.bu.edu.

Fuzzy extractors [DORS08] achieve reliable key derivation from noisy sources (see [BDK+05, DW09, CKOR10] for applications of fuzzy extractors). The setting consists of two algorithms: Generate (used once) and Reproduce (used subsequently). The Generate (Gen) algorithm takes an input w and produces a key r and a public value p. This information allows the Reproduce (Rep) algorithm to reproduce r given p and some value w' that is close to w (according to some predefined metric, such as Hamming distance). Crucially for security, knowledge of p should not reveal r; that is, r should be uniformly distributed conditioned on p. This feature is needed because p is not secret: for example, in a single-user setting (where the user wants to reproduce the key r from a subsequent reading w'), it would be stored in the clear; and in a key agreement application [BDK+05] (where two parties have w and w', respectively), it would be transmitted between the parties.

Fuzzy extractors use ideas from information-reconciliation [BBR88] and are defined (traditionally) as information-theoretic objects. The entropy loss of a fuzzy extractor is the difference between the entropy of w and the length of the derived key r. In the information-theoretic setting, some entropy loss is necessary as the value p contains enough information to reproduce r from any close value w'. A goal of fuzzy extractor constructions is to minimize the entropy loss, increasing the security of the resulting application. Indeed, if the entropy loss is too high, the resulting secret key may be too short to be useful.

We ask whether it is possible to obtain longer keys by considering computational, rather than information theoretic, security.

Our Negative Results We first study (in Section 3) whether it could be fruitful to relax the definition of the main building block of a fuzzy extractor, called a *secure sketch*. A secure sketch is a one-round information reconciliation protocol: it produces a public value s that allows recovery of w from any close value w'. The traditional secrecy requirement of a secure sketch is that w has high min-entropy conditioned on s. This allows the fuzzy extractor of [DORS08] to form the key r by applying a randomness extractor [NZ93] to w, because randomness extractors produce random strings from strings with conditional min-entropy. We call this the *sketch-and-extract* construction.

The most natural relaxation of the min-entropy requirement of the secure sketch is to require HILL entropy [HILL99] (namely, that the distribution of w conditioned on s be *indistinguishable* from a high-min-entropy distribution). Under this definition, we could still use a randomness extractor to obtain r from w, because it would yield a pseudorandom key. Unfortunately, it is unlikely that such a relaxation will yield fruitful results: we prove in Theorem 3.6 that the entropy loss of such secure sketches is subject to the same coding bounds as the ones that constrain information-theoretic secure sketches.

Another possible relaxation is to require that the value w is unpredictable conditioned on s. This definition would also allow the use of a randomness extractor to get a pseudorandom key, although it would have to be a special extractor—one that has a reconstruction procedure (see [HLR07, Lemma 6]). Unfortunately, this relaxation is also unlikely to be fruitful: we prove in Theorem 3.10 that the unpredictability is at most log the size of the metric space minus log the volume of the ball of radius t. For high-entropy sources of w over the Hamming metric, this bound matches the best information-theoretic security sketches.

Our Positive Results Both of the above negative results arise because a secure sketch functions like a decoder of an error-correcting code. To avoid them, we give up on building computational secure sketches and focus directly on the entropy loss in fuzzy extractors. Our goal is to decrease the entropy loss in a fuzzy extractor by allowing the key r to be pseudorandom conditioned on p.

By considering this computational secrecy requirement, we construct the first *lossless* computational

fuzzy extractors (Construction 4.1), where the derived key r is as long as the entropy of the source w. Our construction is for the Hamming metric and uses the code-offset construction [JW99],[DORS08, Section 5] used in prior work, but with two crucial differences. First, the key r is not extracted from w like in the sketch-and-extract approach; rather w "encrypts" r in a way that is decryptable with the knowledge of some close w' (this idea is similar to the way the code-offset construction is presented in [JW99] as a "fuzzy commitment") . Second, the code used is a random linear code, which allows us to use the Learning with Errors (LWE) assumption due to Regev [Reg05, Reg10] and derive a longer key r.

Specifically, we use the recent result of Döttling and Müller-Quade [DMQ13], which shows the hardness of decoding random linear codes when the error vector comes from the uniform distribution, with each coordinate ranging over a small interval. This allows us to use w as the error vector, assuming it is uniform. We also use a result of Akavia, Goldwasser, and Vaikuntanathan [AGV09], which says that LWE has many hardcore bits, to hide r.

Because we use a random linear code, our decoding is limited to reconciling a logarithmic number of differences. Unfortunately, we cannot utilize the results that improve the decoding radius through the use of trapdoors (such as [Reg05]), because in a fuzzy extractor, there is no secret storage place for the trapdoor. If improved decoding algorithms are obtained for random linear codes, they will improve error-tolerance of our construction. Given the hardness of decoding random linear codes [BMvT78], we do not expect significant improvement in error-tolerance of our construction.

In Section 5, we are able to relax the assumption that w comes from the uniform distribution, and instead allow w to come from a symbol-fixing source [KZ07] (each dimension is either uniform or fixed). This relaxation follows from our results about the hardness of LWE when samples have a fixed (and adversarially known) error vector, which may be of independent interest (Theorem 5.2).

An Alternative Approach Computational extractors [Kra10, BDK$^+$11, DSGKM12] have the same goal of obtaining a pseudorandom key r from a source w in the setting without errors. They can be constructed, for example, by applying to pseudorandom generator to the output of an information-theoretic extractor. One way to build a computational *fuzzy* extractor is by using a computational extractor instead of the information-theoretic extractor in the sketch-and-extract construction of [DORS08]. However, this approach is possible only if conditional min-entropy of w conditioned on the sketch s is high enough. We compare the two approaches in Section 4.4.

2 Preliminaries

For a random variable $X = X_1||...||X_n$ where each X_i is over some alphabet Z, we denote by $X_{1,...,k} = X_1||...||X_k$. The *min-entropy* of X is $\mathrm{H}_\infty(X) = -\log(\max_x \Pr[X = x])$, and the *average (conditional)* min-entropy of X given Y is $\tilde{\mathrm{H}}_\infty(X|Y) = -\log(\mathbb{E}_{y\in Y} \max_x \Pr[X = x|Y = y])$ [DORS08, Section 2.4]. The *statistical distance* between random variables X and Y with the same domain is $\Delta(X, Y) = \frac{1}{2}\sum_x |\Pr[X = x] - \Pr[Y = x]|$. For a distinguisher D (or a class of distinguishers $\mathcal{D}$) we write the *computational distance* between X and Y as $\delta^D(X, Y) = |\mathbb{E}[D(X)] - \mathbb{E}[D(Y)]|$. We denote by $\mathcal{D}_{s_{sec}}$ the class of randomized circuits which output a single bit and have size at most s_{sec}. For a metric space $(\mathcal{M}, \mathsf{dis})$, the *(closed) ball of radius t around x* is the set of all points within radius t, that is, $B_t(x) = \{y|\mathsf{dis}(x, y) \le t\}$. If the size of a ball in a metric space does not depend on x, we denote by $|B_t(\cdot)|$ the size of a ball of radius t. For the Hamming metric over Z^n, $|B_t(\cdot)| = \sum_{i=0}^{t} \binom{n}{t}(|Z| - 1)^i$. U_n denotes the uniformly distributed random variable on $\{0, 1\}^n$. Usually, we use bold letters for vectors or matrices, capitalized letters for random variables, and lowercase letters for elements in a vector or samples from a random variable.

2.1 Fuzzy Extractors and Secure Sketches

We now recall definitions and lemmas from the work of Dodis et. al. [DORS08, Sections 2.5–4.1], adapted to allow for a small probability of error, as discussed in [DORS08, Sections 8]. Let $\mathcal{M}$ be a metric space with distance function dis.

Definition 2.1. *An* $(\mathcal{M}, m, \ell, t, \epsilon)$-fuzzy extractor *with error* δ *is a pair of randomized procedures, "generate"* (Gen) *and "reproduce"* (Rep)*, with the following properties:*

1. *The generate procedure* Gen *on input* $w \in \mathcal{M}$ *outputs an extracted string* $r \in \{0,1\}^\ell$ *and a helper string* $p \in \{0,1\}^*$.

2. *The reproduction procedure* Rep *takes an element* $w' \in \mathcal{M}$ *and a bit string* $p \in \{0,1\}^*$ *as inputs. The* correctness *property of fuzzy extractors guarantees that for* w *and* w' *such that* $\mathsf{dis}(w, w') \le t$*, if* R, P *were generated by* $(R, P) \leftarrow \mathsf{Gen}(w)$*, then* $\mathsf{Rep}(w', P) = R$ *with probability (over the coins of* $\mathsf{Gen}, \mathsf{Rep}$*) at least* $1 - \delta$*. If* $\mathsf{dis}(w, w') > t$*, then no guarantee is provided about the output of* Rep.

3. *The* security *property guarantees that for any distribution* W *on* $\mathcal{M}$ *of min-entropy* m*, the string* R *is nearly uniform even for those who observe* P*: if* $(R, P) \leftarrow \mathsf{Gen}(W)$*, then* $\mathbf{SD}((R, P), (U_\ell, P)) \le \epsilon$.

A fuzzy extractor is efficient if Gen *and* Rep *run in expected polynomial time.*

Secure sketches are the main technical tool in the construction of fuzzy extractors. Secure sketches produce a string s that does not decrease the entropy of w too much, while allowing recovery of w from a close w':

Definition 2.2. *An* $(\mathcal{M}, m, \tilde{m}, t)$-secure sketch *with error* δ *is a pair of randomized procedures, "sketch"* (SS) *and "recover"* (Rec)*, with the following properties:*

1. *The sketching procedure* SS *on input* $w \in \mathcal{M}$ *returns a bit string* $s \in \{0,1\}^*$.

2. *The recovery procedure* Rec *takes an element* $w' \in \mathcal{M}$ *and a bit string* $s \in \{0,1\}^*$*. The* correctness *property of secure sketches guarantees that if* $\mathsf{dis}(w, w') \le t$*, then* $\Pr[\mathsf{Rec}(w', \mathsf{SS}(w)) = w] \ge 1 - \delta$ *where the probability is taken over the coins of* SS *and* Rec*. If* $\mathsf{dis}(w, w') > t$*, then no guarantee is provided about the output of* Rec.

3. *The* security *property guarantees that for any distribution* W *over* $\mathcal{M}$ *with min-entropy* m*, the value of* W *can be recovered by the adversary who observes* w *with probability no greater than* $2^{-\tilde{m}}$*. That is,* $\tilde{\mathrm{H}}_\infty(W|\mathsf{SS}(W)) \ge \tilde{m}$.

A secure sketch is efficient *if* SS *and* Rec *run in expected polynomial time.*

Note that in the above definition of secure sketches (resp., fuzzy extractors), the errors are chosen before s (resp., P) is known: if the error pattern between w and w' depends the output of SS (resp., Gen), then there is no guarantee about the probability of correctness.

A fuzzy extractor can be produced from a *secure sketch* and an *average-case randomness extractor*. An average-case extractor is a generalization of a strong randomness extractor [NZ93, Definition 2]) (in particular, Vadhan [Vad12, Problem 6.8] showed that all strong extractors are average-case extractors with a slight loss of parameters):

Definition 2.3. *Let χ_1, χ_2 be finite sets. A function* $\mathtt{ext} : \chi_1 \times \{0,1\}^d \to \{0,1\}^\ell$ *a* (m,ϵ)-average-case extractor *if for all pairs of random variables X, Y over χ_1, χ_2 such that $\tilde{H}_\infty(X|Y) \geq m$, we have* $\Delta((\mathtt{ext}(X, U_d), U_d, Y), U_\ell \times U_d \times Y) \leq \epsilon$.

Lemma 2.4. *Assume* $(\mathsf{SS}, \mathsf{Rec})$ *is an* $(\mathcal{M}, m, \tilde{m}, t)$*-secure sketch with error* δ*, and let* $\mathtt{ext} : \mathcal{M} \times \{0,1\}^d \to \{0,1\}^\ell$ *be a* $(\tilde{m}, \epsilon)$*-average-case extractor. Then the following* $(\mathsf{Gen}, \mathsf{Rep})$ *is an* $(\mathcal{M}, m, \ell, t, \epsilon)$*-fuzzy extractor with error* δ*:*

- $\mathsf{Gen}(w)$: *generate* $x \leftarrow \{0,1\}^d$*, set* $p = (\mathsf{SS}(w), x), r = \mathtt{ext}(w; x)$*, and output* (r, p).
- $\mathsf{Rep}(w', (s, x))$: *recover* $w = \mathsf{Rec}(w', s)$ *and output* $r = \mathtt{ext}(w; x)$.

The main parameter we will be concerned with is the entropy loss of the construction. In this paper, we ask whether a smaller entropy loss can be achieved by considering a fuzzy extractor with a computational security requirement. We therefore relax the security requirement of Definition 2.1 to require a pseudorandom output instead of a truly random output. Also, for notational convenience, we modify the definition so that we can specify a general class of sources for which the fuzzy extractor is designed to work, rather than limiting ourselves to the class of sources that consists of all sources of a given min-entropy m, as in definitions above (of course, this modification can also be applied to prior definitions of information-theoretic secure sketches and fuzzy extractors).

Definition 2.5 (Computational Fuzzy Extractor)**.** *Let $\mathcal{W}$ be a family of probability distributions over $\mathcal{M}$. A pair of randomized procedures "generate"* (Gen) *and "reproduce"* (Rep) *is a* $(\mathcal{M}, \mathcal{W}, \ell, t)$-computational fuzzy extractor *that is* (ϵ, s_{sec})*-hard with error* δ *if* Gen *and* Rep *satisfy the following properties:*

- *The generate procedure* Gen *on input $w \in \mathcal{M}$ outputs an extracted string $R \in \{0,1\}^\ell$ and a helper string $P \in \{0,1\}^*$.*
- *The reproduction procedure* Rep *takes an element $w' \in \mathcal{M}$ and a bit string $P \in \{0,1\}^*$ as inputs. The* correctness *property guarantees that if* $\mathsf{dis}(w, w') \leq t$ *and* $(R, P) \leftarrow \mathsf{Gen}(w)$*, then* $\Pr[\mathsf{Rep}(w', P) = R] \geq 1 - \delta$ *where the probability is over the randomness of* $(\mathsf{Gen}, \mathsf{Rep})$*. If $d(w, w') > t$, then no guarantee is provided about the output of* Rep.
- *The* security *property guarantees that for any distribution $W \in \mathcal{W}$, the string R is pseudorandom conditioned on P, that is* $\delta^{\mathcal{D}_{ssec}}((R, P), (U_\ell, P)) \leq \epsilon$.

Any efficient fuzzy extractor is also a computational fuzzy extractor with the same parameters.

3 Impossibility of Computational Secure Sketches

In this section, we consider whether it is possible in build a secure sketch that retains significantly more computational than information-theoretic entropy. We consider two different notions for computational entropy, and for both of them show that corresponding secure sketches are subject to the same upper bounds as those for information-theoretic secure sketches. Thus, it seems that relaxing security of sketches from information-theoretic to computational does not help.

In particular, for the case of the Hamming metric and inputs that have full entropy, our results are as follows. In Section 3.1 we show that a sketch that retains HILL entropy implies a sketch that retains nearly the same amount of min-entropy. In Section 3.2, we show that the computational unpredictability

of a sketch is at most $\log|\mathcal{M}| - \log|B_t(\cdot)|$. Dodis et al. [DORS08, Section 8.2] construct sketches with essentially the same information-theoretic security[1] .

In Section 3.3, we discuss mechanisms for avoiding these bounds.

3.1 Bounds on Secure Sketches using HILL entropy

HILL entropy is a commonly used computational notion of entropy [HILL99]. It was extended to the conditional case by Hsiao, Lu, Reyzin [HLR07]. Here we recall a weaker definition due to Gentry and Wichs [GW11] (the term relaxed HILL entropy was introduced in [Rey11]); since we show impossibility even for this weaker definition, impossibility for the stronger definition follows immediately.

Definition 3.1. *Let (W, S) be a pair of random variables. W has* relaxed HILL entropy *at least k conditioned on S, denoted $H^{\mathtt{HILL\text{-}rlx}}_{\epsilon,s_{sec}}(W|S) \geq k$ if there exists a joint distribution (X, Y), such that $\tilde{H}_\infty(X|Y) \geq k$ and $\delta^{\mathcal{D}_{s_{sec}}}((W, S), (X, Y)) \leq \epsilon$.*

Intuitively, HILL entropy is as good as average min-entropy for all computationally-bounded observers. Thus, redefining secure sketches using HILL entropy is a natural relaxation of the original information-theoretic definition; in particular, the sketch-and-extract construction in Lemma 2.4 would yield pseudorandom outputs if the secure sketch ensured high HILL entropy. We will consider secure sketches that retain relaxed HILL entropy: that is, we say that $(\mathsf{SS}, \mathsf{Rec})$ is a *HILL-entropy* $(\mathcal{M}, m, \tilde{m}, t)$ *secure sketch* that is (ϵ, s_{sec})-hard with error δ if it satisfies Definition 2.2, with the security requirement replaced by $H^{\mathtt{HILL\text{-}rlx}}_{\epsilon,s_{sec}}(W|\mathsf{SS}(W)) \geq \tilde{m}$.

Unfortunately, we will show below that such a secure sketch implies an error correcting code with approximately $2^{\tilde{m}}$ points that can correct t random errors (see [DORS08, Lemma C.1] for a similar bound on information-theoretic secure sketches). For the Hamming metric, our result essentially matches the bound on information-theoretic secure sketches of [DORS08, Proposition 8.2]. In fact, we show that, for the Hamming metric, HILL-entropy secure sketches imply information-theoretic ones with similar parameters, and, therefore, the HILL relaxation gives no advantage.

The intuition for building error-correcting codes from HILL-entropy secure sketches is as follows. In order to have $H^{\mathtt{HILL\text{-}rlx}}_{\epsilon,s_{sec}}(W|\mathsf{SS}(W)) \geq \tilde{m}$, there must be a distribution X, Y such that $\tilde{\mathrm{H}}_\infty(X|Y) \geq \tilde{m}$ and (X, Y) is computationally indistinguishable from $(W, \mathsf{SS}(W))$. Sample a sketch $s \leftarrow \mathsf{SS}(W)$. We know that SS followed by Rec likely succeeds on $W|s$ (i.e., $\mathsf{Rec}(w', s) = w$ with high probability for $w \leftarrow W|s$ and $w' \leftarrow B_t(w)$). Consider the following experiment: 1) sample $y \leftarrow Y$, 2) draw $x \leftarrow X|y$ and 3) $x' \leftarrow B_t(x)$. By indistinguishability, $\mathsf{Rec}(x', y) = x$ with high probability. This means we can construct a large set $\mathcal{C}$ from the support of $X|y$. $\mathcal{C}$ will be an error correcting code and Rec an efficient decoder. We can then use standard arguments to turn this code into an information theoretic sketch.

To make this intuition precise, we need an additional technical condition: sampling a random neighbor of a point is efficient.

Definition 3.2. *We say a metric space $(\mathcal{M}, \mathsf{dis})$ is* (s_{neigh}, t)-neighborhood samplable *if there exists a randomized circuit Neigh of size s_{neigh} that for all $t' \leq t$, $\mathsf{Neigh}(w, t')$ outputs a random point at distance t' of w.*

We review definitions of Shannon codes [SWBH49]:

[1] The security in [DORS08, Section 8.2] is expressed in terms of entropy of the error rate; recall that $\log B_t(\cdot) \approx H_q(t/n)$, where n is the number of symbols, q is the alphabet size, and H_q is the q-ary entropy function.

Definition 3.3. *Let $\mathcal{C}$ be a set over space $\mathcal{M}$. We say that $\mathcal{C}$ is an* (t, ϵ)-Shannon code *if there exists an efficient procedure* Rec *such that for all $t' \leq t$ and for all $c \in \mathcal{C}$,* $\Pr[\mathsf{Rec}(\mathsf{Neigh}(c, t')) \neq c] \leq \epsilon$. *To distinguish it from the average-error Shannon code defined below, we will sometimes call it* maximal-error *Shannon code.*

This is a slightly stronger formulation than usual, in that for every size $t' < t$ we require the code to correct t' random errors[2].

Shannon codes work for all codewords. We can also consider a formulation that works for an "average" codeword.

Definition 3.4. *Let C be a distribution over space $\mathcal{M}$. We say that C is an* (t, ϵ)-average error Shannon code *if there exists an efficient procedure* Rec *such that for all $t' \leq t$* $\Pr_{c \leftarrow C}[\mathsf{Rec}(\mathsf{Neigh}(c, t')) \neq c] \leq \epsilon$.

An average error Shannon code is one whose average probability of error is bounded by ϵ. See [CT06, Pages 192-194] for definitions of average and maximal error probability. An average-error Shannon code is convertible to a maximal-error Shannon code with a small loss. We use the following pruning argument from [CT06, Pages 202-204] (we provide a proof in Section C.1):

Lemma 3.5. *Let C be a (t, ϵ)-average error Shannon code with recovery procedure* Rec *such that* $\mathrm{H}_\infty(C) \geq k$. *There is a set $\mathcal{C}'$ with $|\mathcal{C}'| \geq 2^{k-1}$ that is a $(t, 2\epsilon)$-(maximal error) Shannon code with recovery procedure* Rec.

We can now formalize the intuition above and show that a sketch that retains $\tilde{m}$-bits of relaxed HILL entropy implies a good error correcting code with nearly $2^{\tilde{m}}$ points (proof in Section C.2).

Theorem 3.6. *Let $(\mathcal{M}, \mathsf{dis})$ be a metric space that is (s_{neigh}, t)-neighborhood samplable. Let $(\mathsf{SS}, \mathsf{Rec})$ be an HILL-entropy $(\mathcal{M}, m, \tilde{m}, t)$-secure sketch that is (ϵ, s_{sec})-secure with error δ. Let s_{rec} denote the size of the circuit that computes* Rec. *If $s_{sec} \geq (t(s_{neigh} + s_{rec}))$, then there exists a value s and a set $\mathcal{C}$ with $|\mathcal{C}| \geq 2^{\tilde{m}-2}$ that is a $(t, 4(\epsilon + t\delta))$-Shannon code with recovery procedure* $\mathsf{Rec}(\cdot, s)$.

For the Hamming metric, any Shannon code (as defined in Definition 3.3) can be converted into an information-theoretic secure sketch (as described in [DORS08, Section 8.2] and references therein). The idea is to use the code offset construction, and convert worst-case errors to random errors by randomizing the order of the bits of w first, via a randomly chosen permutation π (which becomes part of the sketch and is applied to w' during Rec). The formal statement of this result can be expressed in the following Lemma (which is implicit in [DORS08, Section 8.2]).

Lemma 3.7. *For an alphabet Z, let $\mathcal{C}$ over Z^n be a (t, δ) Shannon code. Then there exists a $(Z^n, m, m - (n \log |Z| - \log |\mathcal{C}|), t)$ secure sketch with error δ for the Hamming metric over Z^n.*

Putting together Theorem 3.6 and Lemma 3.7 gives us the negative result for the Hamming metric: a HILL-entropy secure sketch (for the uniform distribution) implies an information-theoretic one with similar parameters:

[2] In the standard formulation, the code must correct a random error of size up to t, which may not imply that it can correct a random error of a much smaller size t', because the volume of the ball of size t' may be negligible compared to the volume of the ball of size t. For codes that are monotone (if decoding succeeds on a set of errors, it succeeds on all subsets), these formulations are equivalent. However, we work with an arbitrary recover functionality that is not necessarily monotone.

Corollary 3.8. *Let Z be an alphabet. Let $(\mathsf{SS}', \mathsf{Rec}')$ be an (ϵ, s_{sec})-HILL-entropy $(Z^n, n \log |Z|, \tilde{m}, t)$-secure sketch with error δ for the Hamming metric over Z^n, with Rec' of circuit size s_{rec}. If $s_{sec} \geq t(s_{rec} + n \log |Z|)$, then there exists a $(Z^n, n \log |Z|, \tilde{m} - 2, t)$ (information-theoretic) secure sketch with error $4(\epsilon + t\delta)$.*

Notes: In Corollary 3.8 we make no claim about the efficiency of the resulting $(\mathsf{SS}, \mathsf{Rec})$, because the proof of Theorem 3.6 is not constructive.

Corollary 3.8 extends to non-uniform distributions: if there exists a distribution whose HILL sketch retains $\tilde{m}$ bits of entropy, then for all distributions W, there is an information theoretic sketch that retains $\mathrm{H}_\infty(W) - (n \log |Z| - \tilde{m}) - 2$ bits of entropy.

3.2 Bounds on Secure Sketches using Unpredictability Entropy

In the previous section, we showed that any sketch that retained HILL entropy could be transformed into an information theoretic sketch. However, HILL entropy is a strong notion. In this section, we therefore ask whether it is useful to consider a sketch that satisfies a minimal requirement: the value of the input is computationally hard to guess given the sketch. We begin by recalling the definition of conditional unpredictability entropy, which captures the notion of "hard to guess" (we relax the definition slightly, similarly to the relaxation of HILL entropy described in the previous section).

Definition 3.9. *[HLR07, Definition 7] Let (W, S) be a pair of random variables. We say that W has* relaxed unpredictability entropy *at least k conditioned on S, denoted by $H^{\mathtt{unp\text{-}rlx}}_{\epsilon, s_{sec}}(W|S) \geq k$, if there exists a pair of distributions (X, Y) such that $\delta^{\mathcal{D}_{s_{sec}}}((W, S), (X, Y)) \leq \epsilon$, and for all circuits $\mathcal{I}$ of size s_{sec},*

$$\Pr[\mathcal{I}(Y) = X] \leq 2^{-k}.$$

We will say that a pair of procedures $(\mathsf{SS}, \mathsf{Rec})$ is a *unpredictability-entropy $(\mathcal{M}, m, \tilde{m}, t)$ secure sketch* that is (ϵ, s_{sec})-hard with error δ if it satisfies Definition 2.2, with the security requirement replaced by $H^{\mathtt{unp\text{-}rlx}}_{\epsilon, s_{sec}}(W|\mathsf{SS}(W)) \geq \tilde{m}$. Note this notion is quite natural: combining such a secure sketch in a sketch-and-extract construction of Lemma 2.4 with a particular type of extractor (called a extractor *reconstructive* [BSW03]), would yield a computational fuzzy extractor (per [HLR07, Lemma 6]).

Unfortunately, the conditional unpredictability entropy $\tilde{m}$ must decrease as t increases, as the following theorem states. (The proof of the theorem, generalized to more metric spaces, is in Section C.3.)

Theorem 3.10. *Let Z be an alphabet $(\mathsf{SS}, \mathsf{Rec})$ be an unpredictability-entropy $(Z^n, m, \tilde{m}, t)$-secure sketch that is (ϵ, s_{sec})-secure with error δ. If $s_{sec} \geq t(|\mathsf{Rec}| + n \log |Z|)$, then $\tilde{m} \leq n \log |Z| - \log |B_t(\cdot)| + \log(1 - \epsilon - t\delta)$.*

In particular, if the input is uniform, the entropy loss is about $\log |B_t(\cdot)|$. As mentioned at the beginning of Section 3, essentially the same entropy loss can be achieved with information-theoretic secure sketches, by using the randomized code-offset construction. However, it is conceivable that unpredictability entropy secure sketches could achieve lower entropy loss with greater efficiency for some parameter settings.

3.3 Avoiding sketch entropy upper bounds

The lower bounds of Corollary 3.8 and Theorem 3.10 are strongest for high entropy sources. This is necessary, if a source contains only codewords (of an error correcting code), no sketch is needed, and thus

there is no (computational) entropy loss. This same situation occurs when considering lower bounds for information-theoretic sketches [DORS08, Appendix C] .

Both of lower bounds arise because Rec must function as an error-correcting code for many points of any indistinguishable distribution. It may be possible to avoid these bounds if Rec outputs a fresh random variable[3]. Such an algorithm is called a computational fuzzy conductor. See [KR09] for the definition of a fuzzy conductor. To the best of our knowledge, a computational fuzzy conductor has not been defined in the literature, the natural definition is to replace the pseudorandomness condition in Definition 2.5 with a HILL entropy requirement.

Our construction (in Section 4) has pseudorandom output and immediately satisfies definition of a computational fuzzy extractor (Definition 2.5). It may be possible to achieve significantly better parameters with a construction that is a computational fuzzy conductor (but not a computational fuzzy extractor) and then applying an extractor. We leave this as an open problem.

4 Computational Fuzzy Extractor based on LWE

In this section we describe our main construction. Security of our construction depends on the source W. We first consider a uniform source W; we consider other distributions in Section 5. Our construction uses the code-offset construction [JW99], [DORS08, Section 5] instantiated with a random linear code over a finite field $\mathbb{F}_q$. Let Decode_t be an algorithm that decodes a random linear code with at most t errors (we will present such an algorithm later, in Section 4.2).

Construction 4.1. *Let n be a security parameter and let $m \geq n$. Let q be a prime. Define* Gen, Rep *as follows:*

Gen	Rep
1. *Input: $w \leftarrow W$ (where W is some distribution over $\mathbb{F}_q^m$).*	1. *Input: (w', p) (where the Hamming distance between w' and w is at most t).*
2. *Sample $\mathbf{A} \in \mathbb{F}_q^{m \times n}, \mathbf{x} \in \mathbb{F}_q^n$ uniformly.*	2. *Parse p as $(\mathbf{A}, \mathbf{c})$; let $\mathbf{b} = \mathbf{c} - w'$.*
3. *Compute $p = (\mathbf{A}, \mathbf{Ax} + w)$, $r = \mathbf{x}_{1,\ldots,n/2}$.*	3. *Let $x = \mathsf{Decode}_t(\mathbf{A}, \mathbf{b})$*
4. *Output (r, p).*	4. *Output $r = x_{1,\ldots,n/2}$.*

Intuitively, security comes from the computational hardness of decoding random linear codes with a high number of errors (introduced by w). In fact, we know that decoding a random linear code is NP-hard [BMvT78]; however, this statement is not sufficient for our security goal, which is to show $\delta^{\mathcal{D}_{ssec}}((X_{1,\ldots,n/2}, P), (U_{n/2 \log q}, P)) \leq \epsilon$. Furthermore, this construction is only useful if Decode_t can be efficiently implemented.

The rest of this section is devoted to making these intuitive statements precise. We describe the LWE problem and the security of our construction in Section 4.1. We describe one possible polynomial-time Decode_t (which corrects more errors than is possible by exhaustive search) in Section 4.2. In Section 4.3, we describe parameter settings that allow us to extract as many bits as the input entropy, resulting in a

[3]If some efficient algorithm can take the output of Rec and efficiently transform it back to the source W, the bounds of Corollary 3.8 and Theorem 3.10 both apply. This means that we need to consider constructions that are hard to invert (either information-theoretically or computationally).

lossless construction. In Section 4.4, we compare Construction 4.1 to using a sketch-and-extract approach (Lemma 2.4) instantiated with a computational extractor.

4.1 Security of Construction 4.1

The LWE problem was introduced by Regev [Reg05, Reg10] as a generalization of "learning parity with noise." We now recall the definition of the decisional version of the problem.

Definition 4.2 (Decisional LWE)**.** *Let n be a security parameter. Let $m = m(n) = \mathtt{poly}(n)$ be an integer and $q = q(n) = \mathtt{poly}(n)$ be a prime*[4]*. Let $\mathbf{A}$ be the uniform distribution over $\mathbb{F}_q^{m\times n}$, X be the uniform distribution over $\mathbb{F}_q^n$ and χ be an arbitrary distribution on $\mathbb{F}_q^m$. The decisional version of the* LWE *problem, denoted* dist-LWE$_{n,m,q,\chi}$*, is to distinguish the distribution $(\mathbf{A}, \mathbf{A}X + \chi)$ from the uniform distribution over $(\mathbb{F}_q^{m\times n}, \mathbb{F}_q^m)$.*

We say that dist-LWE$_{n,m,q,\chi}$ *is (ϵ, s_{sec})-secure if no (probabilistic) distinguisher of size s_{sec} can distinguish the* LWE *instances from uniform except with probability ϵ. If for any $s_{sec} = \mathtt{poly}(n)$, there exists $\epsilon = \mathtt{ngl}(n)$ such that* dist-LWE$_{n,m,q,\chi}$ *is (ϵ, s_{sec})-secure, then we say it is* secure*.*

Regev[Reg05] and Peikert [Pei09] show that dist-LWE$_{n,m,q,\chi}$ is secure when the distribution χ of errors is Gaussian, as follows. Let $\bar{\Psi}_\rho$ be the discretized Gaussian distribution with variance $(\rho q)^2/2\pi$, where $\rho \in (0, 1)$ with $\rho q > 2\sqrt{n}$. If GAPSVP and SIVP are hard to approximate (on lattices of dimension n) within polynomial factors for quantum algorithms, then dist-LWE$_{n,m,q,\bar{\Psi}_\rho^m}$ is secure.

The above formulation of LWE requires the error term to come from the discretized Gaussian distribution, which makes it difficult to use it for constructing fuzzy extractors (because using w and w' to sample Gaussian distributions will increase the distance between the error terms and/or reduce their entropy). Fortunately, recent work Döttling and Müller-Quade [DMQ13] shows the security of LWE, under the same assumptions, when errors come from the uniform distribution over a small interval[5]. This allows us to directly encode w as the error term in an LWE problem by splitting it into m blocks. The size of these blocks is dictated by the following result of Döttling and Müller-Quade:

Lemma 4.3. *[DMQ13, Corollary 1] Let n be a security parameter. Let $q = q(n) = \mathtt{poly}(n)$ be a prime and $m = m(n) = \mathtt{poly}(n)$ be an integer with $m \geq 3n$. Let $\sigma \in (0, 1)$ be an arbitrarily small constant and let $\rho = \rho(n) \in (0, 1/10)$ be such that $\rho q \geq 2n^{1/2+\sigma}m$. If the approximate decision-version of the shortest vector problem (GAPSVP) and the shortest independent vectors problem (SIVP) are hard within a factor of $\tilde{O}(n^{1+\sigma}m/\rho)$ for quantum algorithms in the worst case, then, for χ the uniform distribution over $[-\rho q, \rho q]^m$,* dist-LWE$_{n,m,q,\chi}$ *is secure.*

To extract pseudorandom bits, we use a result of Akavia, Goldwasser, and Vaikuntanathan [AGV09] to show that X has simultaneously many hardcore bits. The result says that if dist-LWE$_{(n-k,m,q,\chi)}$ is secure then any k variables of X in a dist-LWE$_{(n,m,q,\chi)}$ instance are hardcore. We state their result for a general error distribution (noting that their proof does not depend on the error distribution):

Lemma 4.4. *[AGV09, Lemma 2] If* dist-LWE$_{(n-k,m,q,\chi)}$ *is (ϵ, s_{sec}) secure then*

$$\delta^{\mathcal{D}_{s_{sec'}}}((X_{1,\ldots,k}, \mathbf{A}, \mathbf{A}X + \chi), (U, \mathbf{A}, \mathbf{A}X + \chi)) \leq \epsilon\,,$$

where U denotes the uniform distribution over $\mathbb{F}_q^k$, $\mathbf{A}$ denotes the uniform distribution over $\mathbb{F}_q^{m\times n}$, X denotes the uniform distribution over $\mathbb{F}_q^n$, $X_{1,\ldots,k}$ denote the first k coordinates of x, and $s'_{sec} \approx s_{sec} - n^3$.

[4]Unlike in common formulations of LWE, where q can be any integer, we need q to be prime for decoding.

[5]Micciancio and Peikert provide a similar formulation in [MP13]. The result Döttling and Müller-Quade provides better parameters for our setting.

The security of Construction 4.1 follows from Lemmas 4.3 and 4.4 as long as all the parameters are set appropriately (see Theorem 4.7), because we use the hardcore bits of X as our key.

4.2 Efficiency of Construction 4.1

Construction 4.1 is useful only if Decode_t can be efficiently implemented. We need a decoding algorithm for a random linear code with t errors that runs in polynomial time. We present a simple Decode_t that runs in polynomial time and can correct correcting $O(\log n)$ errors (note that this corresponds to a superpolynomial number of possible error patterns). This algorithm is only a proof of concept, and neither the algorithm nor its analysis have been optimized for constants. An improved decoding algorithm can replace our algorithm, which will increase our correcting capability and improve Construction 4.1.

Construction 4.5. *We consider a setting of (n, m, q, χ) where $m \geq 3n$. We describe Decode_t:*

1. *Input $\mathbf{A}, \mathbf{b} = \mathbf{Ax} + w - w'$*
2. *Randomly select rows without replacement $i_1, ..., i_{2n} \leftarrow [1, m]$.*
3. *Restrict $\mathbf{A}, \mathbf{b}$ to rows $i_1, ..., i_{2n}$; denote these $\mathbf{A}_{i_1,...,i_{2n}}, \mathbf{b}_{i_1,...,i_{2n}}$.*
4. *Find n rows of $\mathbf{A}_{i_1,...,i_{2n}}$ that are linearly independent. If no such rows exist, output $\perp$ and stop.*
5. *Denote by $\mathbf{A}', \mathbf{b}'$ the restriction of $\mathbf{A}_{i_1,...,i_{2n}}, \mathbf{b}_{i_1,...,i_{2n}}$ (respectively) to these rows. Compute $\mathbf{x}' = (\mathbf{A}')^{-1}\mathbf{b}'$.*
6. *If $\mathbf{b} - \mathbf{Ax}'$ has more than t nonzero coordinates, go to step (2).*
7. *Output $\mathbf{x}'$.*

Each step is computable in time $O(n^3)$. For Decode_t to be efficient, we need t to be small enough so that with probability at least $\frac{1}{\mathtt{poly}(n)}$, none of the $2n$ rows selected in step 2 have errors (i.e., so that w and w' agree on those rows). If this happens, and $\mathbf{A}_{i_1,...,i_{2n}}$ has rank n (which is highly likely), then $\mathbf{x}' = \mathbf{x}$, and the algorithm terminates. However, we also need to ensure correctness: we need to make sure that if $\mathbf{x}' \neq \mathbf{x}$, we detect it in step 6. This detection will happen if $\mathbf{b} - \mathbf{Ax}' = \mathbf{A}(\mathbf{x} - \mathbf{x}') + (w - w')$ has more than t nonzero coordinates. It suffices to ensure that $\mathbf{A}(\mathbf{x} - \mathbf{x}')$ has at least $2t + 1$ nonzero coordinates (because at most t of those can be zeroed out by $w - w'$), which happens whenever the code generated by $\mathbf{A}$ has distance $2t + 1$.

Setting $t = O(\frac{m}{n} \log n)$ is sufficient to ensure efficiency. Random linear codes have distance at least $O(\frac{m}{n} \log n)$ with probability $1 - e^{-\Omega(n)}$ (the exact statement is in Corollary A.2), so this also ensures correctness. The formal statement is below (proof in Section C.4):

Lemma 4.6 (Efficiency of Decode_t when $t \leq d(m/n - 2) \log n$)**.** *Let d be a positive constant and assume that $\mathsf{dis}(W, W') \leq t$ where $t \leq d(\frac{m}{n} - 2) \log n$. Then Decode_t runs in expected time $O(n^{4d+3})$ operations in $\mathbb{F}_q$ (this expectation is over the choice of random coins of Decode_t, regardless of the input, as long as $\mathsf{dis}(w, w') \leq t$). It outputs X with probability $1 - e^{-\Omega(n)}$ (this probability is over the choice of the random matrix $\mathbf{A}$ and random choices made by Decode_t).*

4.3 Lossless Computational Fuzzy Extractor

We now state a setting of parameters that yields a lossless construction.

Theorem 4.7. *Let n be a security parameter and let the number of errors $t = c \log n$ for some positive constant c. Let d be a positive constant (giving us a tradeoff between running time of* Rep *and $|w|$). Consider the Hamming metric over the alphabet $Z = [-2^{b-1}, 2^{b-1}]$, where $b = \log 2(c/d+2)n^2 = O(\log n)$. Let W be uniform over $\mathcal{M} = Z^m$, where $m = (c/d+2)n = O(n)$. If GAPSVP and SIVP are hard to approximate within polynomial factors using quantum algorithms, then there is a setting of $q = \mathtt{poly}(n)$ such that for any polynomial $s_{sec} = \mathtt{poly}(n)$ there exists $\epsilon = \mathtt{ngl}(n)$ such that the following holds: Construction 4.1 is a $(\mathcal{M}, W, m \log |Z|, t)$-computational fuzzy extractor that is (ϵ, s_{sec})-hard with error $\delta = e^{-\Omega(n)}$. The generate procedure* Gen *takes $O(n^2)$ operations over $\mathbb{F}_q$, and the reproduce procedure* Rep *takes expected time $O(n^{4d+3})$ operations over $\mathbb{F}_q$.*

Proof. Security follows by combining Lemmas 4.3 and 4.4; efficiency follows by Lemma 4.6. For a more detailed explanation of the various parameters and constraints see Section D. □

Theorem 4.7 shows that a computational fuzzy extractor can be built without incurring any entropy loss. We can essentially think of $\mathbf{A}X + W$ as an encryption of X that where decryption works from any close W'.

4.4 Comparison with computational-extractor-based constructions

As we already mentioned in the introduction, an alternative approach to building a computational fuzzy extractor is to use a computational extractor (e.g., [Kra10, BDK+11, DSGKM12]) in place of the information-theoretic extractor in the sketch-and-extract construction. We will call this approach *sketch-and-comp-extract*. (A simple example of a computational extractor is a pseudorandom generator applied to the output of an information-theoretic extractor; note that LWE-based pseudorandom generators exist [AIK06].)

This approach (specifically, its analysis via Lemma 2.4) works as long as the amount of entropy $\tilde{m}$ of w conditioned on the sketch s remains high enough to run a computational extractor. However, as discussed in Section 3, $\tilde{m}$ decreases with the error parameter t due to coding bounds, and it is conceivable that, if W has barely enough entropy to begin with, it will have too little entropy left to run a computational extractor once s is known.

In contrast, our approach does not require the entropy of w conditioned on $p = (\mathbf{A}, \mathbf{A}X+w)$ to be high enough for a computational extractor. Instead, we require that w is not computationally recoverable given p. This requirement is weaker—in particular, in our construction, w may have no information-theoretic entropy conditioned on p. The key difference in our approach is that instead of extracting from w, we hide secret randomness using w. Computational extractors are not allowed to have private randomness [Kra10, Definition 3].

The main advantage of our analysis (instead of sketch-and-comp-extract) is that security need not depend on the error-tolerance t. In our construction, the error-tolerance depends only on the best available decoding algorithm for random linear codes, because decoding algorithms will not reach the information-theoretic decoding radius.

Unfortunately, LWE parameter sizes require relatively long w. Therefore, in practice, sketch-then-comp-extract will beat our construction if the computational extractor is instantiated efficiently based on assumptions other than LWE (for example, a cryptographic hash function for an extractor and a block cipher for a PRG). However, we believe that our conceptual framework can lead to better constructions.

Of particular interest are other codes that are easy to decode up to t errors but become computationally hard as the number of errors increases.

To summarize, the advantage of Construction 4.1 is that the security of our construction does not depend on the decoding radius t. The disadvantages of Construction 4.1 are that it supports a limited number of errors and only a uniformly distributed source. We begin to address this second problem in the next section.

5 Computational Fuzzy Extractor for Nonuniform Sources

While showing the security of Construction 4.1 for arbitrary high-min-entropy distributions is an open problem, in this section we show it for a particular class of distributions called symbol-fixing. First we recall the notion of a symbol fixing source (from [KZ07, Definition 2.3]):

Definition 5.1. *Let $W = (W_1, ..., W_{m+\alpha})$ be a distribution where each W_i takes values over an alphabet Z. We say that it is a $(m+\alpha, m, |Z|)$* symbol fixing source *if for α indices $i_1, \ldots, i_\alpha$, the symbols W_{i_α} are fixed, and the remaining m symbols are chosen uniformly at random. Note that $H_\infty(W) = m \log |Z|$.*

The following theorem states the main technical result of this section, which is of potential interest outside our specific setting. The result is that dist-LWE with symbol-fixing sources is implied by standard dist-LWE (but for n and m reduced by the amount of fixed symbols).

Theorem 5.2. *Let n be a security parameter, m, α be polynomial in n, and $q = \mathtt{poly}(n)$ be a prime and $\beta \in \mathbb{Z}^+$ be such that $q^{-\beta} = \mathtt{ngl}(n)$. Let U denote the uniform distribution over Z^m for an alphabet $Z \subset \mathbb{F}_q$, and let W denote an $(m+\alpha, m, |Z|)$ symbol fixing source over $Z^{m+\alpha}$. If $\mathsf{dist\text{-}LWE}_{n,m,q,U}$ is secure, then $\mathsf{dist\text{-}LWE}_{n+\alpha+\beta,m+\alpha,q,W}$ is also secure.*

Theorem 5.2 also holds for an arbitrary error distribution (not just uniform error) in the following sense. Let χ' be an arbitrary error distribution. Define χ as the distribution where m dimensions are sampled according to χ' and the remaining dimensions have some fixed error. Then, security of $\mathsf{dist\text{-}LWE}_{n,m,q,\chi'}$ implies security of $\mathsf{dist\text{-}LWE}_{n+\alpha+\beta,m+\alpha,q,\chi}$. We show this stronger version of the theorem in Appendix B.

The intuition for this result is as follows. Providing a single sample with no error "fixes" at most a single variable. Thus, if there are significantly more variables than samples with no error, search LWE should still be hard. We are able to show a stronger result that dist-LWE is still hard. The nontrivial part of the reduction is using the additional $\alpha + \beta$ variables to "explain" a random value for the last α samples, without knowing the other variables. The β parameter is the slack needed to ensure that the "free" variables have influence on the last α samples.

Theorem 5.2 allows us to construct a lossless computational fuzzy extractor from block-fixing sources:

Theorem 5.3. *Let n be a security parameter and let $t = c \log n$ for some positive constant c. Let $d \leq c$ be a positive constant and consider the Hamming metric over the alphabet $Z = [-2^{b-1}, 2^{b-1}]$, where $b \approx \log 2(c/d+2)n^2 = O(\log n)$. Let $\mathcal{M} = Z^{m+\alpha}$ where $m = (c/d+2)n = O(n)$ and $\alpha \leq n/3$. Let $\mathcal{W}$ be the class of all $(m+\alpha, m, |Z|)$-symbol fixing sources. If GAPSVP and SIVP are hard to approximate within polynomial factors using quantum algorithms, then there is a setting of $q = \mathtt{poly}(n)$ such that for any polynomial $s_{sec} = \mathtt{poly}(n)$ there exists $\epsilon = \mathtt{ngl}(n)$ such that the following holds: Construction 4.1 is a $(\mathcal{M}, \mathcal{W}, m \log |Z|, t)$-computational fuzzy extractor that is (ϵ, s_{sec})-hard with error $\delta = e^{-\Omega(n)}$. The generate procedure* Gen *takes $O(n^2)$ operations over $\mathbb{F}_q$, and the reproduce procedure* Rep *takes expected time $O(n^{4d+3} \log n)$ operations over $\mathbb{F}_q$.*

Proof. Security follows by Lemmas 4.3 and 4.4 and Theorem 5.2 . Since Lemma 4.3 requires that $q = O(n^{3/2})$, setting $\beta = \omega(1)$ is sufficient for $q^{-\beta} = \mathtt{ngl}(n)$ as required in Theorem 5.2. Efficiency follows by Lemma 4.6. For a more detailed explanation of parameters see Section D.1. □

Acknowledgements

The authors are grateful to Ran Canetti, Yevgeniy Dodis, Nico Döttling, Danielle Micciancio, Jörn Müller-Quade, Christopher Peikert, Oded Regev, Adam Smith, and Daniel Wichs for helpful discussions, creative ideas, and important references. In particular, the authors thank Nico Döttling for describing his result on LWE with uniform errors. The work of Benjamin Fuller is sponsored by the United States Air Force under Air Force Contract FA8721-05-C-0002. Opinions, interpretations, conclusions and recommendations are those of the authors and are not necessarily endorsed by the United States Government.

References

[AGV09] Adi Akavia, Shafi Goldwasser, and Vinod Vaikuntanathan. Simultaneous hardcore bits and cryptography against memory attacks. In Omer Reingold, editor, *Theory of Cryptography*, volume 5444 of *Lecture Notes in Computer Science*, pages 474–495. Springer Berlin Heidelberg, 2009.

[AIK06] Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. On pseudorandom generators with linear stretch in NC 0. *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 260–271, 2006.

[BBR88] Charles H. Bennett, Gilles Brassard, and Jean-Marc Robert. Privacy amplification by public discussion. *SIAM journal on Computing*, 17(2):210–229, 1988.

[BDK+05] Xavier Boyen, Yevgeniy Dodis, Jonathan Katz, Rafail Ostrovsky, and Adam Smith. Secure remote authentication using biometric data. In *EUROCRYPT*, pages 147–163. Springer, 2005.

[BDK+11] Boaz Barak, Yevgeniy Dodis, Hugo Krawczyk, Olivier Pereira, Krzysztof Pietrzak, François-Xavier Standaert, and Yu Yu. Leftover hash lemma, revisited. In *Advances in Cryptology–CRYPTO 2011*, pages 1–20. Springer, 2011.

[BMvT78] Elwyn Berlekamp, Robert McEliece, and Henk van Tilborg. On the inherent intractability of certain coding problems. *IEEE Transactions on Information Theory*, 24(3):384 – 386, May 1978.

[BS00] Sacha Brostoff and M.Angela Sasse. Are passfaces more usable than passwords?: A field trial investigation. *People and Computers*, pages 405–424, 2000.

[BSW03] Boaz Barak, Ronen Shaltiel, and Avi Wigderson. Computational analogues of entropy. In *11th International Conference on Random Structures and Algorithms*, pages 200–215, 2003.

[CKOR10] Nishanth Chandran, Bhavana Kanukurthi, Rafail Ostrovsky, and Leonid Reyzin. Privacy amplification with asymptotically optimal entropy loss. In *Proceedings of the 42nd ACM Symposium on Theory of Computing*, pages 785–794, New York, NY, USA, 2010. ACM.

[CM05] Claude Castelluccia and Pars Mutaf. Shake them up!: A movement-based pairing protocol for CPU-constrained devices. In *Proceedings of the 3rd international conference on Mobile systems, applications, and services*, pages 51–64. ACM, 2005.

[Coo00] Colin Cooper. On the rank of random matrices. *Random Structures & Algorithms*, 16(2):209–232, 2000.

[CT06] Thomas M. Cover and Joy A. Thomas. *Elements of information theory*. Wiley-InterScience, 2nd edition, 2006.

[Dau04] John Daugman. How iris recognition works. *Circuits and Systems for Video Technology, IEEE Transactions on*, 14(1):21 – 30, January 2004.

[DMQ13] Nico Döttling and Jörn Müller-Quade. Lossy codes and a new variant of the learning-with-errors problem. In Thomas Johansson and Phong Q. Nguyen, editors, *EUROCRYPT*, volume 7881 of *Lecture Notes in Computer Science*, pages 18–34. Springer, 2013.

[DORS08] Yevgeniy Dodis, Rafail Ostrovsky, Leonid Reyzin, and Adam Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM Journal on Computing*, 38(1):97–139, 2008.

[DSGKM12] Dana Dachman-Soled, Rosario Gennaro, Hugo Krawczyk, and Tal Malkin. Computational extractors and pseudorandomness. In *Theory of Cryptography*, pages 383–403. Springer, 2012.

[DW09] Yevgeniy Dodis and Daniel Wichs. Non-malleable extractors and symmetric key cryptography from weak secrets. In *Proceedings of the 41st annual ACM Symposium on Theory of Computing*, pages 601–610, New York, NY, USA, 2009. ACM.

[Gur10] Venkatesan Guruswami. Introduction to coding theory - lecture 2: Gilbert-Varshamov bound. University Lecture, 2010.

[GW11] Craig Gentry and Daniel Wichs. Separating succinct non-interactive arguments from all falsifiable assumptions. *STOC. ACM, New York*, pages 99–108, 2011.

[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.

[HLR07] Chun-Yuan Hsiao, Chi-Jen Lu, and Leonid Reyzin. Conditional computational entropy, or toward separating pseudoentropy from compressibility. In *EUROCRYPT*, pages 169–186, 2007.

[JW99] Ari Juels and Martin Wattenberg. A fuzzy commitment scheme. In *Sixth ACM Conference on Computer and Communication Security*, pages 28–36. ACM, November 1999.

[KR09] Bhavana Kanukurthi and Leonid Reyzin. Key agreement from close secrets over unsecured channels. In *EUROCRYPT*, pages 206–223, 2009.

[Kra10] Hugo Krawczyk. Cryptographic extraction and key derivation: The HKDF scheme. In *Advances in Cryptology–CRYPTO 2010*, pages 631–648. Springer, 2010.

[KZ07] Jesse Kamp and David Zuckerman. Deterministic extractors for bit-fixing sources and exposure-resilient cryptography. *SIAM Journal on Computing*, 36(5):1231–1247, 2007.

[MP13] Daniele Micciancio and Chris Peikert. Hardness of SIS and LWE with Small Parameters. Cryptology ePrint Archive, 2013.

[NZ93] Noam Nisan and David Zuckerman. Randomness is linear in space. *Journal of Computer and System Sciences*, pages 43–52, 1993.

[Pei09] Chris Peikert. Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In *Proceedings of the 41st annual ACM Symposium on Theory of Computing*, pages 333–342, New York, NY, USA, 2009. ACM.

[Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *Proceedings of the thirty-seventh annual ACM Symposium on Theory of Computing*, pages 84–93, New York, NY, USA, 2005. ACM.

[Reg10] Oded Regev. The learning with errors problem (invited survey). *Annual IEEE Conference on Computational Complexity*, 0:191–204, 2010.

[Rey11] Leonid Reyzin. Some notions of entropy for cryptography. In *Information Theoretic Security*, pages 138–142. Springer, 2011.

[SD07] G. Edward Suh and Srinivas Devadas. Physical unclonable functions for device authentication and secret key generation. In *Proceedings of the 44th annual Design Automation Conference*, pages 9–14. ACM, 2007.

[SWBH49] Claude E. Shannon, Warren Weaver, Richard E. Blahut, and Bruce Hajek. *The mathematical theory of communication*, volume 117. University of Illinois press Urbana, 1949.

[TSv+06] Pim Tuyls, Geert-Jan Schrijen, Boris Škoriá, Jan Geloven, Nynke Verhaegh, and Rob Wolters. Read-proof hardware from protective coatings. In Louis Goubin and Mitsuru Matsui, editors, *Cryptographic Hardware and Embedded Systems - CHES 2006*, volume 4249 of *Lecture Notes in Computer Science*, pages 369–383. Springer Berlin Heidelberg, 2006.

[Vad12] Salil Vadhan. *Pseudorandomness*. Foundations and Trends in Theoretical Computer Science. Now Publishers, 2012. To appear; preprint at `http://people.seas.harvard.edu/~salil/pseudorandomness/`.

[ZH93] Moshe Zviran and William J. Haga. A comparison of password techniques for multilevel authentication mechanisms. *The Computer Journal*, 36(3):227–237, 1993.

A Properties of Random Linear Codes

For efficient decoding of Construction 4.1, we need the LWE instance to have high distance with overwhelming probability. We will use the q-ary entropy function, denoted $H_q(x)$ and defined as $H_q(x) = x\log_q(q-1) - x\log_q x - (1-x)\log_q(1-x)$. Note that $H_2(x) = -x\log x - (1-x)\log(1-x)$. In the region $[0, \frac{1}{2}]$ for any value $q' \geq q$, $H_{q'}(x) \leq H_q(x)$. The following theorem is standard in coding theory:

Theorem A.1. *[Gur10, Theorem 8] For prime $q, \delta \in [0, 1 - 1/q), 0 < \epsilon < 1 - H_q(\delta)$ and sufficiently large m, the following holds for $n = \lceil (1 - H_q(\delta) - \epsilon)m \rceil$. If $\mathbf{A} \in \mathbb{F}_q^{m \times n}$ is drawn uniformly at random, then the linear code with $\mathbf{A}$ as a generator matrix has rate at least $(1 - H_q(\delta) - \epsilon)$ and relative distance at least δ with probability at least $1 - e^{-\Omega(m)}$.*

Our setting is the case where $m = poly(n) \geq 2n$ and $\delta = O(\log n/n)$. This setting of parameters satisfies Theorem A.1:

Corollary A.2. *Let n be a parameter and let $m = \texttt{poly}(n) \geq 2n$. Let q be a prime and $\tau = O(\frac{m}{n} \log n)$. For large enough values of n, when $\mathbf{A} \in \mathbb{F}_q^{m \times n}$ is drawn uniformly, the code generated by $\mathbf{A}$ has distance at least τ with probability at least $1 - e^{-\Omega(m)} \geq 1 - e^{-\Omega(n)}$.*

Proof. Let c be some constant. Let $\delta = \tau/m = \frac{c \log n}{n}$. We show the corollary for the case when $m = 2n$ (increasing the size of m only increases the relative distance). It suffices to show that for sufficiently large n, there exists $\epsilon > 0$ where $1 - H_q(\frac{c \log n}{n}) - \epsilon = 1/2$ or equivalently that $H_q(\frac{c \log n}{m}) < 1/2$ as then setting $\epsilon = 1/2 - H_q(\frac{c \log n}{n})$ satisfies Theorem A.1. For sufficiently large n:

- $\frac{c \log n}{n} < 1/2$, so we can work with the binary entropy function H_2.
- $\frac{c \log n}{n} < .1 < 1/2$ and thus $H_q(\frac{c \log n}{n}) < H_q(.1)$.

Putting these statements together, for large enough n, $H_q(\frac{c \log n}{n}) < H_q(.1) < H_2(.1) < 1/2$ as desired. This completes the proof. □

We also need that random matrices are full rank with high probability (to allow us to decode). We use the following claim (techniques from Cooper [Coo00]):

Claim A.3. *Let $q \geq 2$ be a prime. Let α, β be integers and let let $\mathbf{S} \xleftarrow{\$} \mathbb{F}_q^{\alpha \times (\alpha+\beta)}$ be uniformly generated. Then $\Pr[\texttt{rank}(\mathbf{S}) = \alpha] > 1 - q^{-\beta}$.*

Proof. Let p_i be the probability that the ith row is linearly dependent on the previous $i - 1$ rows. By the union bound, the probability that α rows are linearly dependent is bounded by $\sum_{i=1}^{\alpha} p_i$. Since $i - 1$ rows can span a space of size at most q^{i-1}, the probability p_i that a randomly chosen ith row is in that space is at most $q^{i-1}/q^{\alpha+\beta}$. So

$$\Pr[\texttt{rank}(\mathbf{S}) < \alpha] = \sum_{i=1}^{\alpha} \frac{q^{i-1}}{q^{\alpha+\beta}} = \frac{q^{\alpha} - 1}{q - 1} \frac{1}{q^{\alpha+\beta}} < q^{-\beta}.$$

□

B Proof of Theorem 5.2

Proof. We assume that all of the fixed blocks are located at the end and their fixed value is 0. If the blocks are fixed to some other value, the reduction is essentially the same. In the reduction, the distinguisher is allowed to depend on the source and can know the positions of the fixed blocks and their values. For a matrix $\mathbf{A}$ we will denote the i-th row by $\mathbf{a}_i$. For a set T of column indices, we denote by $\mathbf{A}_T$ the restriction of the matrix $\mathbf{A}$ to the columns contained in T. Similarly, for a vector $\mathbf{x}$ we denote by $\mathbf{x}_T$ the restriction of $\mathbf{x}$ to the variables contained in T. We use similar notation for the complement of T, denoted T^c. For

a matrix or vector we use T to denote the transpose. We use i as a index into matrix rows and the error vector and j as an index into columns and the solution vector.

Let n be a security parameter, $m, q, \alpha = \mathtt{poly}(n)$. Let β be such that $q^{-\beta} = \mathtt{ngl}(n)$. All operations are computed modulo q, and we omit " mod q" notation. Let χ' be some error distribution over $\mathbb{F}_q^m$ and let χ over $\mathbb{F}_q^{m+n}$ be defined by sampling χ' to obtain values on dimensions $1, ..., m$ and then appending α 0s.

Let D be a distinguisher that breaks $\mathsf{dist\text{-}LWE}_{(m+\alpha),(n+\alpha+\beta),q,\chi}$ with advantage $\epsilon > 1/\mathtt{poly}(n)$. Let $\mathbf{A}$ denote the uniform distribution over $\mathbb{F}_q^{(m+\alpha)\times(n+\alpha+\beta)}$, X denote the uniform distribution over $\mathbb{F}_q^{(n+\alpha+\beta)}$, and U denote the uniform distribution over $\mathbb{F}_q^{m+\alpha}$. Then

$$|\Pr[D(\mathbf{A}, \mathbf{A}X + \chi) = 1] - \Pr[D(\mathbf{A}, U) = 1]| > \epsilon.$$

We build a distinguisher that breaks $\mathsf{dist\text{-}LWE}_{m,n,q,\chi}$. Let $\mathbf{A}'$ denote the uniform distribution over $\mathbb{F}_q^{m\times n}$, X' denote the uniform distribution over $\mathbb{F}_q^n$, and U' denote the uniform distribution over $\mathbb{F}_q^m$. We will build a distinguisher D' of polynomial size for which

$$|\Pr[D'(\mathbf{A}', \mathbf{A}'X' + \chi') = 1] - \Pr[D'(\mathbf{A}', U') = 1]| > (\epsilon - \mathtt{ngl}(n))(1 - \mathtt{ngl}(n)) \approx \epsilon. \quad (1)$$

D' will make a single call to D, so we focus on how to prepare a random block-fixing instance for D from the random instance that D' is given. The code for D' is given in Figure 1.

The distinguisher D' has an advantage when $\mathbf{S}$ is of rank α. This occurs with overwhelming probability:

Claim B.1. *Let* $\mathbf{S} \xleftarrow{\$} \mathbb{F}_q^{\alpha\times(\alpha+\beta)}$ *be randomly generated. Then* $\Pr[\mathtt{rank}(\mathbf{S}) = \alpha] \geq 1 - \mathtt{ngl}(n)$.

Proof. Direct result of Claim A.3 because $q^{-\beta} = \mathtt{ngl}(n)$. □

The probability that a random $\mathbf{S}$ is not full rank is $\mathtt{ngl}(n)$ so the distinguisher D must still have an advantage when the matrix $\mathbf{S}$ is full rank. That is,

$$|\Pr[D(\mathbf{A}, \mathbf{A}X + \chi) = 1|\mathtt{rank}(\mathbf{S}) = \alpha] - \Pr[D(\mathbf{A}, U) = 1|\mathtt{rank}(\mathbf{S}) = \alpha]| > \epsilon - \mathtt{ngl}(n).$$

It suffices to show that D' prepares a good instance for D conditioned on $\mathbf{S}$ being full rank. We show this in the following three claims:

1. If $\mathbf{A}'$ is a random matrix then $\mathbf{A}$ is a random matrix subject to the condition that $\mathtt{rank}(\mathbf{S}) = \alpha$.
2. If $\mathbf{b}' = \mathbf{A}'\mathbf{x}' + \mathbf{e}'$ for uniform $\mathbf{A}'$ and $\mathbf{x}'$, then $\exists\mathbf{x}$ (uniformly distributed and independent of $\mathbf{A}$ and $\mathbf{e}'$) such that $\mathbf{b} = \mathbf{A}\mathbf{x} + \mathbf{e}$, where $\mathbf{e}_i = \mathbf{e}'_i$ for $1 \leq i \leq m$ and $\mathbf{e}_i = 0$ otherwise.
3. If the conditional distribution $\mathbf{b}' \,|\, \mathbf{A}'$ is uniform, then the conditional distribution $\mathbf{b} \,|\, \mathbf{A}$ is also uniform.

Claim B.2. *The matrix* $\mathbf{A}$ *is distributed as a uniformly random choice from the set of all matrices whose bottom-right* $\alpha \times (\alpha + \beta)$ *submatrix* $\mathbf{S}$ *satisfies* $\mathtt{rank}(\mathbf{S}) = \alpha$.

Proof. The bottom α rows of $\mathbf{A}$ (namely, $\mathbf{R}|\mathbf{S}$) are randomly generated (conditioned on $\mathtt{rank}(\mathbf{S}) = \alpha$). The top left $m \times n$ quadrant of $\mathbf{A}$ is also random, because it is produced as a sum of a uniformly random $\mathbf{A}'$ with some values that are uncorrelated with $\mathbf{A}'$. The submatrix of the top-right $m \times (\alpha+\beta)$ quadrant corresponding to $\mathbf{Q}_{T^c}$ (recall this is the restriction of $\mathbf{Q}$ to the columns not in T) is also random, because

1. Input $\mathbf{A}', \mathbf{b}'$, where $\mathbf{A}' \xleftarrow{\$} \mathbb{F}_q^{m \times n}$ and $\mathbf{b}'$ is either uniform over $\mathbb{F}_q^m$ or $\mathbf{b}' = \mathbf{A}'\mathbf{x}' + \mathbf{e}'$ for $\mathbf{e}' \xleftarrow{\$} \chi'$ and uniform $\mathbf{x}' \xleftarrow{\$} \mathbb{F}_q^n$.

2. Choose $\mathbf{R} \xleftarrow{\$} \mathbb{F}_q^{\alpha \times n}$ uniformly at random. Initialize $\mathbf{Q} \in \mathbb{F}_q^{m \times (\alpha+\beta)}$ to be the zero matrix.

3. Let $\mathbf{b}^* = (\mathbf{b}', b^*_{m+1}, \ldots, b^*_{m+\alpha})$, for uniformly chosen $(b^*_{m+1}, \ldots, b^*_{m+\alpha}) \xleftarrow{\$} \mathbb{F}_q^\alpha$.

4. Choose $\mathbf{S} \xleftarrow{\$} \mathbb{F}_q^{\alpha \times (\alpha+\beta)}$ uniformly at random.

   If $\texttt{rank}(\mathbf{S}) < \alpha$, stop and output a random bit.

5. Find a set of α linearly independent columns in $\mathbf{S}$. Let T be the set of indices of these columns.

6. For all $1 \leq j \leq \alpha + \beta$, $j \notin T$:

   Choose $x_{n+j} \xleftarrow{\$} \mathbb{F}_q$ uniformly at random.

   For $i = 1, ..., m$:

   Choose $\mathbf{Q}_{i,j} \xleftarrow{\$} \mathbb{F}_q$ uniformly at random.

   Set $b^*_i = b^*_i + \mathbf{Q}_{i,j} x_{n+j}$.

7. Initialize $\mathbf{A}^* = \left(\begin{array}{c|c} \mathbf{A}' & \mathbf{Q} \\ \hline \mathbf{R} & \mathbf{S} \end{array}\right)$.

8. For $i = 1, ..., m$:

   Choose a row vector $\gamma_i \leftarrow \mathbb{F}_q^{1 \times \alpha}$ uniformly at random.

   Set $\mathbf{a}_i \leftarrow \mathbf{a}^*_i + \gamma_i(\mathbf{R}||\mathbf{S})$

   Set $b_i \leftarrow b^*_i + \gamma_i(b^*_{m+1}, ..., b^*_{m+\alpha})^\mathsf{T}$

9. For $i = m+1, \ldots, m+\alpha$:

   Set $\mathbf{a}_i \leftarrow \mathbf{a}^*_i$

   Set $b_i = b^*_i$.

10. Output $D(\mathbf{A}, \mathbf{b})$.

Figure 1: A PPT D' that distinguishes LWE using distinguisher for LWE w/ block fixing source

it is initialized with random values to which some uncorrelated values are then added. It is important to note that all these values are independent of γ_i values.

Thus, we restrict attention to the $m \times \alpha$ submatrix of $\mathbf{A}$ that corresponds to $\mathbf{Q}_T$ in $\mathbf{A}^*$ (note that these values are 0 in $\mathbf{A}^*$). Consider a particular row i. That row is computed as $\gamma_i \mathbf{S}_T$. Since $\mathbf{S}_T$ is a full rank square matrix and γ_i is uniformly and independently generated, that row is also uniform and independent of other entries in $\mathbf{A}$. □

Claim B.3. *If D' is provided with input distributed as $\mathbf{A}', \mathbf{b}' = \mathbf{A}'\mathbf{x}' + \mathbf{e}'$ then $\mathbf{b} = \mathbf{A}\mathbf{x} + \mathbf{e}$, where*

- $e_i = e'_i$ *for* $1 \leq i \leq m$,
- $e_i = 0$ *for* $m < i \leq m + \alpha$,
- $x_j = x'_j$ *for* $1 \leq j \leq n$,
- *and x_j is uniform and independent of $\mathbf{A}$ and $\mathbf{e}'$ for $n < j \leq n + \alpha + \beta$,*

Proof. Partially define $\mathbf{x}$ as $x_j = x'_j$ if $1 \leq j \leq n$ and x_j as the value generated in step 6 for $j > n$ and $j \notin T$. Define the remaining variables $\mathbf{x}_T$ as the solution to the following system of equations.

$$\mathbf{S}_T \mathbf{x}_T = \begin{pmatrix} b^*_{m+1} \\ \vdots \\ b^*_{m+\alpha} \end{pmatrix} - \mathbf{R}\mathbf{x}' - \mathbf{S}_{T^c}\mathbf{x}_{T^c} \tag{2}$$

A solution $\mathbf{x}_T$ exists as $\mathbf{S}_T$ is full rank. Moreover, it is uniform and independent of $\mathbf{A}$ and $\mathbf{e}$, because $b^*_{m+1}, \ldots, b^*_{m+\alpha}$ are uniform and independent of $\mathbf{A}$ and $\mathbf{e}$.

We now show that $\mathbf{b}^* = \mathbf{A}^*\mathbf{x} + \mathbf{e}$. All entries in matrix $\mathbf{Q}$ corresponding to variables in T are set to zero. Thus, the values of $\mathbf{x}^T$ do not affect b^*_i for $1 \leq i \leq m$. The values of $\mathbf{x}_{T^c}$ are manually set, and $\mathbf{Q}_{i,j}\mathbf{x}_j$ is added to the corresponding b^*_i. Thus, for $1 \leq i \leq m$, we have $\mathbf{b}^* = \mathbf{A}^*\mathbf{x} + \mathbf{e}$. For $m < i$, this constraint is also satisfied by the values of $\mathbf{x}_T$ set in Equation 2.

Thus, it remains to show that step 8 preserves this solution. We now show that for all rows $1 \leq i \leq m$, if $b^*_i = \mathbf{a}^*_i\mathbf{x} + e_i$ then $b_i = \mathbf{a}_i\mathbf{x} + e_i$. Recall the other rows are not modified. We have the following for $1 \leq i \leq m$:

$$\begin{aligned} \mathbf{a}_i\mathbf{x} + e_i &= \left(\mathbf{a}^*_i + \gamma_i(\mathbf{R}||\mathbf{S})\right)\mathbf{x} + e_i \\ &= \mathbf{a}^*_i\mathbf{x} + e_i + \gamma_i(\mathbf{R}||\mathbf{S})\mathbf{x} \\ &= b^*_i + \gamma_i(\mathbf{R}||\mathbf{S})\mathbf{x} \end{aligned}$$

Recall that $b_i = b^*_i + \gamma_i(b^*_{m+1}, ..., b^*_{m+k})$. We consider the product $(\mathbf{R}||\mathbf{S})\mathbf{x}$. It suffices to show that

$(\mathbf{R}||\mathbf{S})\mathbf{x} = (b^*_{m+1}, ..., b^*_{m+\alpha})$,

$$\begin{aligned}(\mathbf{R}||\mathbf{S})\mathbf{x} &= \mathbf{R}\begin{pmatrix}\mathbf{x}_1\\ \vdots \\ \mathbf{x}_n\end{pmatrix} + \mathbf{S}_{T^c}\mathbf{x}_{T^c} + \mathbf{S}_T\mathbf{x}_T \\ &= \mathbf{R}\begin{pmatrix}\mathbf{x}_1\\ \vdots \\ \mathbf{x}_n\end{pmatrix} + \mathbf{S}_{T^c}\mathbf{x}_{T^c} + \begin{pmatrix}b^*_{m+1}\\ \vdots \\ b^*_{m+\alpha}\end{pmatrix} - \mathbf{R}\begin{pmatrix}\mathbf{x}_1\\ \vdots \\ \mathbf{x}_n\end{pmatrix} - \mathbf{S}_{T^c}\mathbf{x}_{T^c} \\ &= \begin{pmatrix}b^*_{m+1}\\ \vdots \\ b^*_{m+\alpha}\end{pmatrix}\end{aligned}$$

This completes the proof of the claim. □

Claim B.4. *If the conditional distribution* $\mathbf{b}' \mid \mathbf{A}'$ *is uniform, then* $\mathbf{b} \mid \mathbf{A}$ *is also uniform.*

Proof. Since $\mathbf{R}, \mathbf{S}$, and $\mathbf{Q}$ are chosen independently of $\mathbf{b}'$, the distribution $\mathbf{b}' \mid \mathbf{A}^*$ is uniform. Let $\mathbf{b}^*$ be the vector generated after step 6. Its first m coordinates are computed by adding the uniform vector $\mathbf{b}'$ to values that are independent of $\mathbf{b}^*$, and its remaining α coordinates $b^*_{m+1}, \ldots, b^*_{m+\alpha}$ are chosen uniformly. Thus $\mathbf{b}^* \mid \mathbf{A}^*$ is uniform.

Let $\mathbf{\Gamma}$ represent the matrix formed by γ_i. It is independent of $\mathbf{b}^*$ and $\mathbf{A}^*$, so $\mathbf{b}^* \mid (\mathbf{A}^*, \mathbf{\Gamma})$ is uniform. Let $\mathbf{\Gamma}' = \left(\begin{array}{c|c}\mathbf{I_m} & \mathbf{\Gamma} \\ \hline \mathbf{0} & \mathbf{I}_\alpha\end{array}\right)$. Note that $\mathbf{b} = \mathbf{\Gamma}'\mathbf{b}^*$. Since $\mathbf{b}^* \mid (\mathbf{A}^*, \mathbf{\Gamma})$ is uniform, and $\mathbf{\Gamma}'$ is invertible, $\mathbf{b} \mid (\mathbf{A}^*, \mathbf{\Gamma})$ must also be uniform. Since $\mathbf{A}$ is a deterministic function of $\mathbf{A}^*$ and $\mathbf{\Gamma}$ (assuming Step 5 is deterministic—if not, we can fix the coins used), the distribution $\mathbf{b} \mid \mathbf{A}$ is the same as $\mathbf{b} \mid (\mathbf{A}^*, \mathbf{\Gamma})$ and is thus also uniform. □

Finally, the reduction runs in polynomial time and together Claims B.2, B.3, and B.4 show that when $\mathtt{rank}(\mathbf{S}) = \alpha$ the distinguisher D' properly prepares the instance thus,

$$\begin{aligned}&\big|\Pr[D'(\mathbf{A}, \mathbf{A}X + \chi) = 1] - \Pr[D'(\mathbf{A}, U) = 1]\big| \\ &= \big|\Pr\left[D'(\mathbf{A}', \mathbf{u}') = 1 | \mathtt{rank}(\mathbf{S}) = \alpha\right] - \Pr\left[D'(\mathbf{A}', \mathbf{b}' = \mathbf{A}'\mathbf{x} + \mathbf{e}) = 1 | \mathtt{rank}(\mathbf{S}) = \alpha\right]\big| \Pr[\mathtt{rank}(\mathbf{S}) = \alpha] \\ &= |\Pr[D(\mathbf{A}, \mathbf{A}X + \chi) = 1 | \mathtt{rank}(\mathbf{S}) = \alpha] - \Pr[D(\mathbf{A}, U) = 1 | \mathtt{rank}(\mathbf{S}) = \alpha]| \Pr[\mathtt{rank}(\mathbf{S}) = \alpha] \\ &\geq (\epsilon - \mathtt{ngl}(n))(1 - \mathtt{ngl}(n)) \approx \epsilon\end{aligned}$$

Where the second line follows because we can detect when $\mathtt{rank}(\mathbf{S}) < \alpha$ and output a random bit in this case. Thus, Equation (1) is satisfied, this completes the proof. □

C Additional Proofs

C.1 Proof of Lemma 3.5

Proof. Let C be the (t, ϵ)-average error Shannon code with recovery procedure Rec such that $\mathrm{H}_\infty(C) \geq k$. Then for all $t' \leq t$

$$\sum_{c \in C} \Pr[C = c] \Pr[c' \leftarrow \mathsf{Neigh}(c, t') \wedge \mathsf{Rec}(c') \neq c] \leq \epsilon.$$

For c denote by $\epsilon_c = \Pr[c' \leftarrow \mathsf{Neigh}(c, t') \wedge \mathsf{Rec}(c') \neq c]$. Then by Markov's inequality:

$$\Pr_{c \in C}[\epsilon_c \leq 2 \mathop{\mathbb{E}}_{c \leftarrow C}[\epsilon_c]] = \Pr_{c \in C}[\epsilon_c \leq 2\epsilon] \geq \frac{1}{2}$$

Let C' denote the of set all $c \in C$ where $\epsilon_c \leq 2\epsilon$. Note that $\Pr_{c \leftarrow C}[c \in C'] \geq 1/2$. Since $H_\infty(C) \geq k$, we know $|C'| \geq 2^{k-1}$ (otherwise $\Pr_{c \leftarrow C}[c \in C'] = \sum_{c \in C'} \Pr[C = c]$ would be less than $2^{k-1}\frac{1}{2^k} = 1/2$). This completes the proof of the Lemma 3.5.

□

C.2 Proof of Theorem 3.6

Proof. Let W be an arbitrary distribution of min-entropy m. Let (X, Y) be a joint distribution such that $\tilde{\mathrm{H}}_\infty(X|Y) \geq k$ and

$$\delta^{\mathcal{D}_{s_{sec}}}((W, \mathsf{SS}(W)), (X, Y)) \leq \epsilon\,,$$

where $s_{sec} \geq t(s_{neigh} + s_{rec})$. One such (X, Y) must exist by the definition of conditional HILL entropy. Define D as:

1. Input $w \in \mathcal{M}, z \in \{0,1\}^*, t$.
2. For all $1 \leq t' \leq t$:
   $w' \leftarrow \mathsf{Neigh}(w, t')$.
   If $\mathsf{Rec}(w', z) \neq w$ output 0.
3. Output 1.

By correctness of the sketch $\Pr[D(W, \mathsf{SS}(W)) = 1] \geq 1 - t\delta$. Since $\delta^D((W, \mathsf{SS}(W)), (X, Y)) \leq \epsilon$, we know $\Pr[D(X, Y) = 1] \geq 1 - \epsilon - t\delta$. Let X_y denote the random variable $X|Y = y$. By Markov's inequality, there exists a set S_Y such that $\Pr[Y \in S_Y] \geq 1/2$ and for all $y \in S_Y$, $\Pr[D(X_y, y) = 1] \geq 1 - 2(\epsilon + t\delta)$.

Because $\tilde{\mathrm{H}}_\infty(X|Y) \geq k$, we know that $\mathbb{E}_{y \leftarrow Y} \max_x \Pr[X_y = x] \leq 2^k$. Applying Markov's inequality to the random variable $\max_x \Pr[X_y = x]$, there exists a set S'_Y such that $\Pr[y \in S'_Y] > 1/2$, and for all $y \in S'_Y$, $\mathrm{H}_\infty(X_y) \geq k - 1$ (we can use the strict version of Markov's inequality here, because the random variable $\max_x \Pr[X_y = x]$ is positive). Fix one value $y \in S_Y \cap S'_Y$ (which exists because the sum of probabilities of S_Y and S'_Y is greater than 1). Thus, for all such that $t', 1 \leq t' \leq t$,

$$\Pr_{x \leftarrow X_y}[x' \leftarrow \mathsf{Sample}(x, t') \wedge \mathsf{Rec}(x', z) = x] \geq 1 - 2(\epsilon + t\delta).$$

Thus, X_y is a $(t, 2(\epsilon + t\delta))$-average error Shannon code with recovery $\mathsf{Rec}(\cdot, y)$ and 2^{k-1} points. The statement of the theorem follows by application of Lemma 3.5. □

C.3 Proof of Theorem 3.10

Instead of proving the result just for Hamming metric over Z^n, we will prove the result for any metric space that is both neighborhood samplable (Definition 3.2) and where picking a random point in the space is easy. We now define this second condition:

Definition C.1. *A metric space space* $(\mathcal{M}, \mathsf{dis})$ *is* s_{sam}-efficiently-samplable *if there exists a randomized circuit* Sample *of size* s_{sam} *that outputs a uniformly random point in* $\mathcal{M}$.

Theorem C.2. *Let W be a distribution over a metric space $(\mathcal{M}, \mathsf{dis})$ that is s_{sam} samplable and (s_{neigh}, t) neighborhood samplable. Furthermore, assume that the number of points within distance t in $\mathcal{M}$ is at least some fixed value $B_t(\cdot)$. Let $(\mathsf{SS}, \mathsf{Rec})$ be an unpredictability-entropy $(\mathcal{M}, \mathrm{H}_\infty(W), \tilde{m}, t)$ secure sketch that is (ϵ, s_{sec})-secure with error δ. If $s_{sec} \geq \max\{t(|\mathsf{Rec}| + s_{neigh}), |\mathsf{Rec}| + s_{sam}\}$, then $\tilde{m} \leq \log|\mathcal{M}| - \log|B_t(\cdot)| + \log(1 - \epsilon - t\delta)$.*

Proof. Let (X, Y) be two random variables such that $\delta^{\mathcal{D}_{s_{sec}}}((W, \mathsf{SS}(W)), (X, Y)) \leq \epsilon$. It suffices to show that $\exists \mathcal{I}$ of size s_{sec} such that $\Pr[\mathcal{I}(Y) = X] \geq |\mathcal{M}|(1 - \epsilon - t\delta)/|B_t(\cdot)|$.

Let $B_t(x)$ denote the random variable representing a random neighbor of distance at most t from x (note that B_t may not be efficiently samplable, because we are assuming only that a neighbor a fixed distance is efficiently samplable). We begin by showing that Rec must recover points of X.

Claim C.3.

$$\Pr[\mathsf{Rec}(B_t(X), Y) = X] =$$
$$\Pr[(x, y) \leftarrow (X, Y) \wedge x' \leftarrow B_t(x) \wedge \mathsf{Rec}(x', y) = x] \geq 1 - \epsilon - t\delta.$$

Proof. Suppose that $\Pr[\mathsf{Rec}(B_t(X), Y) = X] < 1 - \epsilon - t\delta$. We construct the following distinguisher $D \in \mathcal{D}_{s_{sec}}$ (the distinguisher design is slightly complicated by the fact that we don't know at which particular distance t' the recover procedure is most likely to fail, so we have to try all distances):

- Input $w \in \mathcal{M}, s \in \{0,1\}^*$.
- For all $1 \leq t' \leq t$:

  $w' \leftarrow \mathsf{Neigh}(w, t')$.

  If $\mathsf{Rec}(w', z) \neq w$ output 0.
- Output 1.

First note that $|D| = t(|\mathsf{Rec}| + s_{neigh})$. Since $(\mathsf{SS}, \mathsf{Rec})$ has error δ we know that $\forall w, w' \in \mathcal{M}$ where $\mathsf{dis}(w, w') \leq t$

$$\Pr[s \leftarrow \mathsf{SS}(w) \wedge \mathsf{Rec}(w', s) = w] \geq 1 - \delta.$$

This implies that for all $1 \leq t' \leq t$, $\Pr[\mathsf{Rec}(\mathsf{Neigh}(W, t'), \mathsf{SS}(W)) = W)] \geq 1 - \delta$ and thus $\Pr[D(W, \mathsf{SS}(W)) = 1] \geq 1 - t\delta$. If $\Pr[\mathsf{Rec}(B_t(X), Y) = X] < 1 - \epsilon - t\delta$ there must exist at least one $1 \leq t' \leq t$ for which $\Pr[\mathsf{Rec}(\mathsf{Neigh}(X, t'), Y) = X] < 1 - \epsilon - t\delta$. Then

$$\Pr[D(W, \mathsf{SS}(W)) = 1] - \Pr[D(X, Y) = 1] \geq$$
$$(1 - t\delta) - \Pr[\mathsf{Rec}(\mathsf{Neigh}(X, t'), Y) = X] > (1 - t\delta) - (1 - t\delta - \epsilon) > \epsilon.$$

This is a contradiction and the statement of the claim follows. □

Now define $\mathcal{I}$ as follows:

- Input $y \in \{0,1\}^*$.
- Sample $x' \leftarrow \mathsf{Sample}$.
- Output $\mathsf{Rec}(x', y)$.

Note that $|\mathcal{I}| = |\mathsf{Rec}| + s_{sam}$. We now show that $\mathcal{I}$ predicts X:

$$\begin{aligned}
\Pr[\mathcal{I}(Y) = X] = & \\
&= \sum_{x,y \in \mathcal{M}} \Pr[(x,y) \leftarrow (X,Y)] \Pr[\mathcal{I}(y) = x] \\
&= \sum_{x,y \in \mathcal{M}} \Pr[(x,y) \leftarrow (X,Y)] \sum_{x' \in \mathcal{M}} \Pr[\mathsf{Sample} = x'] \Pr[\mathsf{Rec}(x',y) = x] \\
&\geq \sum_{x,y \in \mathcal{M}} \Pr[(x,y) \leftarrow (X,Y)] \sum_{x' \text{s.t.} \mathsf{dis}(x',x) \leq t} \Pr[\mathsf{Sample} = x'] \Pr[\mathsf{Rec}(x',y) = x] \\
&\geq \sum_{x,y \in \mathcal{M}} \Pr[(x,y) \leftarrow (X,Y)] \sum_{x' \text{s.t.} \mathsf{dis}(x',x) \leq t} \frac{|B_t(\cdot)|}{|\mathcal{M}|} \Pr[B_t(x) = x'] \Pr[\mathsf{Rec}(x',y) = x] \\
&\geq \frac{|B_t(\cdot)|}{|\mathcal{M}|}(1 - \epsilon - t\delta)
\end{aligned}$$

(the last step follows by Claim C.3).

Theorem 3.10 follows by noting that Z^n can be sampled in time $n \log |Z|$ and neighborhood sampled in time $n \log |Z|$. □

C.4 Proof of Lemma 4.6

Proof. Note that Decode_t will stop if w and w' agree on all the rows selected in Step 2 (it may also stop for other reasons—namely, in step 4; but we do not use this fact to bound the expected running time). The probability of each selected row having an error is at most $\frac{t}{m-i}$ where i is the number of rows already selected. That is,

$$\begin{aligned}
\Pr[i_1, ..., i_{2n} \text{ have no errors}] &\geq \prod_{i=0}^{2n-1} \left(1 - \frac{t}{m-i}\right) \geq \prod_{i=0}^{2n-1} \left(1 - \frac{d\left(\frac{m}{n} - 2\right)\log n}{m-i}\right) \\
&\geq \prod_{i=0}^{2n-1} \left(1 - \frac{d \log n}{n}\left(\frac{m-2n}{m-i}\right)\right) \geq \prod_{i=0}^{2n-1} \left(1 - \frac{d \log n}{n}\right) \\
&= \left(1 - \frac{d \log n}{n}\right)^{2n} = \left(\left(1 - \frac{d \log n}{n}\right)^{\frac{n}{d \log n}}\right)^{2d \log n} \geq \frac{1}{4^{2d \log n}} = \frac{1}{n^{4d}}.
\end{aligned}$$

(The second-to-last step holds as long as $n \geq 2d \log n$.) Because at each iteration, we select $2n$ rows independently at random, the expected number of iterations is at most n^{4d}; each iteration takes $O(n^3)$ operations in $\mathbb{F}_q$, which gives us the expected running time bound.

The probability that Decode_t outputs $\perp$ is bounded by

$$\begin{aligned}
\Pr[\mathsf{Decode}_t \to \perp] &\leq \sum_{j=1}^{\infty} \Pr[\mathsf{Decode}_t \to \perp \text{ in } j\text{th iteration of step 4}] \\
&= \sum_{j=1}^{\infty} \Pr[\mathsf{Decode}_t \text{ has not stopped after } j-1 \text{ iterations} \wedge \mathtt{rank}(\mathbf{A}_{i_1,\ldots,i_{2n}}) < n] \\
&\leq \sum_{j=1}^{\infty} \Pr[i_1,\ldots,i_{2n} \text{ had errors } j-1 \text{ times} \wedge \mathtt{rank}(\mathbf{A}_{i_1,\ldots,i_{2n}}) < n] \\
&= \sum_{j=1}^{\infty} \Pr[i_1,\ldots,i_{2n} \text{ had errors } j-1 \text{ times}] \cdot \Pr[\mathtt{rank}(\mathbf{A}_{i_1,\ldots,i_{2n}}) < n] \\
&\leq \sum_{j=1}^{\infty} \left(1 - \frac{1}{n^{4d}}\right)^{j-1} \cdot q^{-n} \\
&= n^{4d} e^{-\Omega(n)} = e^{-\Omega(n)}.
\end{aligned}$$

The third line from the bottom follows from the fact that the locations of the errors are assumed to be independent of the sketch, and therefore independent of the matrix $\mathbf{A}$. The second line from the bottom follows from Claim A.3 when $\beta = n$; note that, because we use the union bound and evaluate the probability separately for each value of j, we can treat $\mathbf{A}_{i_1,\ldots,i_{2n}}$ as a randomly chosen $2n \times n$ matrix, ignoring the fact that these matrices are correlated.

We claim that if the code generated by $\mathbf{A}$ has distance at least $2t+1$, then Decode_t will output $\perp$ or the correct $\mathbf{x}' = \mathbf{x}$. Indeed, suppose $\mathbf{x}' \neq \mathbf{x}$. Since $\mathbf{A}(\mathbf{x} - \mathbf{x}')$ has at least $2t+1$ nonzero coordinates by the minimum distance of the code generated by $\mathbf{A}$, and at most t of those can be zeroed out by the addition of $w - w'$, such an $\mathbf{x}'$ will not pass Step 6.

The probability that the code generated by $\mathbf{A}$ has distance lower than $2t+1$ is at most $e^{-\Omega(n)}$ (see Corollary A.2), the probability of outputting $\perp$ is also $e^{-\Omega(n)}$ (computed above). This gives the correctness bound for Decode_t. □

D Parameter Settings for Construction 4.1

In this section, we explain the different parameters that go into our construction. In Theorem 4.7 we give a lossless fuzzy extractor from a security parameter n and an error t. In this section, we discuss constraints imposed by 1) efficient decoding 2) maintaining security of the LWE instance and 3) ensuring no entropy loss of the construction. We begin by reviewing the parameters that make up our construction:

- $|W|$: The length of the source.
- t: Number of errors that can be supported.
- n: LWE security parameter (i.e., number of field elements in X), which must be greater than some minimum value n_0 for security.
- q: The size of the field.
- ρ: The fraction of the field needed for error sampling.

- m: The size of each number of samples in the LWE instance.
- k: The number of hardcore bits in X (from Lemma 4.4).

We will split the source $|W|$ into m blocks each of size $2\rho q + 1$ (that is, $|W| = m \log(2\rho q + 1)$). We will ignore the parameter $|W|$ and focus on t, n, q, ρ, and m. As stated above we have three constraints:

- Maintain security of LWE. If we assume GAPSVP and SIVP are hard to approximate within polynomial factors then Lemma 4.3 says that we get security for all n greater than some minimum n_0 and $q = \texttt{poly}(n)$ and $\rho q \geq 2n^{1/2+\sigma} m = \texttt{poly}(n)$. The only reason to increase ρq over this minimum amount (other than security) is if the number of errors in W decreases with a slightly larger block size. We ignore this effect and assume that $\rho q = 2n^{1/2+\sigma} m$.
- Maintain efficient decoding of Construction 4.5. Using Lemma 4.6, this means that $t \leq d \log n(m/n - 2)$.
- Minimize entropy loss of the construction. We will output $X_{1,\ldots,k}$ so the entropy loss of the construction is $|W| - |X_{1,\ldots,k}|$. We want the entropy loss to be zero, that is, $|W| = |X_{1,\ldots,k}|$. Substituting, one has $m \log 2\rho q + 1 = k \log q$.

Collecting constraints we can support any setting where t, n, q, ρ, m, k satisfy the following constraints (for constants d, f):

$$\begin{aligned} n_0 &< n - k \\ t &\leq d \log n \left(\frac{m}{n} - 2\right) \\ q &= n^f \\ \rho q &= 2n^{1/2+\sigma} m \\ m \log(2\rho q + 1) &= k \log q \end{aligned}$$

Substituting $q = n^f$ and $\rho q = 2n^{1/2+\sigma} m$ yields the following system of equations:

$$\begin{aligned} n_0 &< n - k \\ t &\leq d \log n \left(\frac{m}{n} - 2\right) \\ m \log(4n^{1/2+\sigma} m + 1) &= k \log n^f \end{aligned}$$

This is the most general form of our construction, we can support any n, t, m that satisfy these equations for constants d, f. However, the last equation may have no solution for f constant. Putting the last equation in terms of f one has:

$$\begin{aligned} n_0 &< n - k \\ t &\leq d \log n \left(\frac{m}{n} - 2\right) \\ f &= \frac{m}{k} \frac{\log 4n^{1/2+\sigma} m + 1}{\log n} \end{aligned}$$

To ensure f is a constant, we set $t = c \log n$ for some constant c and that $k = n/g$ for some constant $g > 1$. Finally we assume that m is the minimum value such that $t \leq d \log n(m/n - 2)$ (that is, there are only as many dimensions as necessary for decoding using Lemma 4.6):

$$\begin{aligned} n_0 &< n - k \\ m &= \frac{(c/d + 2)n \log n}{\log n} = (\frac{c}{d} + 2)n \\ f &= \frac{m}{k} \frac{\log 4n^{1/2+\sigma} m + 1}{\log n} = \frac{g(c + 2d)}{d} \frac{\log(\frac{4(c+2d)}{d} n^{3/2+\sigma} + 1)}{\log n} \end{aligned}$$

Note that f is at a constant in n. Assuming $n - k = n(1 - 1/g) > n_0$ and letting $t = c \log n$ we get the following setting:

$$\begin{aligned} m &= (\frac{c}{d} + 2)n \\ q &= n^f = n^{\frac{m}{k} \frac{\log(4n^{1/2+\sigma} m+1)}{\log n}} = \texttt{poly}(n) \\ \rho q &= 2n^{1/2+\sigma} m = 2(\frac{c}{d} + 2)n^{3/2+\sigma} \end{aligned}$$

Note, that $f > \frac{m}{k} \geq \frac{m}{n} \geq \frac{(c/d+2)n}{n} \geq 3$ as long as $d < c$ (this also ensures that $m \geq 3n$, as required for Lemma 4.6 to hold). Since $\rho q = 2n^{1/2+\rho} m = O(n^{5/2})$ in our setting $\rho = O(n^{-1/2})$. Thus, for large enough settings of parameters ρ is less than $1/10$ as required by Lemma 4.3.

Furthermore, we get decoding using $O(n^{4d+3})$ $\mathbb{F}_q$ operations. We can output a k fraction of X and the bits will be pseudorandom (conditioned on $\mathbf{A}, \mathbf{A}X + W$). The parameter g allows is a tradeoff between the number of dimensions needed for security and the size of the field q. In Theorem 4.7, we set $g = 2$ and output the first half of X. Setting $1 < g < 2$ achieves an increase in output length (over the input length of W). We also (arbitrarily) set $\sigma = 1/2$ to simplify the statement of Theorem 4.7, making $\rho q = 2(c/d + 2)n^2$.

D.1 Parameter Settings for Theorem 5.3

We repeat parameter settings for block fixing sources. We now have $m + \alpha$ as the number of samples, while $n + \alpha + \omega(1)$ is the number of variables. We can support any setting where $t, n, q, \rho, m, k, \alpha$ satisfy the following constraints (for $\beta = \omega(1)$ and constants d, f):

$$\begin{aligned} n_0 &< n - k - \alpha - \beta \\ t &\leq d \log n \left(\frac{m}{n} - 2\right) \\ q &= n^f \\ \rho q &= 2n^{1/2+\sigma} m \\ m \log(2\rho q + 1) &= k \log q \end{aligned}$$

Substituting $q = n^f$ and $\rho q = 2n^{1/2+\sigma} m$ yields the following system of equations:

$$\begin{aligned} n_0 &< n - k - \alpha - \beta \\ t &\leq d \log n \left(\frac{m}{n} - 2\right) \\ m \log(4n^{1/2+\sigma} m + 1) &= k \log n^f \end{aligned}$$

As before we can support any setting any n, t, m, α that satisfy these equations for $\beta = \omega(1)$ and constants d, f. However, the last equation may have no solution for f constant. Putting the last equation in terms of f one has:

$$\begin{aligned} n_0 &< n - k - \alpha - \beta \\ t &\leq d \log n \left(\frac{m}{n} - 2\right) \\ f &= \frac{m}{k} \frac{\log(4n^{1/2+\sigma}m + 1)}{\log n} \end{aligned}$$

To ensure f is a constant, we set $t = c \log n$ for some constant c and that $k, \alpha = n/3$ and $\beta = \log n$. Finally we assume that m is the minimum value such that $t \leq d \log n(m/n - 2)$ (that is, there are only as many dimensions as necessary for decoding using Lemma 4.6):

$$\begin{aligned} n_0 &< n/3 - \log n \\ m &= \frac{(c/d + 2)n \log n}{\log n} = (\frac{c}{d} + 2)n \\ f &= \frac{m}{k} \frac{\log(4n^{1/2+\sigma}m + 1)}{\log n} = \left(3(\frac{c}{d} + 2)\right) \frac{\log(4(\frac{c}{d} + 2)n^{3/2+\sigma} + 1)}{\log n} = O(1) \end{aligned}$$

Assuming $n/3 - \log(n) > n_0$ and letting $t = c \log n$ we get the following setting:

$$\begin{aligned} m &= (\frac{c}{d} + 2)n \\ q &= n^f = n^{\frac{m}{n} \frac{\log(4n^{1/2+\sigma}m+1)}{\log n}} = \texttt{poly}(n) \\ \rho q &= 2n^{1/2+\sigma}m = 2(\frac{c}{d} + 2)n^{3/2+\sigma} \end{aligned}$$

As before we arbitrarily set $\sigma = 1/2$, giving $\rho q = 2(\frac{c}{d} + 2)n^2$. Also, if $c < d$ then we get efficient decoding and $\rho = o(1)$ satisfying the condition of Lemma 4.3.