Soft decoding, dual BCH codes, and better list-decodable ε-biased codes[*]

Venkatesan Guruswami[†] Atri Rudra[‡]

Abstract

Explicit constructions of binary linear codes that are efficiently list-decodable up to a fraction $(1/2-\varepsilon)$ of errors are given. The codes encode k bits into $n = \text{poly}(k/\varepsilon)$ bits and are constructible and list-decodable in time polynomial in k and $1/\varepsilon$ (in particular, in our results ε need not be constant and can even be polynomially small in n). These results give the best known polynomial dependence of n on k and $1/\varepsilon$ for such codes. Specifically, they are able to achieve $n \leqslant O(k^3/\varepsilon^{3+\gamma})$ or, if a linear dependence on k is required, $n \leqslant \tilde{O}(k/\varepsilon^{5+\gamma})$, where $\gamma > 0$ is an arbitrary constant. The best previously known constructive bounds in this setting were $n \leqslant O(k^2/\varepsilon^4)$ and $n \leqslant O(k/\varepsilon^6)$. Non-constructively, a random linear encoding of length $n = O(k/\varepsilon^2)$ suffices, but no sub-exponential algorithm is known for list decoding random codes.

The construction with a cubic dependence on ε is obtained by concatenating the recent Parvaresh-Vardy (PV) codes with dual BCH codes, and crucially exploits the soft decoding algorithm for PV codes. The result with the linear dependence on k is based on concatenation of the PV code with an inner code of good minimum distance.

In addition to being a basic question in coding theory, codes that are list-decodable from a fraction $(1/2-\varepsilon)$ of errors for $\varepsilon \to 0$ are important in several complexity theory applications. For example, the construction with near-cubic dependence on ε yields better hardness results for the problem of approximating NP witnesses. In addition, the codes constructed have the property that all nonzero codewords have relative Hamming weights in the range $(1/2-\varepsilon, 1/2+\varepsilon)$; this ε-*biased* property is a fundamental notion in pseudorandomness.

[*]A preliminary conference version of this paper was presented at the *23rd Annual IEEE Conference on Computational Complexity (CCC 2008).*

[†]Computer Science Department, Carnegie Mellon University, Pittsburgh, PA. Part of this work was done while visiting the School of Mathematics, Institute for Advanced Study, Princeton, NJ. Research supported in part by a Packard fellowship, and NSF grants CCF-0343672, CCF-0953155. `guruswami@cmu.edu`

[‡]Department of Computer Science and Engineering, University at Buffalo, State University of New York, Buffalo, NY, 14620. Research supported in part by startup funds from University at Buffalo and by NSF CAREER award CCF-0844796. `atri@buffalo.edu`

1 Introduction

Consider the task of communicating k message bits of information over a channel that is going to adversarially corrupt a fraction $1/2 - \varepsilon$ of the transmitted bits. What is the fewest number $n = n(k, \varepsilon)$ codeword bits we need to communicate so that no matter which of the $(1/2 - \varepsilon)n$ bits are corrupted, we can recover the k message bits efficiently, i.e., in time polynomial in k and $1/\varepsilon$? The main goal of this paper is to understand the asymptotic dependence of n on k and ε. This natural coding-theoretic problem, where we want to correct close to the information-theoretically maximum possible fraction of errors, also arises in several applications of error-correcting codes in complexity theory including: construction of hardcore predicates from one-way functions [10, 31], approximating the VC dimension [23], constructions of extractors [28], membership comparability of NP-complete sets [30], hardness amplifications of Boolean functions [32], worst-case to average-case connection for the permanent [6], and approximating NP-witnesses [20, 9].

The process of recovering the transmitted message from the received corrupted codeword is called *decoding*. If we want the decoder to *uniquely* recover the transmitted codeword, then it is well known that one cannot recover from more than a $1/4$ fraction of errors. Unfortunately, all the applications mentioned above need to work with a fraction $1/2 - \varepsilon > 1/4$ of errors. In fact, in some cases ε could be polynomially small in n. In such high noise regimes, one has to settle with a relaxation of unique decoding called *list decoding*. Under list decoding, the decoder is allowed to output a (small) list of codewords with the guarantee that the transmitted codeword is present in the list. The fact that a list decoder can output a small list of codewords could be problematic since we need to recover the *actual* transmitted codeword. Fortunately, in all the applications mentioned above, the decoder has access to some side information, which allows it to go through the list and pick the transmitted codeword. The only constraint that this introduces is that the size of the list needs to be bounded by a polynomial in n. Note that this is also an *a priori* requirement for the list decoding algorithm to run in polynomial time.

Thus, the natural question to ask is: what is the smallest n one can achieve with a code that needs to be list decoded from a $1/2 - \varepsilon$ fraction of errors? It is well known that there exist codes with $n = O(k/\varepsilon^2)$ such that for every error pattern only $O(1/\varepsilon^2)$ many codewords need to be output; in fact a random code with 2^k codewords of such a block length has this property with high probability [33, 7]. Further, this bound is tight, that is, for $n = o(k/\varepsilon^2)$, for *any* code, there exists an error pattern for which a super-polynomial number of codewords need to be output. Thus, $n = \Theta(k/\varepsilon^2)$ is information-theoretically best possible. The upper bound of $n = O(k/\varepsilon^2)$ above is non-constructive: a code that is guaranteed to have these properties is not known to be efficiently computable, and even if we are content with a randomized Monte Carlo construction of such a code, there is no polynomial time list decoding algorithm known for such codes. This is a serious drawback since the applications need efficient constructions and decoding algorithms.

In summary, the applications of these codes demand that they be list-decodable from a $1/2 - \varepsilon$ fraction of errors in time polynomial in k/ε. Further, these codes need to be constructed in time polynomial in k/ε. In particular, the focus of this paper is on codes with construction and list decoding time complexity of $\text{poly}(k/\varepsilon)$ with as small a block length n as possible.

We should remark that in the complexity-theoretic applications, the exact dependence of n on k often does not matter as long as the exponent of k is some constant. However, in the traditional setting of coding theory and asymptotically good codes, the interesting regime is when the exponent of k equals one. On the other hand, in some complexity theory settings, one would like ε to be as small a function of n as possible. In particular, the goal of approaching the optimal $1/\varepsilon^2$ dependence of the block length n on ε becomes well-motivated in the context of those applications. An example of such an application of list-decodable codes is the hardness of approximating NP witnesses. Here the goal is to show that for NP languages, it is hard to even find a witness with non-trivial agreement (in terms of Hamming distance) with an actual witness. Since this application does not play a central role for the technical contribution of this work, we defer the details of this discussion to Appendix D.

1.1 Our Bounds and Comparison With Previously Best Known Bounds

Prior to this work, the following were the best known dependence of the block length n on k and ε for binary $[n, k]_2$ codes with $\text{poly}(k/\varepsilon)$ construction and list decoding time (for correcting a fraction $1/2 - \varepsilon$ of errors):

(a) By concatenating an outer Reed-Solomon code with an inner Hadamard code, a bound of $n = O(k^2/\varepsilon^4)$ was achieved in [15].

(b) By concatenating Reed-Solomon codes with a binary linear code of relative distance $(1/2 - O(\varepsilon^2))$, a bound of $n = O(k/\varepsilon^6)$ was obtained in [16]. This was a polynomial time *Las Vegas* construction (where the list decoding property of the code is *certified*).

Both the above results exploit the powerful soft decoding algorithm for Reed-Solomon codes (a.k.a., weighted polynomial reconstruction) from [14]. In the case when the time for construction and list decoding are allowed to be *exponential* in $1/\varepsilon$, the recent work on binary codes list-decodable up to the Zyablov bound yields a bound of $n = O(k/\varepsilon^3)$ [13]. Earlier, an encoding length of $n = O(k/\varepsilon^4)$ was obtained in [12], but the construction time was once again exponential in $1/\varepsilon$.

In this work, we improve the previous best known results by designing binary linear codes that achieve n to be about $O(k^3/\varepsilon^3)$ as well as $O(k/\varepsilon^5)$, each with a construction and decoding complexity of $\text{poly}(k/\varepsilon)$. In particular,

(1) By concatenating the Parvaresh-Vardy codes [25] with dual BCH codes (cf. [22]), we achieve $n = O\left(\frac{k^3}{\varepsilon^{3+\gamma}}\right)$ for any constant $\gamma > 0$.

(2) By concatenating the Parvaresh-Vardy codes [25] with a binary linear code of relative distance $(1/2 - O(\varepsilon^2))$, we achieve a bound of $n = O(k/\varepsilon^{5+\gamma})$ for any constant $\gamma > 0$. Using a recent result of Porat and Rothschild [26], we can construct such codes in deterministic $\text{poly}(k/\varepsilon)$ time.[1]

We remark that the Parvaresh-Vardy (PV) codes are a generalization of Reed-Solomon codes and the dual BCH codes are generalizations of Hadamard codes. Thus, for our result (1), we generalized both the outer and inner codes used in result (a) from [15] mentioned earlier. Interestingly, our improvement seems to crucially require generalizing *both* the outer and the inner codes. The

[1]In fact the $n = O(k/\varepsilon^6)$ result from [16] can also be achieved with a deterministic $\text{poly}(k/\varepsilon)$ construction time using the result from [26].

Hadamard code is too wasteful and we cannot afford to use it as inner code when the outer code is a PV code, since PV codes have a much larger alphabet size than Reed-Solomon codes. Dual BCH codes, which are generalized Hadamard codes, are much more efficient in encoding length. On the other hand, they necessarily provide weaker guarantees. In particular, the second moment of a certain coset weight distribution is bounded in the case of Hadamard codes (this follows from the Parseval's identity for Fourier coefficients, and was the key to the result in [15]). For duals of BCH codes with distance $(2t+1)$, an analogous bound holds only for the $2t$'th moment (this was shown in the work [17] on property testing of dual BCH codes). But quite remarkably, there is a *soft decoding* algorithm for PV codes that works under the weaker guarantee of bounded higher order moments, and this enables us to obtain our result (1). We stress that the power of the soft decoding algorithm for PV codes, and the ability to exploit it via meaningful weights passed from the dual BCH decoder, is crucial for our result (1).

Our result (2) achieving $n = O(k/\varepsilon^{5+\gamma})$ is simpler. It is based on list decoding the inner codes up to a radius of $(1/2 - O(\varepsilon))$ (each output list will have size at most $O(1/\varepsilon^2)$ by the Johnson bound). The resulting lists from the inner decoding stage are used by a list recovery algorithm for PV codes to complete the decoding. To achieve the deterministic poly(k/ε) construction time we use the recent result of Porat and Rothschild who present an $2^{O(K)}$ time deterministic algorithm to construct codes with dimension K and block length N on the Gilbert-Varshamov bound. (Previously, it was known how to construct such codes in $2^{O(N-K)}$ time, for example by greedily picking the columns of a parity check matrix, or see [5]. However, such a construction time is too expensive for our purposes.)

ε-biased codes. Our constructions also yield linear codes that are *ε-biased*, namely every nonzero codeword has Hamming weight in the range $[(1/2-\varepsilon)n, (1/2+\varepsilon)n]$. The rows of an $n \times k$ generator matrix of an ε-biased code are called an ε-biased set of length k strings. (We adopt the convention that the columns of the generator matrix span the code.) Our constructions yield explicit ε-biased sets of size $O\left(k^3/\varepsilon^{3+\gamma}\right)$ or $O\left(k/\varepsilon^{5+\gamma}\right)$ which are in addition list-decodable up to a fraction $(1/2-\varepsilon)$ of errors. The concept of an ε-biased set, which originated in the seminal work [24], is a fundamental one in pseudorandomness with a rich body of work and numerous applications.

The best non-constructive bound on the size of an ε-biased set of length k strings equals $O(k/\varepsilon^2)$. (It is also known that the size needs to be *at least* $\Omega(k/(\varepsilon^2 \log(1/\varepsilon)))$.) The best known explicit bounds for ε-biased sets are $O\left(\frac{k^2}{\varepsilon^2 \log^2(k/\varepsilon)}\right)$ [2] and $O\left(\frac{k}{\varepsilon^3 \log(1/\varepsilon)}\right)$ [1] that can be obtained by concatenating respectively Reed-Solomon codes and appropriate algebraic-geometric codes (that lie on the Tsfasman-Vlădut-Zink bound) with the Hadamard code. However, these codes have weaker decoding guarantees than the $(1/2-\varepsilon)$ radius we achieve; in particular, the best known algorithms based on passing soft information from decoding the inner Hadamard codes to outer algebraic soft-decision decoders correct only a fraction $\approx (1/2 - O(\sqrt{\varepsilon}))$ of errors [15]. Recently, an incomparable explicit upper bound of $n = O\left(\left(k/(\varepsilon^2 \log(1/\varepsilon))\right)^{5/4}\right)$ was obtained for ε-biased $[n,k]_2$ codes by concatenating Hermitian codes with the Hadamard code [3], but once again these codes are not known to be list-decodable up to a fraction $(1/2-\varepsilon)$ of errors.

We would like to point out that in some applications of ε-biased codes, ε needs to be exponentially small in k. For example, in construction of ε-biased pseudorandom generators, we need such a small ε. In particular, the list decodability is not important but one desires that any row of the corresponding generator matrix can be constructed in time $\mathrm{poly}(k, \log(1/\varepsilon))$. This property holds for our result (1) as both the outer and inner codes are explicit, i.e. any entry in the generator matrix can be constructed in time poly-logarithmic in the size of the matrix. On the other hand, our result (2) does not have this property as the inner code from [26] is not explicit– indeed such a result would imply explicit codes on the Gilbert-Varshamov bound, which is an outstanding open question in coding theory.

Remark 1 (Extension to non-binary alphabets)**.** Our construction with linear dependence of n on k based on concatenation of PV codes with inner codes meeting the Gilbert-Varshamov bound generalizes in a straightforward way to larger alphabets. However, our construction with near-cubic dependence on ε based on dual BCH codes at the inner level only works over the binary alphabet. Over a p-ary alphabet, BCH codes of block length n and designed distance d (where d is a fixed constant and n is allowed to grow) have dimension about $n - d(1-1/p)\log_p n$, so the dual BCH codes have size about $n^{d(1-1/p)}$. For $p > 2$, this is much larger than the $\approx n^{d/2}$ size of binary BCH code. As a result, the upper bound on the required moment of the coset weight distribution of non-binary dual BCH codes (corresponding to Lemma 9) is too weak for use in the soft decoding scheme.

Remark 2. Our constructions can also be based on folded Reed-Solomon codes, which were shown to achieve the optimal trade-off between list decoding radius and rate over large alphabets by the authors [13], in place of Parvaresh-Vardy codes. However, as our constructions are in the low-rate regime, using folded RS codes does not offer any asymptotic advantage, and so we state our results based on the simpler PV codes.

1.2 Organization of the Paper

We review some basic terminology from coding theory in Section 2. Our final construction combines different codes which are described along with their salient properties in Section 3. In particular, the outer Parvaresh-Vardy codes and their list decoding properties are discussed in Section 3.1 and the inner dual BCH codes and their necessary coset weight distribution properties are discussed in Section 3.4. We then present and analyze our two binary code constructions that are list decodable from a fraction $(1/2 - \varepsilon)$ of errors in Section 4. We present a simple lower bound showing a quadratic dependence of the block length n on ε in Section 5. We conclude the main body of the paper with some open questions in Section 6. The application of list-decodable codes to the hardness of approximating NP witnesses is discussed in Appendix D.

2 Coding Basics

In this section, we will quickly review the basic terminology concerning codes that we will use. A code of *dimension* k and *block length* n over an alphabet Σ is a subset of Σ^n of size $|\Sigma|^k$. The *rate* of such a code equals k/n. Each vector in C is called a codeword. In this paper, we will focus on the case when Σ is a finite field. We will denote by $\mathbb{F}_q$ the field with q elements, where q is a prime power. A code C over $\mathbb{F}_q$ is called a linear code if C is a subspace of $\mathbb{F}_q^n$. In this case the dimension of the code coincides with the dimension of C as a vector space over $\mathbb{F}_q$. By abuse of notation we will also think of a code C as a map from elements in $\mathbb{F}_q^k$ to their corresponding codeword in $\mathbb{F}_q^n$. If

C is linear, then the linear code corresponding to its orthogonal space is called the dual of C and is denoted by $C^{\perp}$.

The Hamming distance between two vectors $\mathbf{u}, \mathbf{v} \in \Sigma^n$, denoted by $\Delta(\mathbf{u}, \mathbf{v})$, is the number of places they differ in. The (minimum) distance of a code C is the minimum Hamming distance between any two pairs of distinct codewords from C. The relative distance is the ratio of the distance to the block length. The Hamming weight of a vector $\mathbf{u} \in \mathbb{F}_q^n$, denoted by $wt(\mathbf{u})$, is the number of non-zero positions in $\mathbf{u}$. A linear code over $\mathbb{F}_q$ of block length n, dimension k, and minimum distance d will be denoted compactly as an $[n, k, d]_q$ code; when the distance d is omitted, such a code will be referred to as an $[n, k]_q$ code.

A list decoding algorithm for a code C of block length n needs to do the following. Given an error parameter $0 \leqslant \rho < 1$ and a received word $\mathbf{y} \in \Sigma^n$ the decoder needs to output all codewords $\mathbf{c} \in C$ such that $\Delta(\mathbf{c}, \mathbf{y}) \leqslant \rho n$. We define the associated notion of a code being list decodable as follows. Note that this definition is a combinatorial one and does not address the efficiency of the algorithmic list decoding task.

Definition 1. *A code $C \subseteq \Sigma^n$ of block length n is (ρ, L)-list-decodable, if for every $\mathbf{y} \in \Sigma^n$, there are at most L codewords which satisfy $\Delta(\mathbf{c}, \mathbf{y}) \leqslant \rho n$.*

Code Concatenation. Concatenated codes are constructed from two different kinds of codes that are defined over alphabets of different sizes. Say we are interested in a code over a finite field $\mathbb{F}_p$ (in this paper, we will present all our results for $p = 2$). Then the *outer code* C_{out} is defined over $\mathbb{F}_q$, where $q = p^k$ for some positive integer k and has block length N. The second type of code, called the *inner code*, which is denoted by C_{in} is defined over $\mathbb{F}_p$ and is of dimension k (note that the message space of C_{in} and the alphabet of C_{out} have the same size). The concatenated code, denoted by $C = C_{\text{out}} \circ C_{\text{in}}$, is defined as follows. Let the rate of C_{out} be R and let the block length of C_{in} be n. Define $K = RN$ and $r = k/n$. The input to C is a vector $\mathbf{m} = \langle m_1, \ldots, m_K \rangle \in (\mathbb{F}_{p^k})^K$. Let $C_{\text{out}}(\mathbf{m}) = \langle x_1, \ldots, x_N \rangle$. The codeword in C corresponding to $\mathbf{m}$ is defined as follows

$$C(\mathbf{m}) = \langle C_{\text{in}}(x_1), C_{\text{in}}(x_2), \ldots, C_{\text{in}}(x_N) \rangle.$$

It can be verified that C has rate, dimension and block length of rR, kK and nN. Further, the minimum distance of C is at least the product of the distances of C_{out} and C_{in}.

3 Coding Ingredients

3.1 Parvaresh-Vardy codes and their soft decoding

The following theorem can be deduced via a straightforward adaptation of the work of Parvaresh and Vardy [25] to the "soft-decoding" setting. The case $s = 1$ of the statement was shown by Guruswami and Sudan [14] for Reed-Solomon codes. Let us first define the codes constructed by Parvaresh and Vardy.

Definition 2. *For an integer parameter $s > 1$, the Parvaresh-Vardy (PV) code for $s > 1$ encodes a polynomial $f(X)$ of degree less than K by the evaluations of $f(X)$, and $s-1$ correlated polynomials*

$$f_j(X) = (f(X))^{h^j} \pmod{E(X)} \quad \textit{for} \quad 1 \leqslant j \leqslant s-1 \ ,$$

at N distinct elements of $\mathbb{F}_q$. Here $E(X)$ is an arbitrary irreducible polynomial over $\mathbb{F}_q$ of degree K, and h is an integer parameter (which has to be large enough for the list decoding guarantee).

Note that when h is a power of p^ℓ where $p = \mathrm{char}(\mathbb{F}_q)$ and $\ell \geqslant 1$ is an arbitrary integer, the above encoding is $\mathbb{F}_{p^\ell}$-linear (this will be important for us to get *linear* codes in the end, after concatenation with a suitable linear inner code).

Theorem 1. *For all integers $s \geqslant 1$, for all prime powers r and all powers q of r, every pair of integers $1 < K \leqslant N \leqslant q$, there is an explicit $\mathbb{F}_r$-linear map $E : \mathbb{F}_q^K \to \mathbb{F}_{q^s}^N$ such that:*

1. *The image of E, $C \subseteq \mathbb{F}_{q^s}^N$, is a code of minimum distance at least $N - K + 1$.*
2. *Let $W = (s+1)! \sum_{i,\alpha} \binom{w_{i,\alpha}+s}{s+1}$ and $L = \left(r \cdot \left(\left(\frac{W}{K-1}\right)^{1/(s+1)} + 2\right)\right)^s$. There is a list decoding algorithm, that given a collection of nonnegative* integers *$\{w_{i,\alpha}\}$, $1 \leqslant i \leqslant N$, $\alpha \in \mathbb{F}_{q^s}$, runs in time* $\mathrm{poly}(q, L)$*, and outputs a list of size at most L that includes precisely the set of codewords $(c_1, \ldots, c_N) \in C$ that satisfy*

$$\sum_{i=1}^{N} w_{i,c_i} > \left\lceil ((K-1)^s W)^{\frac{1}{s+1}} \right\rceil + 1 \ . \qquad (1)$$

Since the theorem is not proved in the literature in the exact form stated above, for the sake of completeness, we sketch the proof of Theorem 1 in Appendix A.

We now state a version that holds for *all* nonnegative weights (and eliminates the pseudopolynomial dependence on the weights). It is obtained by a suitable scaling and rounding of the weights to convert them to integer weights and then applying the above statement. A similar proof for RS codes ($s = 1$) appears in [11, Chap. 6]. For the sake of completeness, we include a proof below.

Theorem 2. *For all integers $s \geqslant 1$, for all prime powers r and all powers q of r, every pair of integers $1 < K \leqslant N \leqslant q$, there is an explicit $\mathbb{F}_r$-linear map $E : \mathbb{F}_q^K \to \mathbb{F}_{q^s}^N$ such that:*

1. *The image of E, $C \subseteq \mathbb{F}_{q^s}^N$, is a code of minimum distance at least $N - K + 1$.*
2. *Let $\{w_{i,\alpha}\}$, $1 \leqslant i \leqslant N$, $\alpha \in \mathbb{F}_{q^s}$ be a collection of non-negative weights. Let $w_{max} = \max_{i,\alpha} w_{i,\alpha}$ and $W' = \left(\sum_{i,\alpha} w_{i,\alpha}^{s+1}\right)/w_{\max}^{s+1}$. Then for every $\zeta > 0$ there is an algorithm, that given $\{w_{i,\alpha}\}$, $1 \leqslant i \leqslant N$, $\alpha \in \mathbb{F}_{q^s}$, runs in time bounded by $(rs/\zeta)^{O(s)}(W'q)^{O(1)}$, and outputs a list of size at most $(rs)^{O(s)}W'/\zeta^s$ that includes precisely the set of codewords $c \in C$ satisfying*

$$\sum_{i=1}^{N} w_{i,c_i} > (s+1)\left((K-1)^s \sum_{i,\alpha} w_{i,\alpha}^{s+1}\right)^{\frac{1}{s+1}} + \zeta \cdot (N+2) w_{max} \ .[2] \qquad (2)$$

[2]We can make the multiplicative factor of $(s+1)$ a constant arbitrarily close to 1 with suitable choice of parameters. For the low-rate regime we are interested in, this does not make much difference so we state here a crude bound that suffices.

***Proof*.** The linear map E is the one from Theorem 1. If all the weights equal 0, then the condition (2) cannot be satisfied for any codeword, so assume that not all the $w_{i,\alpha}$'s are zero. As mentioned above, the idea is to scale the weights by multiplication by a large constant to convert them to integers and then use the guarantee of Theorem 1.

Let B be a large enough integer (the choice $B = \lceil \frac{1}{\zeta} \rceil$ will do). Set $w'_{i,\alpha} = \left\lfloor \frac{Bw_{i,\alpha}}{w_{\max}} \right\rfloor$. Note that $\frac{Bw_{i,\alpha}}{w_{max}} - 1 \leqslant w'_{i,\alpha} \leqslant \frac{Bw_{i,\alpha}}{w_{max}}$. Each $w'_{i,\alpha}$ is a nonnegative integer that is at most B. Clearly the condition (1) will be satisfied by a certain codeword $\mathbf{c} = (c_1, \ldots, c_N)$ with respect to the integer weights $w'_{i,\alpha}$ if the following condition is met:

$$\sum_{i=1}^{N} \left(\frac{Bw_{i,c_i}}{w_{\max}} - 1 \right) > \sqrt[s+1]{(K-1)^s (s+1)! \sum_{i,\alpha} \binom{w'_{i,\alpha} + s}{s+1}} + 2.$$

Now if $w'_{i,\alpha} = 0$ then $\binom{w'_{i,\alpha}+s}{s+1} = 0$. Since $w'_{i,\alpha}$ is an integer $(s+1)!\binom{w'_{i,\alpha}+s}{s+1} \leqslant \Big(w'_{i,\alpha}(s+1)\Big)^{s+1}$. Thus, the condition above is met if

$$\sum_{i=1}^{N} \left(\frac{Bw_{i,c_i}}{w_{\max}} - 1 \right) > (s+1) \sqrt[s+1]{(K-1)^s \sum_{i,\alpha} \left(\frac{Bw_{i,\alpha}}{w_{max}} \right)^{s+1}},$$

which in turn is equivalent to the condition

$$\sum_{i=1}^{N} w_{i,c_i} > (s+1) \sqrt[s+1]{(K-1)^s \sum_{i,\alpha} w_{i,\alpha}^{s+1}} + \frac{(N+2)w_{max}}{B}.$$

By the choice of B, this is implied by (2).

It remains to justify the claims about the runtime and output list size. As as we have seen before, $W = (s+1)! \sum_{i,\alpha} \binom{w'_{i,\alpha}+s}{s+1}$ is at most $(s+1)^{s+1} \sum_{i,\alpha} (w'_{i,\alpha})^{s+1}$. As $w'_{i,\alpha} \leqslant Bw_{i,\alpha}/w_{max}$,

$$W \leqslant \left(\frac{B(s+1)}{w_{max}} \right)^{s+1} \sum_{i,\alpha} w_{i,\alpha}^{s+1} \leqslant \left(\frac{2s}{\zeta} \right)^{s+1} W'$$

by the choice of B and definition of W'. The claimed bounds on runtime and list size now follow from those of Theorem 1. ■

A List Recovery Bound for Parvaresh-Vardy Codes. The following is a corollary of Theorem 1 using only 0, 1 weights $w_{i,\alpha}$, namely $w_{i,\alpha} = 1$ if $\alpha \in S_i$ (for some subsets $S_i \subseteq \mathbb{F}_{q^s}$ for $1 \leqslant i \leqslant N$) and 0 otherwise. This variant is called list recovery in the literature.

Corollary 3. *For all integers $s \geqslant 1$, for all prime powers r and all powers q of r, every pair of integers $1 < K \leqslant N \leqslant q$, there is an explicit $\mathbb{F}_r$-linear map $E : \mathbb{F}_q^K \to \mathbb{F}_{q^s}^N$ such that:*

1. *The image of E, $C \subseteq \mathbb{F}_{q^s}^N$, is a code of minimum distance at least $N - K + 1$.*

2. *There is an algorithm, that given a collection of subsets $S_i \subseteq \mathbb{F}_{q^s}$ for $i = 1, 2, \ldots, N$, each of size at most ℓ, runs in* $\mathrm{poly}((rs)^s, q, \ell)$ *time, and outputs a list of size at most $O((rs)^s N\ell/K)$ that includes precisely the set of codewords $(c_1, \ldots, c_N) \in C$ that satisfy $c_i \in S_i$ for at least αN values of i, provided*
$$\alpha > (s+1)(K/N)^{s/(s+1)}\ell^{1/(s+1)} \ . \tag{3}$$

3.2 Johnson Bound

The following well known result called the Johnson bound states that a code of good minimum distance also has good list-decodability. We state it only for binary codes. Proofs can be found, for example, in [15] and [11, Chap. 3].

Lemma 4. *Let C be a binary code of block length n and relative distance $\delta < 1/2$. Then, for every $L \geqslant 1$, C is*
$$\left(\frac{1}{2}\left(1 - \sqrt{1 - 2\delta + \frac{2\delta}{L}}\right), L\right) \textit{-list-decodable} \ .$$

In particular, the above implies that a code of relative distance $(1/2 - \varepsilon^2)$ is $(1/2 - \varepsilon, \frac{1}{\varepsilon^2})$-list-decodable, a fact we will make use of later on (in Theorem 14). Note that this is a combinatorial guarantee– such a code need not be efficiently list decodable.

3.3 Codes on the Gilbert-Varshamov Bound

We will need the following recent result due to Porat and Rothschild:

Theorem 5 ([26])**.** *Let q be a prime power and let $n \geqslant 1$ be a large enough integer. Then for any $d < (1-1/q)n$, $\varepsilon > 0$ and $k \leqslant \lfloor (1-H_q(d/n))n \rfloor$, where $H_q(x) = -x\log_q(x/(q-1)) - (1-x)\log_q(1-x)$ is the q-ary entropy function, there exists a deterministic* $\mathrm{poly}\left(q^k, n\right)$ *algorithm to compute an $[n, k, d]_q$ code.*

The above implies the following result:

Corollary 6. *Let $k \geqslant 1$ be a large enough integer and let $\varepsilon > 0$ be a real. Then in* $\mathrm{poly}\left(2^k, k/\varepsilon\right)$ *time one can construct an ε^2-biased $[O(k/\varepsilon^4), k]_2$ code.*

The claim about the bias of the code was not stated explicit in [26] but their algorithm can be easily modified to make sure that the constructed code also has this extra property.

3.4 Dual BCH Codes

3.4.1 Weight Distribution and Duals of Linear Codes

In this subsection, we will recollect some known results on linear codes. Given a binary linear code C of block length n, and $\mathbf{y} \in \mathbb{F}_2^n$, let $C_\mathbf{y}$ denote the code $C \cup (C + \mathbf{y})$, where $\mathbf{y} + C \overset{def}{=} \{\mathbf{c} + \mathbf{y} | \mathbf{c} \in C\}$ is a *coset* of C. For any $0 \leqslant w \leqslant n$, we will use $A_w(C)$ to denote the number of codewords of Hamming weight w.

If $f(X)$ is a univariate polynomial over the reals, then we have

$$\mathop{\mathbf{E}}_{\mathbf{c}\in C}[f(wt(\mathbf{c}))] = \frac{1}{|C|}\cdot\sum_{i=0}^{n} A_i(C)f(i). \tag{4}$$

Our goal will be to compute the expectation of a certain low-degree polynomial over a coset of a linear code. The following lemma is handy for this task.

Lemma 7 ([18])**.** *Let C be a binary linear code of block length n and let $f(X)$ be a polynomial over the reals. The following holds for every $\mathbf{y}\in\mathbb{F}_2^n$:*

$$\mathop{\mathbf{E}}_{\mathbf{c}\in C}[f(wt(\mathbf{y}+\mathbf{c}))] = 2\mathop{\mathbf{E}}_{\mathbf{c}'\in C_{\mathbf{y}}}\left[f(wt(\mathbf{c}'))\right] - \mathop{\mathbf{E}}_{\mathbf{c}\in C}[f(wt(\mathbf{c}))].$$

***Proof*.** Indeed, by definition

$$\mathop{\mathbf{E}}_{\mathbf{c}'\in C_{\mathbf{y}}}\left[f(wt(\mathbf{c}'))\right] = \frac{1}{2}\mathop{\mathbf{E}}_{\mathbf{c}\in C}[f(wt(\mathbf{c}))] + \frac{1}{2}\mathop{\mathbf{E}}_{\mathbf{c}\in C+\mathbf{y}}[f(wt(\mathbf{c}))].$$

■

We will make crucial use of the following result from [17], which states that the weight distribution of a linear code appears binomial to all polynomials of degree bounded by the distance of its dual. For the sake of completeness, we present its proof in the Appendix B.

Lemma 8 ([17])**.** *Let $t\geqslant 1$ be an integer. Let C be a binary linear code of block length n such that $C^{\perp}$ has minimum distance at least $2t+1$. Then for every polynomial $f(X)$ of degree at most $2t$ over the reals:*

$$\sum_{i=0}^{n} A_i(C)f(i) = \frac{|C|}{2^n}\sum_{i=0}^{n}\binom{n}{i}f(i).$$

The above result implies the following:

Lemma 9. *Let $t\geqslant 1$ be an integer. Let C be a binary linear code of block length n such that $C^{\perp}$ has minimum distance at least $2t+1$. Then the following holds for every $\mathbf{y}\in\mathbb{F}_2^n$:*

$$\sum_{\mathbf{c}\in C}\left(1-2\frac{wt(\mathbf{y}+\mathbf{c})}{n}\right)^{2t} \leqslant 2\cdot\frac{|C|}{n^t}\cdot\frac{(2t)!}{2^t t!}.$$

***Proof*.** By definition, $C\subseteq C_{\mathbf{y}}$ and thus, $C_{\mathbf{y}}^{\perp}\subseteq C^{\perp}$. In other words, $C_{\mathbf{y}}^{\perp}$ has a minimum distance of at least $2t+1$. This implies that by Lemma 8, the following are true:

$$\sum_{i=0}^{n} A_i(C_{\mathbf{y}})\left(1-\frac{2i}{n}\right)^{2t} = \frac{|C_{\mathbf{y}}|}{2^n}\sum_{i=0}^{n}\binom{n}{i}\left(1-\frac{2i}{n}\right)^{2t},$$

$$\sum_{i=0}^{n} A_i(C)\left(1-\frac{2i}{n}\right)^{2t} = \frac{|C|}{2^n}\sum_{i=0}^{n}\binom{n}{i}\left(1-\frac{2i}{n}\right)^{2t}.$$

The above along with Lemma 7 and (4), implies that

$$\mathop{\mathbf{E}}_{\mathbf{c}\in C}\left[\left(1-2\frac{wt(\mathbf{y}+\mathbf{c})}{n}\right)^{2t}\right]=\frac{1}{2^n}\sum_{i=0}^{n}\left(1-\frac{2i}{n}\right)^{2t}.$$

The lemma follows from the above relation and the following inequality (which is proved in Appendix A of [17]):

$$\frac{1}{2^n}\sum_{i=0}^{n}\binom{n}{i}(n-2i)^{2t}\leqslant 2n^t\frac{(2t)!}{2^t t!}.$$

■

3.4.2 Moments of Coset Weight Distribution of Dual BCH Codes

Let $t,m\geqslant 1$ be integers such that $2t-2<2^{m/2}$. Recall that an $[2^m-1,2^m-2t-1,2t+1]_{2^m}$ Reed-Solomon code C_1 is the evaluation of degree 2^m-2t-1 polynomials over all the non-zero elements in $\mathbb{F}_{2^m}$. Let $C_{BCH(t,m)}$ denote the code $C_1\cap\mathbb{F}_2^{2^m-1}$. $C_{BCH(t,m)}$ is the well known BCH code.

Theorem 10 ([22])**.** *Let $m,t\geqslant 1$ be integers such that $2t-2<2^{m/2}$. Then $C_{BCH(t,m)}$ is an $[2^m-1,2^m-mt-1,d]_2$ code, where $d\geqslant 2t+1$.*

Let $C_{dBCH(t,m)}=(C_{BCH(t,m)})^{\perp}$. Thus, $C_{dBCH(t,m)}$ is an $[n\overset{def}{=}2^m-1,k\overset{def}{=}mt]_2$ code. Note that Lemma 9 can be applied to $C_{dBCH(t,m)}$. Further, $|C_{dBCH(t,m)}|=2^{mt}\leqslant(2(2^m-1))^t=2^t n^t$. Thus, by Lemma 9 we have

Lemma 11. *Let $m,t\geqslant 1$ be integers such that $2t-2<2^{m/2}$. Then the following is true for any $\mathbf{y}\in\mathbb{F}_2^n$:*

$$\sum_{\mathbf{c}\in C_{dBCH(t,m)}}\left(1-2\frac{\Delta(\mathbf{y},\mathbf{c})}{n}\right)^{2t}\leqslant\frac{2(2t)!}{t!}.$$

We will also need the following property of the dual BCH code.

Theorem 12 (Weil-Carlitz-Uchiyama Bound, [22])**.** *Let $m,t\geqslant 1$ be integers such that $2t-2<2^{m/2}$. Then for every non-zero $\mathbf{c}\in C_{dBCH(t,m)}$,*

$$2^{m-1}-(t-1)2^{m/2}\leqslant wt(\mathbf{c})\leqslant 2^{m-1}+(t-1)2^{m/2}.$$

4 Binary Codes List Decodable in Presence of High Noise

4.1 Construction With Near-Cubic Dependence on ε

We will prove the following result in this section.

Theorem 13. *Given an integer $\mathcal{K} > 1$ and reals $\gamma > 0$ and $0 < \varepsilon < 1/2$, the following holds. There exists an explicit binary linear code of dimension at least $\mathcal{K}$ and block length $\mathcal{N}$ at most $\left(\frac{1}{\gamma}\right)^{O(1)} \cdot \left(\frac{\mathcal{K}^3}{\varepsilon^{3+\gamma}}\right)$ with the following properties:*

(i) There is an algorithm that can list decode the code from a fraction $(1/2-\varepsilon)$ of errors in time at most $\left(\frac{\mathcal{K}}{\gamma\varepsilon}\right)^{O(1/\gamma)}$ and outputs a list of size at most $\left(\frac{1}{\gamma\varepsilon}\right)^{O(1/\gamma)}$.

(ii) The code is $\frac{5\varepsilon}{6}$-biased, i.e., all the codewords have Hamming weights in the range $\left[(\frac{1}{2}-\frac{5\varepsilon}{6})\mathcal{N}, (\frac{1}{2}+\frac{5\varepsilon}{6})\mathcal{N}\right]$.

***Proof*.** The idea is to concatenate the Parvaresh-Vardy codes guaranteed by Theorem 2 with an appropriate dual BCH code. The decoding algorithm will decode the inner blocks (by brute-force) and pass appropriate soft information to the outer decoder of Theorem 2. The details follow. Since the proof is long, we break it into stages. We start with the parameter choices which have to made carefully.

Parameter choices. Given $\mathcal{K}$, ε and γ we will pick positive integer parameters r, q, K, N, s, t and m (for use in Theorem 2 and choosing the inner dual BCH code) as follows.

We pick $K = \left\lceil \frac{2\mathcal{K}}{\log \mathcal{K}} \right\rceil$. Let $t = \left\lceil \frac{1+\gamma}{2\gamma} \right\rceil$, and $s = 2t-1$.

Let $r = 2$ and q be the smallest power of 2 larger than $\left(\frac{2(2t)!}{t!}\right)^{1/s} \frac{K}{(\varepsilon/(2s+2))^{1+1/s}}$. In particular,

$$\left(\frac{2(2t)!}{t!}\right)^{1/s} \frac{K}{(\varepsilon/(2s+2))^{1+1/s}} \leqslant q \leqslant 2\left(\frac{2(2t)!}{t!}\right)^{1/s} \frac{K}{(\varepsilon/(2s+2))^{1+1/s}}. \tag{5}$$

The reason for the specific choice of q is to ensure a decoding radius of $(1/2-\varepsilon)$ and will become clear at the end of the proof.

Finally we take $N = q$ and define $m = \left\lceil \frac{s \log q}{t} \right\rceil$.

Code construction. Let C_{out} be the $\mathbb{F}_2$-linear code over $\mathbb{F}_q$ from Theorem 2 with dimension K and block length N for parameter s. Let $C_{dBCH(t,m)}$ be the $[n = 2^m-1, k = mt]_2$ dual BCH code. We must verify that with our choice of parameters, $2t-2 < 2^{m/2}$ holds, so that the condition of Theorem 10 is met. To check this, note that $2^{m/2} \geqslant q^{s/(2t)} \geqslant (2s+2)^{(1+1/s)\cdot s/(2t)} = 2s+2 > 2t-2$, where the relationships follow from the choices of m, q and s respectively. Note that $k \geqslant s \log q$, and we take C_{in} to be an arbitrary subspace of $C_{dBCH(t,m)}$ of dimension $k' = s\log q$.

The binary code C with the claimed properties of Theorem 13 will be the concatenated code $C_{\text{out}} \circ C_{\text{in}}$. The claim on explicitness of C follows from the fact that both C_{out} and C_{in} are explicit codes. We begin by verifying the parameters of C.

Rate of C. The dimension of C equals $Ks\log q$ which is clearly at least $\mathcal{K}$ by the choice of K and q. The block length $n = 2^m - 1$ of C_{in} satisfies $q^{2-1/t} - 1 \leqslant n \leqslant 2q^{2-1/t}$. Now, the block length of C satisfies

$$\mathcal{N} = nN = nq \leqslant 2q^{3-1/t} \leqslant 16\left(\frac{2(2t)!}{t!}\right)^{(3t-1)/(ts)} \frac{K^{3-1/t}}{(\varepsilon/(2s+2))^{(3-1/t)(1+1/s)}}. \qquad (6)$$

It can be verified that by our choices of s, t and K, $K^{3-1/t}$ is $O(\mathcal{K}^3)$, $(3-1/t)(1+1/s) \leqslant 3+\gamma$ and $16\left(\frac{2(2t)!}{t!}\right)^{(3t-1)/(ts)}(2s+2)^{(3-1/t)(1+1/s)}$ is $(1/\gamma)^{O(1)}$, which proves the claimed bound on $\mathcal{N}$.

ε-biasedness of C. We now verify the claim that C is $\frac{5\varepsilon}{6}$-biased. First, note that since C_{out} is $\mathbb{F}_2$-linear and C_{in} is linear, the concatenated code is a binary linear code. (5) and the fact that $N = q$ implies that $K/N \leqslant \varepsilon$. This along with Theorem 2 implies that C_{out} has relative distance at least $1-\varepsilon$. Thus, by Theorem 12, all the non-zero codewords in C have relative Hamming weights in the range $\left[(1-\varepsilon)\left(\frac{2^{m-1}}{2^m-1}-\frac{(t-1)2^{m/2}}{2^m-1}\right), \frac{2^{m-1}}{2^m-1}+\frac{(t-1)2^{m/2}}{2^m-1}\right]$. We first claim that $\left[\frac{2^{m-1}}{2^m-1}-\frac{(t-1)2^{m/2}}{2^m-1}, \frac{2^{m-1}}{2^m-1}+\frac{(t-1)2^{m/2}}{2^m-1}\right] \subseteq \left[\frac{1}{2}-\frac{t+1}{2^{m/2}}, \frac{1}{2}+\frac{t+1}{2^{m/2}}\right]$. One can check that this inclusion is satisfied if

$$2^{m/2+1} \geqslant \frac{t+1}{2^{m/2}} + \frac{1}{2}.$$

Using the fact that $2t-2 < 2^{m/2}$, the above is satisfied if $2^{m/2+1} \geqslant 1 + 2/2^{m/2}$, which is true for $m \geqslant 1$. The claim on C being $\frac{5\varepsilon}{6}$-biased follows from the fact that $\frac{t+1}{2^{m/2}} \leqslant \varepsilon/3$. The latter inequality is easily checked, since

$$2^{m/2} \geqslant q^{s/(2t)} = q^{1-1/(2t)} \geqslant \frac{1}{(\varepsilon/(2s+2))^{(1+1/s)(1-1/(2t))}} = 4t/\varepsilon \geqslant 3(t+1)/\varepsilon\ ,$$

where the last inequality follows from the fact that $t \geqslant 1$.

List decoding algorithm. Finally, we turn to the claims on list decodability of C. The list decoding algorithm for C is as follows. Let $\mathbf{y} = (\mathbf{y}_1, \ldots, \mathbf{y}_N) \in (\mathbb{F}_2^n)^N$ be the received word. We now define appropriate weights for use in the soft decoding algorithm from Theorem 2. For every $1 \leqslant i \leqslant N$ and $\alpha \in \mathbb{F}_{q^s}$ define:

$$w_{i,\alpha} = \max\left(1 - \frac{2\Delta(\mathbf{y}_i, C_{\text{in}}(\alpha))}{n}, 0\right). \qquad (7)$$

Define $w_{\max} = \max_{i,\alpha} w_{i,\alpha}$. (Since $\mathbf{y}$ is promised to have a codeword of C within distance $\mathcal{N}/2$, $w_{\max} > 0$.) Finally, run the soft decoder from Theorem 2 with the choice

$$\zeta = \frac{\varepsilon}{3w_{\max}}\ . \qquad (8)$$

Correctness of algorithm. We will defer the analysis of the running time and worst case list size to the end of the proof. First, we show that the algorithm above outputs a list that contains $\mathbf{c}'$, for any codeword $\mathbf{c}' \in C$ such that $\Delta(\mathbf{c}', \mathbf{y}) \leqslant (1/2-\varepsilon)\mathcal{N}$.

As $s+1=2t$ is even, $w_{i,\alpha}^{s+1} \leqslant \left(1-2\frac{\Delta(\mathbf{y}_i, C_{\text{in}}(\alpha))}{n}\right)^{s+1}$. Since C_{in} is a subcode of the dual BCH code $C_{dBCH(t,m)}$ (and $2t-2 < 2^{m/2}$), Lemma 11 implies the following property of these weights (for every $1 \leqslant i \leqslant N$):

$$\sum_{\alpha \in \mathbb{F}_{q^s}} w_{i,\alpha}^{s+1} \leqslant \frac{2(2t)!}{t!}. \tag{9}$$

Let $\mathbf{c} = (c_1, \ldots, c_N) \in C_{\text{out}}$ be the intermediate codeword corresponding to $\mathbf{c}'$. Thus, if we use the soft decoding algorithm from Theorem 2 with the choice $\zeta = \frac{\varepsilon}{3w_{max}}$, it will output $\mathbf{c}$ if the following condition is satisfied:

$$\sum_{i=1}^{N} w_{i,c_i} \geqslant (s+1) \sqrt[s+1]{K^s \sum_{i=1}^{N} \sum_{\alpha \in \mathbb{F}_{q^s}} w_{i,\alpha}^{s+1}} + \frac{\varepsilon N}{2},$$

where we have assumed that $N \geqslant 4$ so that $\varepsilon N/2 \geqslant \varepsilon(N+2)/3$. Now as $w_{i,\alpha} \geqslant \left(1-2\frac{\Delta(\mathbf{y}_i, C_{\text{in}}(\alpha))}{n}\right)$, the above condition is satisfied if the following is true:

$$\sum_{i=1}^{N} \left(1-2\frac{\Delta(\mathbf{y}_i, C_{\text{in}}(c_i))}{n}\right) \geqslant (s+1) \sqrt[s+1]{K^s \sum_{i=1}^{N} \sum_{\alpha \in \mathbb{F}_{q^s}} w_{i,\alpha}^{s+1}} + \frac{\varepsilon}{2} N\ .$$

By (9) the above will be satisfied for $\mathbf{c}$ if (where we have used the fact that $\sum_{i=1}^{N} \Delta(\mathbf{y}_i, C_{\text{in}}(c_i)) = \Delta(\mathbf{y}, \mathbf{c}')$):

$$N - 2\frac{\Delta(\mathbf{c}', \mathbf{y})}{n} \geqslant (s+1) \sqrt[s+1]{\frac{2(2t)!}{t!} K^s N} + \varepsilon N/2,$$

or equivalently

$$1 - 2\frac{\Delta(\mathbf{c}', \mathbf{y})}{\mathcal{N}} \geqslant (s+1) \sqrt[s+1]{\frac{2(2t)!}{t!} \left(\frac{K}{N}\right)^s} + \frac{\varepsilon}{2}\ .$$

From the lower bound in (5) and the fact that $N = q$, the above condition is met as long as $\Delta(\mathbf{c}', \mathbf{y}) \leqslant \frac{1}{2}(1-\varepsilon)\mathcal{N}$. This proves the claim that C can be list decoded from a fraction $1/2 - \varepsilon$ of errors.

List size and running time of algorithm. We now bound the worst case list size output by our list decoding algorithm. By Theorem 2, the worst case list size is at most $(2s)^{O(s)} \left(\frac{1}{w_{max}\zeta}\right)^{s+1} \cdot \sum_{i,\alpha} w_{i,\alpha}^{s+1}$, which by (9) and the choice of ζ in (8) is upper bounded by $(s/\varepsilon)^{O(s)}$. The claimed bound on the worst case list size follows from the choice of s.

To complete the proof, we bound the running time of the list decoding algorithm, which has two parts. The first part is the time needed to compute the weights as defined by (7). Computing the weights in a brute-force manner takes time $O(nNq^s)$, which by the choices of n, N, q, $\mathcal{N}$ and s is upper bounded by $\mathcal{N}^{O(1/\gamma)}$. The second component of the running time, by Theorem 2 and the choice of q and $\mathcal{N}$ is upper bounded by $(s/\varepsilon)^{O(s)}\mathcal{N}^{O(1)}$. The total running time is thus bounded by

$\left(\frac{\mathcal{N}}{\gamma\varepsilon}\right)^{O(1/\gamma)}$. The bound on $\mathcal{N}$ implies a running time of $O\left(\frac{\mathcal{K}}{\gamma\varepsilon}\right)^{O(1/\gamma)}$, as desired. This completes the proof of Theorem 13. ■

4.2 Construction with Linear Dependence on Dimension

Using an alternate concatenated code construction, with a different inner code but again the PV code as outer code, we prove the following result in this section.

Theorem 14. *Given an integer $\mathcal{K} > 1$ and reals $0 < \gamma, \varepsilon < 1/2$, the following holds. There exists a binary linear code of dimension at least $\mathcal{K}$ and block length $\mathcal{N} \leqslant O\left(\frac{\mathcal{K}\log(1/\varepsilon)}{\gamma^{1+\gamma}\varepsilon^{5+\gamma}}\right)$ with the following properties:*

(i) There is an algorithm that can list decode the code from a fraction $1/2 - \varepsilon$ of errors in time at most $(\mathcal{K}/(\gamma\varepsilon))^{O(1/\gamma)}$ and outputs a list of size at most $O\Big((1/\gamma)^{O(1/\gamma)} \cdot (1/\varepsilon)^{3+\gamma}\Big)$.

(ii) There is a deterministic construction of the code in time $(\mathcal{K}/(\gamma\varepsilon))^{O(1/\gamma)}$.

(iii) The code is $\frac{3\varepsilon}{4}$-biased, that is, all nonzero codewords have Hamming weights in the range $\big[(\frac{1}{2} - \frac{3\varepsilon}{4})\mathcal{N}, (\frac{1}{2} + \frac{3\varepsilon}{4})\mathcal{N}\big]$.

Proof. Given $\mathcal{K}, \varepsilon$ and γ, pick parameters r, q, K, N, s for the codes guaranteed by Corollary 3 as follows. Let $K = \left\lceil \frac{2\mathcal{K}}{\log \mathcal{K}} \right\rceil$, and $s = \lceil \frac{3}{\gamma} \rceil$. Let $r = 2$ and take q to be the smallest power of 2 greater than $\frac{32K(s+1)^{1+1/s}}{\varepsilon^{1+3/s}}$, and pick $N = q$.

Let C_{out} be the $\mathbb{F}_2$-linear code guaranteed by Corollary 3 with dimension K and block length N for parameter s. For the inner code C_{in}, we will use the code guaranteed by Corollary 6. In particular, we can construct an $[n,k]_2$ code that is $\varepsilon^2/4$-biased (that is, all non zero-codewords have relative Hamming weights in $[\frac{1}{2} - \frac{\varepsilon^2}{4}, \frac{1}{2} + \frac{\varepsilon^2}{4}]$) for $k = s\log_2 q$ and $n = O(k/\varepsilon^4)$. Further, Corollary 6 guarantees that we can construct such a code in time $\text{poly}\left(2^k, k/\varepsilon\right) = \text{poly}\left(q^s, k/\varepsilon\right) = (\mathcal{K}/(\gamma\varepsilon))^{O(1/\gamma)}$ time. This gives a polynomial time construction of our final code C^*, which is the concatenation of C_{out} and C_{in}.

The proof that C^* is ε-biased is similar to the argument in Theorem 13. By the choices of K and N, $K/N \leqslant \varepsilon$, which by Theorem 2 implies that C_{out} has relative distance $1 - \varepsilon$. Thus, all the non-zero codewords have relative Hamming weights in the range $\left[(1-\varepsilon)\left(\frac{1}{2} - \frac{\varepsilon^2}{4}\right), \frac{1}{2} + \frac{\varepsilon^2}{4}\right] \subseteq \left[\frac{1}{2} - \frac{3\varepsilon}{4}, \frac{1}{2} + \frac{3\varepsilon}{4}\right]$, as required.

The dimension of C^* is $Kk = Ks\log q$ which is easily seen to be at least $\mathcal{K}$. The block length $\mathcal{N} = Nn$ of C^* satisfies

$$\begin{aligned}\mathcal{N} \leqslant \frac{64K(s+1)^{1+1/s}}{\varepsilon^{1+3/s}} \cdot O(\log K + \log(1/\varepsilon)) \cdot O(1/\varepsilon^4) &\leqslant O(s^{1+1/s}\mathcal{K}\log(1/\varepsilon)/\varepsilon^{5+3/s}) \\ &\leqslant O(1/\gamma^{1+\gamma} \cdot \mathcal{K}\log(1/\varepsilon)/\varepsilon^{5+\gamma}) \ .\end{aligned}$$

It remains to give a list decoding algorithm for C^* that can correct a fraction $(1/2 - \varepsilon)$ of errors. This follows from a rather standard method, and we give the details for completeness below. Let

$\mathbf{y} \in \mathbb{F}_2^{\mathcal{N}}$ be a received word. If $\mathbf{c} \in C^*$ agrees with $\mathbf{y}$ on at least $(1/2+\varepsilon)\mathcal{N}$ positions, clearly on at least $\varepsilon N/2$ of the inner encoding blocks, $\mathbf{c}$ and $\mathbf{y}$ agree on at least $\left(\frac{1}{2}+\frac{\varepsilon}{2}\right)n$ positions. Since C_{in} has relative distance at least $\left(\frac{1}{2}-\frac{\varepsilon^2}{4}\right)$, by the Johnson bound (Lemma 4), we know that C_{in} is $((1-\varepsilon)/2, 4/\varepsilon^2)$-list-decodable. Therefore, we can decode $\mathbf{y}$ by decoding each inner block to a radius of $(1-\varepsilon)n/2$, returning a set S_i of at most $4/\varepsilon^2$ symbols of $\mathbb{F}_{q^s}$ for each position of C_{out}. We can then run the list recovery algorithm guaranteed by Corollary 3 for C_{out} and output all outer codewords whose i'th symbols belong to S_i for at least $\varepsilon N/2$ positions.

All that remains to be done is to check that the list recovery Condition (3) of Corollary 3 is met with the choice $\alpha = \varepsilon/2$, $\ell = 4/\varepsilon^2$, and the above choice of K, N, s. Of course the specific choice above was made precisely in order to satisfy this. Indeed, we have $K/N \leqslant \varepsilon^{1+3/s}/(32(s+1)^{1+1/s})$, so the required condition is met if

$$\frac{\varepsilon}{2} > \varepsilon^{\frac{s+3}{s+1}}(1/32)^{\frac{s}{s+1}}(4/\varepsilon^2)^{\frac{1}{s+1}} = \frac{\varepsilon}{2^{\frac{5s-2}{s+1}}} \ ,$$

which is true for every $s \geqslant 1$. The bound on the list size follows from Corollary 3 and noting that $s = O(1/\gamma)$, $\ell = O(1/\varepsilon^2)$, $r = 2$ and $N/K = O(s^2/\varepsilon^{1+3/s})$. The decoding complexity consisting of two parts: the time to decode the inner codes by brute-force which takes $q^{O(s)}$ time, and the time to run the list recovering algorithm for the outer code, which takes time $s^{O(s)}(q\ell)^{O(1)}$ by Corollary 3. The combined runtime is thus bounded by $(\mathcal{N}/(\gamma\varepsilon))^{O(1/\gamma)}$, which in turn is bounded by $(\mathcal{K}/(\gamma\varepsilon))^{O(1/\gamma)}$. ■

5 A Negative Result

In this section, we will state the following limit on codes that are list decodable from a very high fraction of errors. For the sake of completeness, we present its (standard) proof below.

Theorem 15. *Let $a > 0$ be real. Let C be a binary code of block length n. If C is a $\left(\frac{1}{2} - a\sqrt{\frac{\log n}{n}}, L\right)$-list decodable code then*

$$|C| \leqslant L \cdot n^{4a^2 \log e + 1}.$$

To prove Theorem 15, we will need the following lemma.

Lemma 16. *Let $\mathbf{x} \in \{0,1\}^n$ be a fixed vector. Then for a vector $\mathbf{y}$ chosen uniformly at random from $\{0,1\}^n$, the following is true for large enough n:*

$$\Pr\left[\Delta(\mathbf{x},\mathbf{y}) \leqslant \frac{n}{2} - a\sqrt{n\log n}\right] \geqslant \frac{1}{n^{4a^2\log e+1}}.$$

The lemma follows (with slightly different constants) from a lower bound on the tail of a Binomial distribution (cf. [21, Chap. 4]). For the sake of completeness, we present a proof in Appendix C. Armed with the lemma above, let us prove Theorem 15.

Proof of Theorem 15: Fix an arbitrary codeword $\mathbf{c} \in C$. Pick a received word $\mathbf{y} \in \{0,1\}^n$ uniformly at random. By Lemma 16,

$$\Pr_{\mathbf{y}}\left[\mathbf{c} \in B\left(\mathbf{y}, \frac{n}{2} - a\sqrt{n\log n}\right)\right] \geqslant \frac{1}{n^{4a^2\log e+1}},$$

where $B(\mathbf{y}, e')$ is the Hamming ball of radius e' centered at $\mathbf{y}$. Since the choice of $\mathbf{c}$ was arbitrary,

$$\mathbb{E}_{\mathbf{y}}\left[\left|C \cap B\left(\mathbf{y}, \frac{n}{2} - a\sqrt{n \log n}\right)\right|\right] \geqslant \frac{|C|}{n^{4a^2 \log e + 1}}.$$

By the assumption on the list-decodability of C, for every $\mathbf{y} \in \{0,1\}^n$,

$$\left|C \cap B\left(\mathbf{y}, \frac{n}{2} - a\sqrt{n \log n}\right)\right| \leqslant L,$$

which implies that

$$\frac{|C|}{n^{4a^2 \log e + 1}} \leqslant L,$$

as desired. □

Note that if a code C has polynomial time list decoding algorithm, then the worst case list size has to be bounded by a polynomial in the block length of the code. Thus, we have the following corollary of Theorem 15:

Corollary 17. *Let C be a binary code of block length n that can be list decoded from a fraction $1/2 - O\left(\sqrt{\frac{\log n}{n}}\right)$ of errors in time polynomial in n. Then C has a dimension of $O(\log n)$.*

6 Open Questions

The most obvious question left open by our work is to design $[n,k]_2$ binary linear codes that are list decodable from a fraction $1/2 - \varepsilon$ of errors in polynomial time with $n = k^{O(1)}/\varepsilon^a$ for some $2 \leqslant a < 3$. One possible route to prove such a result would be list decode the construction of ζ-biased codes based on the concatenation of Reed-Solomon (or appropriate algebraic) codes with the Hadamard code [2] up to a radius of $(1/2 - O(\zeta))$. However, this seems beyond the reach of current techniques (the current best algorithms only list decode up to a radius of $(1/2 - O(\sqrt{\zeta}))$ [15]), and in fact it is not clear if this is possible at all (even combinatorially, in terms of being guaranteed a small list up to this radius).

The known list decoding algorithms for concatenated codes run in two stages. In the first stage the inner codes are decoded while ignoring the structure of the outer code. In the second stage one uses the intermediate information from the first stage to decode the outer code. The bottleneck with the analysis of most of these algorithms is that they treat the different inner decodings independently, and do not exploit the fact that the various symbols being decoded arise from an outer codeword and therefore have some global structure. Choosing soft information during the inner decodings with a global view (see [19] for an example) might be one possible avenue for further progress in list decoding concatenated codes.

Another open question is to obtain a construction that beats the quartic dependence of n on $1/\varepsilon$ over non-binary alphabets. Remark 1 explains why our construction with inner dual BCH code, that achieves near-cubic dependence on $1/\varepsilon$ in the binary case, does not immediately extend to larger alphabets.

Acknowledgments

The second author would like to thank Ken Regan and Alan Selman for helpful discussions. We thank the anonymous reviewers for several useful suggestions that improved the presentation of the paper.

References

[1] N. Alon, J. Bruck, J. Naor, M. Naor, and R. Roth. Construction of asymptotically good low-rate error-correcting codes through pseudo-random graphs. *IEEE Transactions on Information Theory*, 38:509–516, 1992.

[2] N. Alon, O. Goldreich, J. Håstad, and R. Peralta. Simple constructions of almost k-wise independent random variables. *Random Structures and Algorithms*, 3:289–304, 1992.

[3] A. Ben-Aroya and A. Ta-Shma. Constructing small-bias sets from algebraic-geometric codes. In *Proceedings of the 50th Annual IEEE Symposium on Foundations of Computer Science*, pages 191–197, 2009.

[4] E. Berlekamp. Factoring polynomials over large finite fields. *Mathematics of Computation*, 24:713–735, 1970.

[5] R. A. Brualdi and V. Pless. Greedy codes. *J. Comb. Theory, Ser. A*, 64(1):10–30, 1993.

[6] J.-Y. Cai, A. Pavan, and D. Sivakumar. On the hardness of the permanent. *Proceedings of the 16th International Symposium on Theoretical Aspects of Computer Science*, March 1999.

[7] P. Elias. Error-correcting codes for list decoding. *IEEE Transactions on Information Theory*, 37:5–12, 1991.

[8] U. Feige, M. Langberg, and K. Nissim. On the hardness of approximating NP witnesses. *Proceedings of the 3rd International Workshop on Approximation Algorithms for Combinatorial Optimization (APPROX)*, pages 120–131, September 2000.

[9] A. Gál, S. Halevi, R. J. Lipton, and E. Petrank. Computing from partial solutions. *Proceedings of the 14th Annual IEEE Conference on Computation Complexity*, pages 34–45, 1999.

[10] O. Goldreich and L. Levin. A hard-core predicate for all one-way functions. *Proceedings of the 21st Annual ACM Symposium on Theory of Computing*, pages 25–32, May 1989.

[11] V. Guruswami. *List decoding of error-correcting codes.* Number 3282 in Lecture Notes in Computer Science. Springer, 2004.

[12] V. Guruswami, J. Hastad, M. Sudan, and D. Zuckerman. Combinatorial bounds for list decoding. *IEEE Transactions on Information Theory*, 48(5):1021–1035, 2002.

[13] V. Guruswami and A. Rudra. Explicit codes achieving list decoding capacity: Error-correction up to the Singleton bound. *IEEE Transactions on Information Theory*, 54(1):135–150, January 2008. Preliminary version appeared as "Explicit capacity-achieving list-decodable codes" in *Proceedings of STOC 06.*

[14] V. Guruswami and M. Sudan. Improved decoding of Reed-Solomon and algebraic-geometric codes. *IEEE Transactions on Information Theory*, 45:1757–1767, 1999.

[15] V. Guruswami and M. Sudan. List decoding algorithms for certain concatenated codes. In *Proceedings of the 32nd Annual ACM Symposium on Theory of Computing (STOC)*, pages 181–190, 2000.

[16] V. Guruswami and M. Sudan. Decoding concatenated codes using soft information. In *Proceedings of the 17th Annual IEEE Conference on Computational Complexity (CCC)*, pages 148–157, 2002.

[17] T. Kaufman and S. Litsyn. Almost orthogonal linear codes are locally testable. In *Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 317–326, 2005.

[18] M. Kiwi. Testing and weight distributions of dual codes. *ECCC Technical Report TR-97-010*, 1997.

[19] R. Koetter. On optimal weight assignments for multivariate interpolation list-decoding. In *Proc. 2006 IEEE Information Theory Workshop*, pages 37–41, March 2006.

[20] S. R. Kumar and D. Sivakumar. Proofs, codes, and polynomial-time reducibilities. *Proceedings of the 14th Annual IEEE Conference on Computation Complexity*, 1999.

[21] M. Ledoux and M. Talagrand. *Probability in Banach Spaces.* Springer, 1991.

[22] F. J. MacWilliams and N. J. A. Sloane. *The Theory of Error-Correcting Codes.* Elsevier/North-Holland, Amsterdam, 1981.

[23] E. Mossel and C. Umans. On the complexity of approximating the VC dimension. *J. Comput. Syst. Sci.*, 65(4):660–671, 2002. Preliminary version in the 16th Annual Conference on Computational Complexity (CCC), 2001.

[24] J. Naor and M. Naor. Small-bias probability spaces: Efficient constructions and applications. *SIAM Journal on Computing*, 22(4):838–856, 1993.

[25] F. Parvaresh and A. Vardy. Correcting errors beyond the Guruswami-Sudan radius in polynomial time. In *Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science*, pages 285–294, 2005.

[26] E. Porat and A. Rothschild. Explicit non-adaptive combinatorial group testing schemes. In *Proceedings of the 35th International Colloquium on Automata, Languages and Programming (ICALP)*, pages 748–759, 2008.

[27] A. Rudra. *List Decoding and Property Testing of Error Correcting Codes.* PhD thesis, University of Washington, 2007.

[28] R. Shaltiel and C. Umans. Simple extractors for all min-entropies and a new pseudo-random generator. *Journal of the ACM*, 52(2):172–216, 2005.

[29] D. Sheldon and N. E. Young. Hamming approximation of NP witnesses, November 2003. Manuscript.

[30] D. Sivakumar. On membership comparable sets. *J. Comput. Syst. Sci.*, 59(2):270–280, October 1999.

[31] M. Sudan. List decoding: Algorithms and applications. *SIGACT News*, 31:16–27, 2000.

[32] M. Sudan, L. Trevisan, and S. Vadhan. Pseudorandom generators without the XOR lemma. *J. Comput. Sys. Sci.*, 62(2):236–266, 2001.

[33] V. V. Zyablov and M. S. Pinsker. List cascade decoding. *Problems of Information Transmission*, 17(4):29–34, 1981 (in Russian); pp. 236-240 (in English), 1982.

A Proof of Theorem 1

We first begin by instantiating the parameter h in the definition of the PV code with parameter $s \geqslant 1$. Define h to be the smallest power of r larger than $\sqrt[s+1]{W/(K-1)}+2$. Note that this implies that

$$\sqrt[s+1]{\frac{W}{K-1}} + 2 < h \leqslant r \cdot \left(\sqrt[s+1]{\frac{W}{K-1}} + 2 \right) . \tag{10}$$

The map E corresponds to the PV code with the parameters above (where the evaluation points are $\{\beta_1, \ldots, \beta_N\} \subseteq \mathbb{F}_q$). The discussion above Theorem 1 shows that the map is $\mathbb{F}_r$-linear. The explicitness follows from the fact that the PV code is explicit. The distance of the PV code is at least the distance of the RS code with dimension K and block length N, which is $N-K+1$. Thus, to complete the proof, we need to prove the claim about the soft decodability of the PV code.

In the remainder of the proof, we will assume that the reader is familiar with the notions of weighted degree and multiplicities. (For more details see [25], [13], or [27, Chap. 3].)

For every $1 \leqslant i \leqslant N$, let S_i denote the set of elements $\alpha \in \mathbb{F}_{q^s}$ such that $w_{i,\alpha} > 0$. Let

$$D = \left\lceil \sqrt[s+1]{(K-1)^s \cdot W} \right\rceil + 1 . \tag{11}$$

Then given as input the weights $\{w_{i,\alpha}\}$ ($1 \leqslant i \leqslant N, \alpha \in \mathbb{F}_{q^s}$), the soft decoding algorithm is as follows:

1. Interpolate a non-zero multivariate polynomial $Q(X, Y_1, \ldots, Y_s)$ with $(1, K-1, \ldots, K-1)$-weighted degree at most D such that the polynomial has multiplicity $w_{i,\alpha}$ at point (β_i, α) for every $\alpha \in S_i$.

2. Output all polynomials $f(X)$ of degree at most $K-1$ such that $Q(X, f(X), f_1(X), \ldots, f_{s-1}(X)) \equiv 0$ and satisfy (1).

First, we argue that the algorithm above will output all codewords $(c_1, \ldots, c_N) = (f(\beta_1), \ldots, f(\beta_N)) \in C$ that satisfy (1). Towards this end, we argue that there always exists a non-zero polynomial Q with the required properties in Step 1. It is well-known that any polynomial $Q(X, Y_1, \ldots, Y_s)$ with

$(1, K-1, \ldots, K-1)$-weighted degree D has at least $\frac{D^{s+1}}{(s+1)!(K-1)^s}$ many monomials. Further, a constraint that $Q(X, Y_1, \ldots, Y_s)$ has multiplicity m at $(\beta, \alpha) \in \mathbb{F}_q \times \mathbb{F}_{q^s}$ implies $\binom{m+s}{s+1}$ many constraints in the coefficient of the monomials of Q (which we think of as variables). The proof of both these facts can be found in [27, Sec. 3.5]. Since we are guaranteed a non-zero solution if the number of variables exceed the number of constraints, we would be done if

$$\frac{D^{s+1}}{(s+1)!(K-1)^s} > \sum_{i=1}^{N} \sum_{\alpha \in \mathbb{F}_{q^s}} \binom{w_{i,\alpha} + s}{s+1},$$

which is satisfied by our choice of D.

Next, we argue that Step 2 of the algorithm is correct. Towards this we need to show that for any $f(X)$ of degree at most $K-1$ such that

$$\sum_{i=1}^{N-1} w_{i,(f(\beta_i), f_1(\beta_i), \ldots, f_{s-1}(\beta_i))} > \lceil \sqrt[s+1]{(K-1)^s W} \rceil + 1 \ , \tag{12}$$

we have $Q(X, f(X), f_1(X), \ldots, f_{s-1}(X)) \equiv 0$. Consider the univariate polynomial

$$R(X) \stackrel{\text{def}}{=} Q(X, f(X), f_1(X), \ldots, f_{s-1}(X)) \ .$$

Note that for every $1 \leqslant i \leqslant N$, $R(X)$ has $w_{i,(f(\beta_i), f_1(\beta_i), \ldots, f_{s-1}(\beta_i))}$ roots at β_i. Thus, $R(X)$ has at least $\theta \stackrel{\text{def}}{=} \sum_{i=1}^{N-1} w_{i,(f(\beta_i), f_1(\beta_i), \ldots, f_{s-1}(\beta_i))}$ roots. Further, note that $R(X)$ has degree at most D. Thus, $R(X) \equiv 0$ if we have $\theta > D$, and this condition is met by (11) and (12).

Finally, we bound the running time of the algorithm above (and at the same time show that the algorithm outputs at most L codewords). Step 1 entails solving a system of linear equations in at most $D^{s+1}/(K-1)^s$ variables[3] and $W/(s+1)!$ constraints. This can be solved in time poly(W) (e.g. by Gaussian elimination). Since $W \leqslant ((K-1) \cdot L)^{1+1/s}$ and $K \leqslant q$, we conclude that this step takes time poly(q, L).

We now turn to Step 2 of the algorithm. By first dividing out $Q(X, Y_1, \ldots, Y_s)$ enough times by $E(X)$, we can assume that $Q(X, Y_1, \ldots, Y_s)$ is not divisible by $E(X)$. Then we consider the polynomial $T(Y_1, \ldots, Y_s) \stackrel{\text{def}}{=} Q(X, Y_1, \ldots, Y_s) \mod E(X)$ as a polynomial over $\mathbb{F}_{q^K} = \mathbb{F}_q[X]/(E(X))$. Since we have $f_j = f(X)^{h^j} \mod (E(X))$, this implies that we would be done if we find all the roots of $R(Y_1) \stackrel{\text{def}}{=} T(Y_1, Y_1^h, \ldots, Y_1^{h^{s-1}})$. We have to guard against the possibility that $R(Y_1)$ might be the zero polynomial. But it is easy to see that this cannot happen if the degree of T in each Y_i is less than h (see, for example [25, Lemma 15].) Since the degree of T in each Y_i is at most $D/(K-1)$, this holds for the choice of h, D in (10) and (11).

Since $R(Y_1)$ has degree at most h^s, the number of codewords output by the polynomial is upper bounded by h^s, which in turn is upper bounded by L by our choice of h. Finally, all roots (over $\mathbb{F}_{q^K}$) of $R(Y_1)$ can be found by the deterministic factoring algorithm from [4], which will run in time poly(q, L), as desired. This completes the proof.

[3]It is easy to show that the number of monomials of $(s+1)$-variate polynomial with $(1, K-1, \ldots, K-1)$-weighted degree at most D is at most $D^{s+1}/(K-1)^s$.

B Proof of Lemma 8

The proof uses the MacWilliams transform and Krawtchouk polynomials, which we define next.

Definition 3 (Krawtchouk Polynomial)**.** *The Krawtchouk polynomial of degree j is defined as follows:*

$$P_j^n(X) = \sum_{i=0}^{j} (-1)^i \binom{X}{j}\binom{N-X}{j-i}.$$

The MacWilliams transform relates the weight distribution of of a binary linear code to the weight distribution of its dual.

Theorem 18 (MacWilliams Transform [22])**.** *For a binary linear code C of block length n,*

$$A_j(C^{\perp}) = \frac{1}{|C|} \cdot \sum_{i=0}^{n} A_i(C) P_j^n(i).$$

We will also need the fact that Krawtchouk polynomials are orthogonal:

Lemma 19 ([18])**.** *Let i, j, r, s and n be non-negative integers. Then the following holds:*

$$\sum_{i=0}^{n} \binom{n}{i} P_r^n(i) P_s^n(i) = 2^n \binom{n}{r} \delta_{r,s},$$

where $\delta_{r,s} = 1$ if $r = s$ and is 0 otherwise.

We will now present the proof of Lemma 8.

As the Krawtchouk polynomials form an orthogonal basis and $f(X)$ is a polynomial of degree at most $2t$,

$$f(X) = \sum_{j=0}^{2t} \alpha_j P_j^n(X), \tag{13}$$

for some suitable coefficients α_k. In particular, Lemma 19 implies that

$$\alpha_0 = \frac{1}{2^n} \sum_{i=0}^{n} \binom{n}{i} f(i). \tag{14}$$

Thus, we have

$$\begin{aligned}
\sum_{i=0}^{n} A_i(C) f(i) &= \sum_{j=0}^{2t} \alpha_k \sum_{i=0}^{n} A_i(C) P_j^n(i) && (15)\\
&= |C| \sum_{j=0}^{2t} \alpha_k A_i(C^{\perp}) && (16)\\
&= |C| \cdot \alpha_0 && (17)\\
&= \frac{|C|}{2^n} \cdot \sum_{i=0}^{n} \binom{n}{i} f(i). && (18)
\end{aligned}$$

In the above (15) follows from (13). (16) follows from Theorem 18. (17) follows from the assumption that $C^{\perp}$ has distance of at least $2t+1$. Finally, (18) follows from (14). The proof is complete.

C Proof of Lemma 16

Note that we need to prove that

$$\Pr_{\mathbf{y}}\left[\mathbf{y} \in B\left(\mathbf{c}, \frac{n}{2} - a\sqrt{n \log n}\right)\right] \geqslant \frac{1}{n^{4a^2 \log e + 1}}.$$

Since $\mathbf{y}$ is chosen uniformly at random,

$$\Pr_{\mathbf{y}}\left[\mathbf{y} \in B\left(\mathbf{c}, \frac{n}{2} - a\sqrt{n \log n}\right)\right] = \frac{\left|B\left(\mathbf{c}, \frac{n}{2} - a\sqrt{n \log n}\right)\right|}{2^n}.$$

To complete the proof, we will show that for large enough n:

$$\log\left(\left|B\left(\mathbf{c}, \frac{n}{2} - a\sqrt{n \log n}\right)\right|\right) \geqslant n - (4a^2 \log e + 1) \log n. \tag{19}$$

It is easy to check that

$$\left|B\left(\mathbf{c}, \frac{n}{2} - a\sqrt{n \log n}\right)\right| \geqslant \binom{n}{\frac{n}{2} - a\sqrt{n \log n}} = \frac{n!}{(n/2 - a\sqrt{n \log n})!(n/2 + a\sqrt{n \log n})!}.$$

Before we proceed, we recall Stirling's approximation for $b!$:

$$\sqrt{2\pi b}\left(\frac{b}{e}\right)^b e^{\lambda_1(b)} < b! < \sqrt{2\pi b}\left(\frac{b}{e}\right)^b e^{\lambda_2(b)}.$$

where

$$\lambda_1(b) = \frac{1}{12b+1} \text{ and } \lambda_2(b) = \frac{1}{12b}.$$

Thus,

$$\left|B\left(\mathbf{c}, \frac{n}{2} - a\sqrt{n \log n}\right)\right| \geqslant E(n) \cdot \frac{n^n}{\left(\frac{n}{2} - a\sqrt{n \log n}\right)^{\frac{n}{2} - a\sqrt{n \log n}} \left(\frac{n}{2} + a\sqrt{n \log n}\right)^{\frac{n}{2} + a\sqrt{n \log n}}},$$

where

$$E(n) = \frac{1}{\sqrt{2\pi\left(\frac{n}{4} - a^2 \log n\right)}} \cdot \frac{e^{\lambda_1(n)}}{e^{\lambda_2\left(\frac{n}{2} - a\sqrt{n \log n}\right) + \lambda_2\left(\frac{n}{2} + a\sqrt{n \log n}\right)}}.$$

For large enough n, $E(n) \geqslant \frac{1}{n}$, which implies that

$$\left|B\left(\mathbf{c}, \frac{n}{2} - a\sqrt{n \log n}\right)\right| \geqslant \frac{n^{n-1}}{\left(\frac{n}{2} - a\sqrt{n \log n}\right)^{\frac{n}{2} - a\sqrt{n \log n}} \left(\frac{n}{2} + a\sqrt{n \log n}\right)^{\frac{n}{2} + a\sqrt{n \log n}}},$$

Thus,

$$\begin{aligned}\log\left(\left|B\left(\mathbf{c}, \frac{n}{2} - a\sqrt{n \log n}\right)\right|\right) \geqslant (n-1)\log n &- \left(\frac{n}{2} - a\sqrt{n \log n}\right)\left(\log\left(\frac{n}{2}\right) + \log\left(1 - 2a\sqrt{\frac{\log n}{n}}\right)\right) \\ &- \left(\frac{n}{2} + a\sqrt{n \log n}\right)\left(\log\left(\frac{n}{2}\right) + \log\left(1 + 2a\sqrt{\frac{\log n}{n}}\right)\right)\end{aligned}$$

$$
\begin{aligned}
&= n - \log n - \left(\frac{n}{2} - a\sqrt{n\log n}\right)\log\left(1 - 2a\sqrt{\frac{\log n}{n}}\right)\\
&\qquad - \left(\frac{n}{2} + a\sqrt{n\log n}\right)\log\left(1 + 2a\sqrt{\frac{\log n}{n}}\right)\\
&\geqslant n - \log n + 2a\log e\sqrt{\frac{\log n}{n}}\cdot\left(\frac{n}{2} - a\sqrt{n\log n}\right)\\
&\qquad - 2a\log e\left(\frac{n}{2} + a\sqrt{n\log n}\right)\sqrt{\frac{\log n}{n}} \qquad (20)\\
&= n - (4a^2\log e + 1)\log n.
\end{aligned}
$$

In the above (20) follows from the fact that for any $0 \leqslant x < 1$, $\ln(1+x) \leqslant x$, $\ln(1-x) \leqslant -x$ and for large enough n, $2a\sqrt{\frac{\log n}{n}} < 1$. The proof is complete.

D Complexity-theoretic motivation: Approximating NP-witnesses

Every language L in the complexity class NP comes equipped with a polynomial-time decidable witness predicate R_L such that $L = \{x | \exists y, |y| = |x|^c \wedge R_L(x, y)\}$. Note that L might have multiple witness predicates. For an NP-complete language L and given a string $x \in L$, unless $\mathsf{P} = \mathsf{NP}$, one cannot find in polynomial-time a witness y such that $R_L(x, y)$ is true. One natural approximation (considered in [9, 20]) is to try and find a string z that does not differ much from a satisfying witness y.

Kumar and Sivakumar [20] showed the following connection of approximating NP witness to list decodable codes.

Theorem 20. *Let $\{C_i\}_i$ be a family of binary codes such that for every integer $i > 1$, C_i is an $[n_i, i]_2$ code that can be list decoded from a fraction $1/2 - \varepsilon$ of errors (for some $\varepsilon = \varepsilon(n_i) > 0$) in $\mathrm{poly}(i, 1/\varepsilon)$ time. Then for any language L in NP, there exists a polynomial-time decidable witness predicate R'_L such that for $x \in L$ the following holds. If one can compute a string $z \in \{0,1\}^{|y|}$ such that $\Delta(z, y) \leqslant |y|/2 - \varepsilon|y|$ in polynomial time, where $R'_L(x, y)$ is true, then one can compute a witness y' that satisfies $R'_L(x, y')$ in polynomial time.*

We now briefly sketch the idea behind the proof of the theorem above. Let R_L be a polynomial-time decidable witness predicate for L. We define R'_L as follows:

$$R'_L(x, y) = \left[(|y| = |x|^d) \wedge \left(\exists z \left[R_L(x, z) \wedge |z| = |x|^c \wedge y = C_{|z|}(z)\right]\right)\right]$$

where $C_{|z|}$ is the code of dimension $|z|$ referred to in Theorem 20.

Consider an $x \in \{0,1\}^n$ such that $x \in L$ and define $N = n^d$. Let $y' \in \{0,1\}^N$ be such that $\Delta(y', C_N(z')) \leqslant N/2 - \varepsilon N$ for some witness $z' \in \{0,1\}^{n^c}$ such that $R_L(x, z')$ is satisfied. By running the list decoding algorithm for C_N, one can obtain a polynomial-sized list $\{z'_1, z'_2, \ldots, z'_\ell\}$ that contains z'. One can then prune the list to find some satisfying witness for R_L by checking if $R_L(x, z'_i)$ is satisfied for some $1 \leqslant i \leqslant \ell$.

Recall that our result implies that there are codes with $n = O(k^3/\varepsilon^{3+\gamma})$ that are list decodable in $\mathrm{poly}(n)$ time up to $1/2 - \varepsilon$ fraction of errors. Note that if we choose $\varepsilon = n^{-1/3+\gamma}$, then $n = \mathrm{poly}(k)$ (assuming γ is a constant). Thus, Theorem 20 immediately implies the following corollary:

Corollary 21. *For every $\gamma > 0$ the following holds. For every language in* NP*, there exists a polynomial time decidable witness predicate R'_L with the following property: There is an algorithm that given $x \in L$ and an arbitrary string of length N that is promised to agree in at least $N/2 + N^{2/3+\gamma}$ positions with some unknown N-bit witness y satisfying $R'_L(x, y)$, runs in* $\mathrm{poly}(|x|, N)$ *time and outputs a witness y' that satisfies $R'_L(x, y')$.*

Thus, for any NP-complete language, it is NP-hard to compute a string that agrees in at least $N/2 + N^{2/3+\gamma}$ (for $\gamma > 0$) positions with some N-bit witness (for some witness predicate). This improves upon the previous "hardness of approximation" result of $N/2+N^{3/4+\gamma}$ from [15]. The best one could hope for via these methods is a bound of $N/2 + O(\sqrt{N \log N})$, as we show in Section 5.

Related Work on NP Witness Approximation. The model of approximating NP-witness of [20] has the slight disadvantage that the polynomial-time decidable witness predicate R'_L for the language $L \in \mathsf{NP}$ is somewhat unnatural. For example, for the *SAT* problem, we would like the witness for R'_L (and R_L) to be a truth assignment. This model was first considered by Gál, Halevi, Lipton and Petrank [9] (though their original notion of approximation was to correctly compute some bits of a satisfying witness). Feige, Langberg and Nissim in [8] show that it is NP-hard to compute an assignment that agrees with $N/2+N^{\eta}$ many bits of a satisfying assignment for a 3CNF formula (for some $\eta > 0$). Sheldon and Young extended this result to work for every $\eta > 0$ [29]. Similar results were shown for other specific NP-complete problems in [8, 29].

We now discuss the result of Sheldon and Young for SAT in some more detail. As was mentioned earlier, one cannot hope to go beyond a "hardness of approximation" result of $N/2 + \sqrt{N}$ using Theorem 20. By contrast, the result for SAT in [29] shows a hardness of approximation result for $N/2 + N^{\eta}$ for *any* $\eta > 0$ (and hence, "beats" the list decoding approach). Sheldon-Young *do not* use list decodable codes in their reduction. In particular, their reductions works by repeating an input variable enough number of times (and suitably changing the SAT formula) so that assigning the "wrong" value to the chosen bit will result in an assignment that agrees with any satisfying assignment in strictly less than $N/2 + N^{\eta}$ many positions. Thus, if there exists an algorithm that can output an assignment that agrees in at least $N/2 + N^{\eta}$ positions with some satisfying assignment, then the assignment computed by the algorithm will assign the "correct" value to the chosen bit. Repeating this procedure recovers a satisfying assignment (assuming one exists).

We note that the reduction outline above is specific to SAT (though the idea can be generalized to some other "natural" NP-complete problems). By contrast, the reduction of Theorem 20 works for *arbitrary* NP language L. (Further, the witness predicate R'_L of Theorem 20 can be constructed from *any* witness predicate R_L for L.)