Proceedings of the International Congress of Mathematicians
Hyderabad, India, 2010

Bridging Shannon and Hamming: List Error-Correction with Optimal Rate

Venkatesan Guruswami *

Abstract. Error-correcting codes tackle the fundamental problem of recovering from errors during data communication and storage. A basic issue in coding theory concerns the modeling of the channel noise. Shannon's theory models the channel as a stochastic process with a known probability law. Hamming suggested a combinatorial approach where the channel causes worst-case errors subject only to a limit on the number of errors. These two approaches share a lot of common tools, however in terms of quantitative results, the classical results for worst-case errors were much weaker.

We survey recent progress on list decoding, highlighting its power and generality as an avenue to construct codes resilient to worst-case errors with information rates similar to what is possible against probabilistic errors. In particular, we discuss recent explicit constructions of list-decodable codes with information-theoretically optimal redundancy that is arbitrarily close to the fraction of symbols that can be corrupted by worst-case errors.

Mathematics Subject Classification (2000). Primary 11T71; Secondary 94B35.

Keywords. Error-correction algorithms; Explicit constructions; Reed-Solomon codes; Algebraic-geometric codes; Shannon capacity; List decoding; Polynomial reconstruction.

1. Introduction

Error-correcting codes enable reliable storage and transmission of data by providing a way to detect and rectify the errors caused by intervening noise. Mathematically, a code can be specified by an *encoding* function $E : \mathcal{M} \to \Sigma^n$ that maps a *message* $m \in \mathcal{M}$ into a redundant string $E(m)$ (called *codeword*) over some alphabet Σ. The set of all codewords is a subset $C \subset \Sigma^n$ called the *code* (or sometimes codebook). Only a small fraction of all possible strings in Σ^n are valid codewords, and the redundancy built into codewords is judiciously chosen in order to enable *decoding* the message m even from a somewhat distorted version of the codeword $E(m)$.

At a high level, the goal is correct many errors without introducing too much redundancy — these conflicting goals impose a fundamental trade-off, and the basic goals of coding theory are to understand this trade-off, and construct explicit codes

*Supported in part by a David and Lucile Packard Fellowship and NSF CCF-0953155.

and efficient decoding algorithms for operating close to this trade-off. A convenient normalized measure of the redundancy of a code is given by its *rate*, which is defined as $\frac{\log |\mathcal{M}|}{n \log |\Sigma|}$ (unless specified logarithms are to the base 2). The rate of a code measures the proportion of non-redundant information conveyed per bit of the codeword. The larger the rate, the less redundant the encoding. Often the message set $\mathcal{M}$ is identified with strings over the code's alphabet Σ (say with Σ^k) in which the case the rate is simple k/n. An important special case is when Σ is a finite field $\mathbb{F}$ and the encoding map is a linear transformation $\mathbb{F}^k \to \mathbb{F}^n$. Such codes are called *linear codes*.

1.1. Modeling errors: Shannon vs. Hamming. An important issue in coding theory is the modeling of errors caused by the intervening noisy "channel." There are two broad approaches to this. Shannon's approach, in his pioneering work [44] that led to the birth of information theory, was to model the channel as a stochastic process with a precisely defined probability law. A simple example over alphabet $\Sigma = \{0, 1\}$ is the binary symmetric channel where the channel flips each transmitted bit with probability ρ, *independently* of other bits. Shannon precisely characterized the largest rate (called "capacity") at which reliable communication is possible on such channels. In one of the early uses of the probabilistic method, he showed the existence of codes with rates approaching capacity that (together with maximum likelihood decoding) achieved exponentially small probability of miscommunication. Shannon's work did not give a method to explicitly construct good codes or design efficient error-correction algorithms. Further there was no crisply abstracted (or at least easy to reason about) criterion for when a code was good in this model.

A different approach, implicit in the roughly concurrent work of Hamming [32], is to model the channel by a *worst-case* or *adversarial* process that can corrupt the codeword arbitrarily, subject only to a limit on the total number of errors caused. Both the locations of the corrupted symbols and the actual errors are assumed to be worst-case. The Hamming approach is more combinatorial, and notions such as the minimum distance between codewords, and connections to sphere packing (of Hamming balls), emerge as criteria for construction of good codes. Techniques from many areas of mathematics including algebra, graph theory, combinatorics, number theory, and the theory of algebraic function fields and curves, have been very successfully brought to bear on the code construction problem.

To contrast the Shannon and Hamming models, in the former the channel behavior is oblivious to the overall message or codeword being transmitted, with the channel action on the i'th symbol only dependent on that symbol, and perhaps a fixed amount of state information held by the channel. In the Hamming model, the channel behavior can depend arbitrarily on the codeword, causing a maliciously chosen error pattern. While this might be a pessimistic viewpoint, even if the channel is not an adversary, assuming independent errors governed by a precise channel law might be too strong. Codes for the Hamming model obviate the need for a very precise noise model, and are robust against a wide range of channel behaviors.

Unfortunately, requiring that the code be able to correct *every* pattern of up to certain fraction, say ρ, of errors (instead of *typical* patterns) poses stringent limits. In particular, in order to ensure that a codeword $c \in \Sigma^n$ will not be confused for a codeword $c' \neq c$ even after up to ρn errors distort c, c must have Hamming distance more than $2\rho n$ from every other codeword. Thus every pair of distinct codewords must differ in at least a fraction 2ρ of positions. This distance requirement limits the number of codewords one can pack, and thus the rate of the code. For example, for codes over $\Sigma = \{0, 1\}$, when the fraction ρ of worst-case errors exceeds $1/4$, the rate of communication must approach zero, whereas even close to a fraction $1/2$ of random errors can be corrected with positive rate of communication in Shannon's model. Thus the generality of the worst-case noise model apparently comes at the price of a significant loss in rate.

1.2. List decoding. A simple relaxation of the requirement on the decoder, however, enables bridging this gap between worst-case and random errors. In this model, called *list decoding*, the decoder, given as input a noisy received word $y \in \Sigma^n$, must output a list of all codewords that are within Hamming distance ρn from y (where ρ is a bound on the fraction of worst-case errors). Thus, the decoder outputs all possible codewords that could have been corrupted into y if at most ρn errors occurred. Of course for this to be useful the codewords must be "sparsely distributed" so that no ball of radius ρn contains too many codewords. Surprisingly, allowing a small list suffices to approach the Shannon limit while correcting *all* error patterns. In particular, when transmitting bits, there *exist* codes list-decodable up to a ρ fraction of errors with rate approaching $1 - h(\rho)$ (where $h(x) = -x \log x - (1-x)\log(1-x)$ is the binary entropy function), which is the capacity of the binary symmetric channel that flips bits randomly and independently with probability ρ. In particular, the rate is positive even for $\rho \to 1/2$.

Let us address the obvious question that might immediately arise in the reader's mind: Why is list decoding a meaningful model for recovery from worst-case errors? There are several compelling reasons, of which we mention a few. It is true that there are worst-case patterns of ρ fraction of errors that can lead to received words that cannot be unambiguously decoded as soon as ρ equals half the minimum fractional distance between codewords. However, for many codes, including random codes and all codes over large alphabets (such as Reed-Solomon codes) [36, 42], it can be shown that for most patterns of almost *twice* as many errors, there will be a unique close-by codeword. Thus, the list decoder outputting more than one codeword is a rare event. Therefore, it effectively decodes unambiguously, but we have obviated the need for a precise probabilistic channel model, and the results are robust to a wide variety of noise processes, including complicated combinations of i.i.d and bursty errors!

Further, with list decoding, instead of citing a pathological error pattern as an excuse to not decode other correctable patterns of errors, the decoder *always* corrects more errors, outputting a small list in the worst-case. Even for the worst-case errors which cannot be unambiguously decoded, returning a small list of possibilities is certainly no worse than declaring a decoding failure, and possibly

quite useful if there is some context related information that can be used to identify the correct codeword from the list. In this vein, we should mention that in the last decade or two, list decoding has found many surprising applications beyond the coding theory, in algorithms, computational complexity theory, and cryptography. In these applications, there is either an application-specific method for breaking ties amongst the codewords in the list, or having a small list is not an issue at all. Finally, the primitive of list decoding seems rather versatile in implying solutions in some other models for bridging between worst-case and random errors also; we briefly touch upon this aspect at the end of the survey in Section 7.

The notion of list decoding dates back to work in the late 1950s by Elias [8] and Wozencraft [54]. Existential results indicating the potential of list decoding to correct many more errors have also been around for a while [55]. However, list decoding was revived with an algorithmic focus only in works motivated by complexity theory, beginning with the works of Goldreich and Levin [14], Ar *et al.* [2], and Sudan [48]. Ultimately, the usefulness of list decoding comes from the fact that, after a long hiatus, efficient list decoding algorithms for some important codes have been discovered in the last 10-15 years. These have turned some of the existential results on list decoding into constructive ones, and even led to the explicit construction of efficiently list-decodable codes with rates approaching the optimal information-theoretic limit.

In this survey, we discuss some of the background on error-correction, including some of the existential results on the potential of list decoding, and then focus on recent progress on list decoding algorithms for algebraic codes. We also mention some of the key open questions in the subject.

Related surveys: Surveys covering related material include a longer survey on algorithmic results for list decoding [19] and a "Research Highlight" written for a broad computer science audience [26]. For reasons of space, we do not discuss the several compelling applications of list decoding beyond coding theory in this survey. Some of these are surveyed in [49, 50] or [17, Chap. 12], but these do not include some recent applications such as [43, 31]. Various "pseudorandom" objects can be described and studied in a unified way via appropriate variants of list decoding, and these are surveyed in [52].

1.3. Organization.

We warm-up in Section 2 with a discussion of a particularly simple noise model, namely erasures. Over large alphabets, Reed-Solomon codes, which play an important role in list decoding, give a simple and optimal solution to the erasure recovery problem. Over the binary alphabet, we will mention how list decoding enables communication even when the fraction of erased bits approaches the obvious information-theoretic limit, though such codes are not known explicitly.

We then turn in Section 3 to the more challenging problem of decoding from errors. We state the existence theorems (established by the probabilistic method) which show that via list decoding it is possible to achieve rate similar to the Shannon model.

We next turn to constructive results, which aim to realize the potential of list decoding with explicit codes and efficient algorithms. In Section 4, we discuss list decoding of Reed-Solomon (RS) codes, including the "method of multiplicities" which leads to a "soft-decision" decoding algorithm of practical importance. Next, in Section 5, we discuss recently discovered variants of Reed-Solomon codes, called *folded* RS codes, and how they can be list decoded up to the information-theoretic limit, achieving the optimal trade-off between rate and fraction of errors decoded. Using a powerful "list recovery" property of these codes, one can reduce the alphabet size to a constant, and also construct good binary list-decodable codes, which we will briefly touch upon in Section 6. Finally, in Section 7, we will mention some alternate approaches to list decoding for bridging between the worst-case and random error models, and indicate the versatility of list decoding in devising solutions in these models as well.

2. Decoding from erasures

We begin the technical discussion by considering the rather benign *erasure channel*. For $\alpha \in (0,1)$ and an alphabet Σ, the erasure channel $\mathsf{Erase}_\alpha(\Sigma)$ erases an *arbitrary* subset of up to a fraction α of the symbols transmitted, leaving the rest unaltered. In other words, it distorts a codeword $c \in \Sigma^n$ to $y \in (\Sigma \cup \{?\})^n$ with $y_i \in \{c_i, ?\}$ for $1 \leqslant i \leqslant n$ and $y_i = \{?\}$ for at most αn locations i (these correspond to the erased symbols). Note the locations of the erasures are known, so the decoding problem is just to "interpolate" the erased symbols.[1] Erasures are relevant to modeling packet loss on the internet, where the identity of the missing packets can be determined using header information.

2.1. Optimal erasure recovery over large alphabets.

Over large alphabets Σ, an important and well-studied family of codes called Reed-Solomon (RS) codes, give an optimal solution to the erasure recovery problem.

Definition 1 (Reed-Solomon codes). For a finite field $\mathbb{F}$ with $|\mathbb{F}| \geqslant n$, an n-tuple $S = (a_1, a_2, \ldots, a_n)$ of n distinct elements of $\mathbb{F}$, and an integer $1 \leqslant k \leqslant n$, the Reed-Solomon code $\mathrm{RS}_{\mathbb{F},S}[n,k]$ is the k-dimensional subspace of $\mathbb{F}^n$ defined as

$$\mathrm{RS}_{\mathbb{F},S}[n,k] = \{(p(a_1), p(a_2), \ldots, p(a_n)) \mid p \in \mathbb{F}[X] \text{ is a polynomial of degree } < k\}\,.$$

In other words, the message $(m_0, \ldots, m_{k-1}) \in \mathbb{F}^k$ is viewed as a polynomial $m_0 + m_1 X + \cdots + m_{k-1}X^{k-1} \in \mathbb{F}[X]$ and it is encoded by its evaluation at $a_1, a_2, \ldots, a_n$.

Since two distinct polynomials of degree $k-1$ over $\mathbb{F}$ can have equal evaluations on at most $k-1$ points, any two distinct codewords of the above RS code differ in at least $n-k+1$ locations. Equivalently, given the evaluations of an unknown message polynomial p at any subset of k locations a_i, the polynomial p is uniquely

[1] In a harsher noise model called the *deletion channel* some symbols are deleted and the location of the deleted symbols is not known; we will not discuss this channel here.

determined, and in fact can be efficiently recovered by polynomial interpolation. Thus, the RS code enables recovery from up to $n-k$ erasures, or a fraction $1-R$ of erasures where $R = k/n$ is the rate of the code.

This trade-off between number of erasures and rate of the code is optimal, since it is information-theoretically impossible to recover the k symbols of the message from a string with less than k non-erased symbols. Thus Reed-Solomon codes give a simple and optimal solution for recovering from erasures. However, the alphabet size of these codes is at least as large as the block length n.

2.2. Algebraic-geometric codes. A generalization of RS codes called algebraic-geometric (AG) codes [16] can approach the above optimal trade-off over a large but fixed alphabet size that is independent of the block length. An AG code over a finite field $\mathbb{F}$ is based on algebraic function field K over $\mathbb{F}$ (a function field is a finite field extension of $\mathbb{F}(X)$). The message space of such an AG code is a linear space $\mathcal{L} \subset K$ of functions with a bounded number, say $< \ell$, of poles, all confined to a single point P_∞ of the algebraic curve corresponding to K. A function $f \in \mathcal{L}$ is encoded by its evaluation at a set S of n $\mathbb{F}$-rational points of K different from P_∞. Since a function with $< \ell$ poles has $< \ell$ zeroes, a function $f \in \mathcal{L}$ is uniquely determined by its evaluations at ℓ points. The AG code thus enables recovery from up to $n-\ell$ erasures. (Given the evaluations of a basis for $\mathcal{L}$ at points in S, interpolating a function $f \in \mathcal{L}$ from its evaluation at ℓ points can be efficiently done by solving a linear system.) An excellent treatment of AG codes can be found in Stichtenoth's book [47].

By the Riemann-Roch theorem, the dimension of $\mathcal{L}$ is at least $\ell - g$ where $g = g(K)$ is the genus of K, and so the code has rate at least $(\ell - g)/n$. When $|\mathbb{F}| = q = r^2$ is a square, there are known constructions of function fields K which have many rational points $N(K)$ in comparison to their genus $g(K)$, attaining the so-called *Drinfeld-Vladut* bound, with $N(K)/g(K) \to \sqrt{q} - 1$ [51, 12]. Using these function fields, one can prove the following (see [45] for an efficient construction algorithm):

Theorem 1. *For $0 < R < 1$ and $\varepsilon > 0$, one can explicitly construct a family of linear codes of rate R over an alphabet of size $O(1/\varepsilon^2)$ that enable polynomial time recovery from a fraction $1 - R - \varepsilon$ erasures.*

A lower bound of $\Omega(1/\varepsilon)$ is known on the alphabet size necessary for such codes. The following is a fascinating and longstanding open question.

Open Problem 1. *Close (or reduce) the gap between the $\Omega(1/\varepsilon)$ lower bound and $O(1/\varepsilon^2)$ upper bound on the size of alphabet needed for codes of rate R that enable recovery from a fraction $(1 - R - \varepsilon)$ of erasures.*

There are also combinatorial approaches to proving the above theorem, based on certain expander graphs [1, 23], which additionally achieve *linear* complexity encoding/decoding algorithms, but these require an alphabet size of $\exp((1/\varepsilon)^{O(1)})$. Interestingly, for random linear codes over $\mathbb{F}_q$ to have the erasure recovery property

stated in Theorem 1, one needs $q \geqslant \exp(\Omega(1/\varepsilon))$. AG codes thus beat the bounds achieved by the probabilistic method, a pretty rare phenomenon in combinatorial constructions!

2.3. Binary codes and list decoding from erasures. We now turn to erasure recovery with codes over a fixed small alphabet, and specifically binary codes (with alphabet $\{0,1\}$) for definiteness. This will be the first foray into the theme of this survey on the distinction between worst-case and random noise.

A simple argument shows that a non-trivial binary code (with rate bounded away from 0 for large block lengths n) must have two codewords that differ in at most $n/2$ codeword positions. This implies that there are patterns of $n/2$ erasures from which unambiguous recovery of the erased symbols is not possible. In other words, for erasure fractions $\alpha \geqslant 1/2$, the rate of communication must approach 0. Nevertheless, for a *random* noise model BEC_α (the binary erasure channel) where each bit is erased independently with probability $\alpha \in (0,1)$, it is well known [7] that there are codes of rate R approaching "capacity" $1-\alpha$ that enable erasure recovery with high probability (over the random noise caused by BEC_α). [2] In fact, a random linear code of such rate has this property with high probability. Explicit codes achieving capacity of BEC_α with polynomial time encoding/decoding algorithms are also known based on Forney's idea of code concatenation [11] (see [18, Sec. 3] for a description) and ultra-efficient algorithms are known based on low-density parity check codes [40, Chap. 3].

A binary linear code C is given by a linear transformation that maps a column vector $x \in \mathbb{F}_2^k$ to $Mx \in \mathbb{F}_2^n$ for some matrix $M \in \mathbb{F}_2^{n\times k}$ whose columns span C. For $T \subseteq \{1,2,\ldots,n\}$, the message x can be uniquely recovered from $(Mx)_{|T}$ (i.e., the bits of Mx corresponding to locations in T) iff $M_{|T}$, the submatrix of M with rows indexed by T, has rank k. Thus the capacity theorem equivalently states that for each $R \in (0,1)$ and $\varepsilon > 0$, there is a matrix $M \in \mathbb{F}_2^{n\times Rn}$ such that a $1 - o_n(1)$ fraction of its $(R+\varepsilon)n \times Rn$ submatrices have rank Rn. (Moreover, this is true for a random matrix w.h.p.)

It turns out that it is possible to ensure that the small fraction of exceptional submatrices also have rank *close* to k. The proof is by the probabilistic method.

Theorem 2. *For each $R \in (0,1)$ and $\varepsilon > 0$ and all large enough n, there is a matrix $M \in \mathbb{F}_2^{n\times Rn}$ such that every $(R+\varepsilon)n \times Rn$ submatrix of M has rank at least $Rn - C/\varepsilon$ for some absolute constant $C < \infty$. Moreover, this is true for a random matrix with probability $1 - e^{-\Omega_{\varepsilon,R}(n)}$.*

The linear code generated by the columns of such a matrix can be *list decoded* from *every* pattern of $(1-R-\varepsilon)$ erasures, obtaining a list of at most $2^{O(1/\varepsilon)}$ codewords consistent with the non-erased bits. Note that the *list size* is independent of n, and allowing for such a list enables recovering from *worst-case* erasures, without a loss in rate compared to what is possible in the probabilistic BEC_α model.

[2] Again, this rate is optimal, since w.h.p. there will be $\approx \alpha n$ erasures, and one must be able recover all the Rn message bits from the remaining $\approx (1-\alpha)n$ bits.

One can show that a list size of $\Omega(1/\varepsilon)$ (or equivalently rank deficiency of $\log(1/\varepsilon)$) is necessary.

Open Problem 2. *Close this exponential gap between the lower and upper bounds on list size for erasure list-decodable linear codes as a function of the distance ε to the optimal trade-off.*

If one does not insist on linearity, binary codes of rate $(1-\alpha-\varepsilon)$ attaining a list size of $O(1/\varepsilon)$ for recovering from a worst-case fraction α of erasures are known, see [17, Chap. 10] and the references therein. The following is another significant open question.

Open Problem 3. *Construct a matrix with properties guaranteed by Theorem 2* explicitly *(deterministically in time polynomial in n). Finding such an explicit matrix, even with a relaxed rank bound of $Rn - g(1/\varepsilon)$ for an arbitrary function $g(\cdot)$ (or even $Rn - g(1/\varepsilon)\log n$), is also open.*

Such a construction would be very interesting as it would give explicit list-decodable codes for bridging between probabilistic and worst-case erasures.

3. List decoding from errors: existential results

We turn to the problem of recovering not just from missing information (as in the erasures model) but from erroneous information. For concreteness we focus on binary codes in the initial discussion, and we will mention related bounds for larger alphabets towards the end of this section.

3.1. Random errors. Perhaps the simplest model of random bit errors is the *binary symmetric channel* BSC_ρ which is a memoryless probabilistic process that flips each bit independently with probability ρ, where $0 < \rho < 1/2$. Shannon's famous noisy coding theorem identifies the *capacity* of this channel to be $1 - h(\rho)$, where $h : [0,1] \to [0,1]$ is the binary entropy function $h(x) = -x\log_2 x - (1-x)\log_2(1-x)$. Specifically, for any $\varepsilon > 0$, there are codes $C \subseteq \{0,1\}^n$ of rate $1 - h(\rho) - \varepsilon$ for which there is a deterministic decoding function that recovers c from $c + e$ with probability $1 - e^{-\Omega_{\varepsilon,\rho}(n)}$ over the choice of $e \in \{0,1\}^n$ from the binomial distribution with mean ρn. This result can be viewed as a packing of "mostly-disjoint" Hamming spheres of radius $\approx \rho n$, each with volume (number of points) $\approx 2^{h(\rho)n}$, centered around the $2^{(1-h(\rho)-\varepsilon)n}$ codewords of C, such that most of the points in each of these spheres do not belong to any other sphere.

This view also immediately suggests the converse theorem, provable by a volume packing argument, that a rate exceeding $1-h(\rho)$ is not possible for communication on BSC_ρ. For each $c \in C$ and an error vector e added by BSC_ρ, w.h.p. $c+e$ belongs to a region consisting of $\approx 2^{h(\rho)n}$ strings. For reliable recovery of c, the decoding function must map most of the strings in this region to c. This implies one cannot pack more than $\approx 2^{(1-h(\rho))n}$ codewords.

Shannon's theorem is proved via the probabilistic method and only guarantees the existence of codes of rate close to $1-h(\rho)$ for reliable communication on BSC_ρ. Subsequent work on algebraic error-correction and concatenated codes gave a polynomial time construction of such codes together with a decoding algorithm [10], though the complexity bounds are of the form $n^{O(1)}\exp(O(1/\varepsilon))$ for achieving rate $1-h(\rho)-\varepsilon$ and thus scale poorly with the gap to capacity (see also [18, Sec. 3]) . Nevertheless, for each fixed $\varepsilon > 0$, one could say that (in theory) good codes of rate within ε of optimal have been constructed for the BSC_ρ.

3.2. Worst-case errors. The case for worst-case or adversarial errors is more challenging. Let us consider a worst-case channel, denoted ADV_ρ, which corrupts an adversarially chosen subset of up to ρn codeword bits (where n is the block length). Unambiguous recovery of the codeword c from $c+e$ for an arbitrary error vector of e of Hamming weight at most ρn clearly requires that the Hamming ball of radius ρn around c is disjoint from similar Hamming balls centered at every other codeword. As discussed in the introduction, this stringent combinatorial packing requirement places strong upper bounds on the code's rate. In particular, for $\rho > 1/4$, the rate must approach 0, whereas for BSC_ρ, communication at positive rate was possible for every $\rho < 1/2$. Also, for $\rho < 1/4$, the rate for communication on ADV_ρ must be bounded away from the capacity $1-h(\rho)$ of BSC_ρ.

While such a perfectly disjoint packing of nearly $2^{(1-h(\rho))n}$ Hamming balls of radius ρn in $\{0,1\}^n$ does not exist, it turns out that it is possible to pack $2^{(1-h(\rho)-\varepsilon)n}$ such Hamming balls such that no $O(1/\varepsilon)$ of them intersect at a point, for any $\varepsilon > 0$. In fact a random packing has such a property with high probability. This shows that in the model of *list decoding*, there is hope to construct codes of rate approaching $1-h(\rho)$ to correct errors caused by ADV_ρ if the decoder is allowed output a small list of codewords in the worst-case. The formal statement follows.

Definition 2. For $0 < \rho < 1$ and an integer $\ell \geqslant 1$, a code $C \subseteq \Sigma^N$ is said to be (ρ,ℓ)-*list decodable* if for every $y \in \Sigma^n$, there are at most ℓ codewords $c \in C$ such that the Hamming distance between y and c is at most ρn. Equivalently, Hamming balls of radius ρn around the codewords cover each point in Σ^n at most ℓ times.

The following theorem states that even with a modest list size ℓ, one can get close to a rate of $1-h(\rho)$ for (ρ,ℓ)-list decodable binary codes. The bound $1-h(\rho)$ on rate is best possible, since the expected number of codewords of C in a ball of radius ρn around a random $y \in \{0,1\}^n$ is at least $\frac{|C|}{2^n} \cdot 2^{(h(\rho)-o(1))n}$ which grows super-polynomially in n if the rate of C exceeds $1-h(\rho)+o(1)$.

Theorem 3. *For $\rho \in (0,1/2)$ and an integer $\ell \geqslant 1$, for all large enough n there is a (ρ,ℓ)-list decodable binary code of block length n and rate at least $1-h(\rho)-1/\ell$.*

Proof. The proof is an application of the probabilistic method [55, 9], and we sketch the simple argument. Let $R = 1-h(\rho)-1/\ell$. Pick a code $C \subseteq \{0,1\}^n$

by uniformly and independently picking $M = 2^{Rn}$ codewords. Fix a center y and a subset S of $(\ell+1)$ codewords of C. Since these codewords are independent, the probability that all of them land in the ball of radius ρn around y (which has volume at most $2^{h(\rho)n}$) is at most $\left(\frac{2^{h(\rho)n}}{2^n}\right)^{\ell+1}$. A union bound over all 2^n choices of y and at most $M^{\ell+1}$ choices of S shows that if $R = 1 - h(\rho) - 1/\ell$, the code fails to be (ρ, ℓ)-list-decodable with probability at most $2^{-\Omega(n)}$. □

The above argument only guarantees an arbitrary, not necessarily linear, code. The existence of (ρ, ℓ)-list decodable binary *linear* codes with a similar rate is more complicated to show, because codewords in a linear code are only pairwise (and not ℓ-wise) independent. Such linear codes were shown to exist via an application of the semi-random method in [22] (this method only worked for binary codes and did not yield a high probability result). Very recently, it was shown that a random linear code meets the bound of Theorem 3 with high probability [21] (and the proof also worked over all finite fields).

The above theorem implies that it is in principle possible to communicate at rate close to $1-h(\rho)$ even against adversarial errors, provided that in the worst-case we allow the decoder to output a small list which includes the correct codeword. Unfortunately, no explicit codes with the list-decodability property guaranteed by Theorem 3 are known (see Open Problem 5).

We now state the analogous bounds for codes over an alphabet Σ with q elements. For $\rho \in [0, 1 - 1/q]$, the capacity of a probabilistic q-ary symmetric channel that flips $a \in \Sigma$ to each $a' \neq a \in \Sigma$ with probability $\rho/(q-1)$ equals $1 - h_q(\rho)$, where $h_q : [0,1] \to [0,1]$ is the q-ary entropy function $h_q(x) = x \log_q(q-1) - x \log_q x - (1-x)\log_q(1-x) = x \log_q(q-1) + \frac{h(x)}{\log_2 q}$. For worst-case errors that arbitrarily corrupt up to a ρ fraction of symbols (where $0 < \rho < 1 - 1/q$), a similar argument to Theorem 3 guarantees the existence of q-ary (ρ, ℓ)-list decodable codes of rate at least $1 - h_q(\rho) - \frac{1}{\ell}$, showing that using list decoding it is possible to match the rate achievable for the q-ary symmetric channel also for worst-case errors.

3.3. Large alphabets.

The function h_q satisfies $h_q(\rho) \leqslant \rho + \frac{h(\rho)}{\log_2 q}$, so keeping $\rho \in (0,1)$ fixed and letting q increase, one sees that for every $\varepsilon > 0$, $1 - h_q(\rho) \geqslant 1 - \rho - \varepsilon$ when $q \geqslant 2^{1/\varepsilon}$. Thus we have the following existential result for list decoding over large alphabets:

Theorem 4. *For every R, $0 < R < 1$, and ε, $0 < \varepsilon < 1 - R$, every $q \geqslant 2^{2/\varepsilon}$ and all large enough integers n, there is a q-ary code of block length n that is $(1 - R - \varepsilon, \frac{2}{\varepsilon})$-list decodable.*

With a code of rate R, the largest error fraction ρ that one can hope to tolerate if $1 - R$. To see this, consider a benign channel that simply erases (or replaces with some fixed string) the last ρn symbols of the codeword. Clearly the only useful information available to the decoder is the first $(1-\rho)n$ symbols of the codeword from which it must be able to recover the Rn message symbols. Thus we must have $1 - \rho \geqslant R$, or $\rho \leqslant 1 - R$.

The above theorem therefore says that with list decoding one can approach this simple information-theoretic limit, and error correct as long as the error fraction is even slightly below the fraction of redundant symbols built into the codewords. The challenge, of course, is to construct an *explicit* list-decodable code with such guarantees, along with an efficient list error-correction algorithm. Such a construction was recently obtained by the author and Rudra [25] (see also [26]), albeit with a list size and decoding complexity of $n^{\Omega(1/\varepsilon)}$. The codes achieving this optimal trade-off between rate and fraction of errors corrected are a variant of Reed-Solomon codes called "folded Reed-Solomon codes." To describe these codes and the ideas underlying their list decoding, we begin with a discussion of list decoding algorithms for RS codes. These algorithms are important for their own sake due to the ubiquity and practical applications of RS codes.

4. List decoding of Reed-Solomon codes

Consider the Reed-Solomon code $\mathrm{RS}_{\mathbb{F},S}[n,k]$ from Definition 1. As discussed in Section 2.1, one can decode this code (uniquely) from $n-k$ erasures, which is optimal. We now turn to correcting errors in a codeword of this RS code.

4.1. Unique decoding RS codes. We begin with the case when the number of errors is small enough so that unambiguous recovery of the correct message is possible. Specifically, let $p \in \mathbb{F}[X]$ be a degree $k-1$ polynomial, and let $y \in \mathbb{F}^n$ be such that $y_i = p(a_i)$ for all but τ values of $i \in \{1,2,\ldots,n\}$. (Thus, y is a corrupted version of the RS codeword encoding p, with at most τ errors.) Since distinct codewords of $\mathrm{RS}_{\mathbb{F},S}[n,k]$ differ in at least $n-k+1$ positions, when $\tau \leqslant (n-k)/2$, then p is the unique polynomial whose evaluations differ from y in at most τ positions. However, it is not obvious how to recover p *efficiently* given y. If we knew the locations of errors, i.e., the set $E = \{i \mid p(a_i) \neq y_i\}$, then we could recover p by polynomial interpolation using its values at a_i, $i \notin E$. There are too many (exponential in n) possibilities for the set E to simply try them all.

Back in 1960, even before polynomial running time was formalized as the notion underlying efficient algorithms, Peterson [39] described a polynomial time algorithm to solve the above problem! We now describe the idea behind a different algorithm, due to Welch and Berlekamp [53], following the elegant description by Gemmell and Sudan [13]. Since for each $i \notin E$, $y_i = p(a_i)$, the bivariate polynomial

$$P(X,Y) = (Y - p(X)) \prod_{i \in E} (X - a_i)$$

satisfies $P(a_i, y_i) = 0$ for $i = 1,2,\ldots,n$. If we could somehow compute $P(X,Y)$, then one can efficiently find its linear (in Y) factor $Y - p(X)$ and recover p. Note that $P(X,Y) = E_1(X)Y - N_1(X)$ for some polynomials $E_1, N_1 \in \mathbb{F}[X]$ with degrees at most τ and $\tau + k - 1$ respectively. Therefore, in an attempt to find $P(X,Y)$, we interpolate a *nonzero* bivariate polynomial $Q(X,Y) = E_2(X)Y - N_2(X)$ satisfying:

1. $\deg(E_2) \leqslant \tau$ and $\deg(N_2) \leqslant \tau + k - 1$.

2. $Q(a_i, y_i) = 0$ for $i = 1, 2, \ldots, n$.

This can be done by setting up a system of linear equations over $\mathbb{F}$ with unknowns being the coefficients of $E_2(X)$ and $N_2(X)$, and n homogeneous linear constraints $Q(\alpha_i, y_i) = 0$ in these unknowns. Solving this linear system can certainly be done in polynomial time, and also admits fast, practical methods.

There may be many polynomials $Q \in \mathbb{F}[X, Y]$ satisfying the above constraints, but one can prove that all of them must have $Y - p(X)$ as a factor. To prove this, consider the polynomial $R(X) = Q(X, p(X))$. Whenever $p(a_i) = y_i$, we have $R(a_i) = Q(a_i, p(a_i)) = Q(a_i, y_i) = 0$. So R has at least $(n - \tau)$ distinct roots. The degree of R is at most $\max\{\deg(N_2), \deg(E_2) + k - 1\} \leqslant \tau + k - 1$. Thus if $n - \tau > \tau + k - 1$, i.e., if $\tau \leqslant \frac{n-k}{2}$, R must be the zero polynomial. Recalling that $R(X) = Q(X, p(X))$, this means that $Y - p(X)$ is a factor of $Q(X, Y)$. Note that $p(X)$ can be efficiently computed as the ratio $\frac{N_2(X)}{E_2(X)}$.

This gives an efficient algorithm to correct up to $\frac{n-k}{2}$ errors, or a fraction $\frac{1-R}{2}$ of errors as a function of the rate R. This trade-off is the best possible if we insist on unique recovery of the correct codeword, as it is easy to see that any code of rate R must have two distinct codewords that differ in at most a fraction $1 - R$ of locations. Recall that with list decoding, Theorem 4 says that there exist codes for which it is possible to correct a factor two more errors.

4.2. Reed-Solomon list decoding. We now turn to list decoding Reed-Solomon codes beyond the fraction $(1 - R)/2$ of errors. Before turning to the algorithmic aspects, we pause to comment on a combinatorial fact: using the fact that any two distinct codewords of $\mathrm{RS}_{\mathbb{F},S}[n, k]$ differ in more than $n - k$ positions, it can be shown (via the so-called "Johnson bound," see for instance [17, Chap. 3]) that for a number of errors $\tau \leqslant n - \sqrt{nk} = (1 - \sqrt{R})n$, the size of the list that needs to be output by the decoder is guaranteed to be small (at most $O(n^2)$). Whether one can prove a polynomial list size bound for RS codes for even larger τ or whether the list size must necessarily grow super-polynomially beyond the Johnson bound remains an interesting open question. Some partial results establishing combinatorial limitations of list decoding Reed-Solomon codes appear in [24, 3].

One of the key results in algorithmic list decoding is that RS codes can be list decoded up to the $1 - \sqrt{R}$ bound *efficiently* [48, 29]. By the AM-GM inequality, $1 - \sqrt{R} > (1 - R)/2$ for $R \in (0, 1)$, so this gives an improvement over the traditional bounds for every rate. Further, for $R \to 0$, one can decode when the fraction of errors approaches 100%, thus enabling meaningful recovery even when noise overwhelms the correct information. This qualitative aspect is at the root of the several influential applications of list decoding in complexity theory and cryptography [49], [17, Chap. 12].

4.2.1. Sudan's algorithm. We begin with Sudan's elegant algorithm for list decoding up to the bound of $\approx 1 - \sqrt{2R}$ [48]. The approach is via bivariate polynomial interpolation, based on finding a nonzero polynomial $Q \in \mathbb{F}[X, Y]$ such that every degree $k-1$ polynomial $p \in \mathbb{F}[X]$ that must be output will be among the monic linear factors $Y - p(X)$ of $Q(X, Y)$. The idea is that $Y - p(X)$ passes through at least $n - \tau$ points (a_i, y_i) with relatively low-degree. So if a relatively low-degree Q is interpolated through *all* the points, i.e., $Q(a_i, y_i) = 0$ for every $i \in \{1, 2, \ldots, n\}$, then one might hope that $Y - p(X)$ will emerge as a factor. Of course, this would be impossible if there are too many such target polynomials $p(X)$, but we know that if τ is not too large, there can be at most a few such polynomials. (The argument that the algorithm works does *not* need this bound, and gives an algorithmic proof of the combinatorial bound.)

Lemma 5. *Suppose $Q \in \mathbb{F}[X, Y]$ has degree in X, Y at most d_X, d_Y and satisfies $Q(a_i, y_i) = 0$ for $i \in \{1, 2, \ldots, n\}$. Let p be a degree $k-1$ polynomial p such that $p(a_i) = y_i$ for at least $n - \tau$ values of $i \in \{1, 2, \ldots, n\}$. If $\tau < n - d_X - (k-1)d_Y$, then $Y - p(X)$ must be a factor of $Q(X, Y)$.*

Proof. Consider $R(X) = Q(X, p(X))$. It has degree at most $d_X + (k-1)d_Y$. On the other hand, $R(a_i) = 0$ whenever $p(a_i) = y_i$, and thus it has $n - \tau$ distinct roots. If $n - \tau > d_X + (k-1)d_Y$, we must have $R(X) = 0$, or equivalently $(Y - p(X))|Q(X, Y)$. □

How small a degree suffices for such a non-zero Q to exist? Note that once we guarantee the existence of Q, such a bivariate polynomial can be efficiently found by solving a system of n homogeneous linear equations in the coefficients of its monomials, with one such linear constraint per interpolation condition $Q(a_i, y_i) = 0$. For a non-zero Q to exist, it suffices if the number of monomials of Q exceeds n. With degree bounds d_Y and d_X for the degree of Q in Y and X, the number of monomials is $(d_X + 1)(d_Y + 1)$. If we take $d_Y = \ell$ to be the target list size, $d_X = \lfloor \frac{n}{\ell} \rfloor$ ensures $(d_X + 1)(d_Y + 1) > n$, and therefore the existence of the desired Q. The above lemma then implies that up to τ errors can be list decoded with lists of size ℓ, if $\tau \leqslant n - \left(\frac{n}{\ell} + k\ell\right)$. Taking $\ell = \sqrt{n/k}$ to maximize the upper bound on τ, we get an algorithm to list decode up to $n - 2\sqrt{kn}$ errors, or a fraction $1 - 2\sqrt{R}$ of errors.

What is relevant to Lemma 5 is the "$(1, k-1)$-weighted total degree" $d_X + (k-1)d_Y$ rather than the individual degrees d_X, d_Y. Optimizing for the number of monomials with bounded $(1, k-1)$-weighted total degree, one can improve the fraction of errors list decoded to $\approx 1 - \sqrt{2R}$.

One aspect we did not address is finding the factors $Y - p(X)$ of a bivariate polynomial $Q \in \mathbb{F}[X, Y]$. The task of bivariate polynomial factorization admits polynomial time algorithms [33]. Since the actual task here is easier, it can also be solved by root finding over a suitable extension field, or via a simple randomized reduction to Berlekamp's univariate polynomial factorization algorithm (see [19, Sec. 4.5] for a discussion, or [41] for a more efficient method tailored to finding linear bivariate factors).

4.2.2. The method of multiplicities. We now discuss the work of the author and Sudan [29], which introduced the powerful, if somewhat mysterious, idea of using multiplicities during interpolation to obtain an improved algorithm to list decode RS codes of rate R up to a fraction $1-\sqrt{R}$ of errors. Note that this matches the combinatorial Johnson bound mentioned earlier, and beyond this radius we do not know if the number of codewords is guaranteed to be polynomially bounded, which is an a priori requirement for efficient list decoding.

We do not have space to develop the rationale of using multiplicities via some illustrative examples, and we point the reader to [19, Chap. 4] or [17, Chap. 6] for such a description. Here we sketch the technical aspects of the algorithm and its analysis. Recall that the above algorithm involves two steps: an interpolation step that finds a nonzero polynomial $Q \in \mathbb{F}[X,Y]$, and a factorization/root-finding step where all polynomials $p \in \mathbb{F}[X]$ that must be output are found amongst the factors $Y-p(X)$ of $Q(X,Y)$. The second step will remain the same in the improved algorithm. In the first step, we will demand more from the polynomial Q, namely that it has a zero of multiplicity w at each (a_i, y_i), where $w \geqslant 1$ is the "multiplicity parameter" that governs the performance of the algorithm. (The choice $w = 1$ corresponds to Sudan's algorithm.) This will require an increase in the degree of Q, since for each (a_i, y_i), we will require that all Hasse derivatives of Q at (a_i, y_i) of order up to w vanish, which leads to $\binom{w+1}{2}$ homogeneous linear constraints per point. To ensure that $(d_X+1)(d_Y+1) > n\binom{w+1}{2}$, we can scale d_X, d_Y up by a factor of $\approx w/\sqrt{2}$.

However, this increase is more than compensated in the second step. In particular, the analog of Lemma 5 can conclude that $Q(X, p(X)) = 0$ assuming $w(n-\tau) > d_X + (k-1)d_Y$, thanks to a factor w gain in number of roots at each a_i for which $p(a_i) = y_i$. Optimizing the choice of d_X, d_Y as before, one can obtain an algorithm for list decoding $\approx 1-\sqrt{2R}$ errors. Further, optimizing the $(1, k-1)$-weighted total degree instead of the individual degrees d_X, d_Y, the fraction of errors list decoded improves to $1-\sqrt{R}$. We record the main claim about Reed-Solomon list decoding below.

Theorem 6 ([29]). *For every Reed-Solomon code* $\mathrm{RS}_{\mathbb{F},S}[n,k]$*, there is a list decoding algorithm to correct up to* $n-\sqrt{kn}$ *errors that runs in time polynomial in* $n, |\mathbb{F}|$ *and outputs a list of at most* $O(n^2)$ *polynomials.*[3]

Remark 1. The above algebraic list decoding algorithms, including the one using multiplicities in the interpolation phase, can also be generalized to the family of algebraic-geometric codes (which were briefly described in Section 2.2) [29], [17, Chap. 6]. The algorithm runs in polynomial time given access to a polynomial amount of pre-processed information about the AG code and underlying function field. Further, the algorithm can be generalized to work in an abstract algebraic framework and decode "redundant residue codes" where the messages comprise elements of a ring of bounded size (as per some measure), and are encoded by

[3] In most common instantiations of RS codes, $|\mathbb{F}|$ grows polynomially in n, in fact $|\mathbb{F}| = n$ or $n+1$. But over large fields, one can also have a randomized list decoding algorithm running in time polynomial in $n, \log|\mathbb{F}|$.

their residue modulo a collection of ideals [27], [17, Chap. 7]. A number-theoretic example of such codes are Chinese Remainder codes, where an integer $m \in [0, B)$ is encoded by its residue modulo primes $p_1, p_2, \ldots, p_n$ for $B \ll \prod_{i=1}^{n} p_i$ [15].

Remark 2. The polynomial interpolation method and the method of multiplicities have been recently used with spectacular success to obtain near-tight results on the size of Kakeya sets over finite fields [5, 6].

4.2.3. Soft-decision decoding. The multiplicities based decoding has another benefit, which has received widespread attention from a practical standpoint. This is the ability to exploit "soft information" concerning the reliability of various symbols. When a field element, say the i'th symbol $y_i \in \mathbb{F}$ of a RS codeword, is transmitted on a physical channel, it is "modulated" into some real signal $\Lambda_i \in \mathbb{R}^b$. The channel noise distorts the signal into Λ_i'. The receiver must then "demodulate" this signal into the most likely field element(s), giving a (typically small) set of field elements $\{\alpha_{i,1}, \ldots, \alpha_{i,a}\} \subset \mathbb{F}$ each with an associated weight corresponding to a reliability/confidence estimate . (These confidence estimates are called the "soft" information.)

Multiplicities provide a way to encode this soft information during the interpolation, with larger multiplicities for pairs $(a_i, \alpha_{i,j})$ with a higher weight. By using multiplicities in proportion to the weights, one can prove the following general result about list decoding codewords with large "weighted agreement."

Theorem 7 ([29]). *For the RS code* $\mathrm{RS}_{\mathbb{F},S}[n,k]$ *and any* $\varepsilon > 0$*, there is a decoding algorithm with runtime polynomial in* $n, |\mathbb{F}|, 1/\varepsilon$ *with the following guarantee. Given as input a collection of non-negative rational weights* $W_{i,\alpha}$ *for* $1 \leqslant i \leqslant n$ *and* $\alpha \in \mathbb{F}$*, the algorithm finds a list of all codewords* $c \in \mathrm{RS}_{\mathbb{F},S}[n,k]$ *satisfying*

$$\sum_{i=1}^{n} W_{i,c_i} \geqslant \sqrt{(k-1)\sum_{i,\alpha} W_{i,\alpha}^2} + \varepsilon \max_{i,\alpha} W_{i,\alpha} \ . \tag{1}$$

In the case when for each i, $W_{i,\alpha} = 1$ for $\alpha = y_i$ and 0 otherwise, the above gives the decoding guarantee of Theorem 6 for list decoding $y = (y_1, y_2, \ldots, y_n) \in \mathbb{F}^n$. To put the above result to good use, we need a good way to assign the weights $W_{i,\alpha}$ that capture the likelihood that the i'th symbol y_i equals α. Koetter and Vardy [34] developed a "front end" that chooses weights that are optimal in a certain sense, based on channel observations and transition probabilities, and also made important complexity optimizations in the interpolation step. This yields a soft-decision decoding algorithm for RS codes that has led to good coding gains in practice. It is worth noting that even though list decoding was motivated by worst-case errors, the ideas have also led to important advances for decoding under probabilistic channels. Theorem 7 is also useful in decoding concatenated codes, where the symbols of a RS codeword are further encoded by a binary inner code. Here the weights are obtained by decoding the inner code and are carefully picked to have bounded ℓ_2 norm (for plugging into (1)); see [30], [17, Chap. 8] for further details on such uses of soft decoding of RS codes.

5. List decoding with optimal rate: Folded RS codes

For several years after the publication of [29], there was no improvement to the $1-\sqrt{R}$ trade-off between fraction of list decoded errors and rate R. Also, as mentioned earlier we still do not know if this bound can be improved upon for decoding RS codes. However, some recent work has shown that for some variants of RS codes, one can in fact do better. We now sketch the key ideas in this line of work which culminated with a list-decodable code construction achieving the optimal $1-R-\varepsilon$ trade-off for any desired $\varepsilon > 0$.

5.1. Encoding multiple polynomials. The 1/2 in the exponent of the $1-\sqrt{R}$ decoding radius for RS codes came as a result of bivariate interpolation: a polynomial $Q(X,Y)$ of $(1,k-1)$-weighted degree D has $\approx \frac{D^2}{2k}$ monomials, which leads to a $D = O(\sqrt{kn}) = O(\sqrt{R}n)$ bound on the degree needed to interpolate through n points (and this in turn leads to a $(1-O(\sqrt{R}))n$ bound for the number of errors corrected). If we had the flexibility to interpolate in 3 dimensions, then a polynomial $Q(X,Y,Z)$ with $(1,k-1,k-1)$-weighted D has $\approx \frac{D^3}{6k^2}$ monomials, so $D = O(R^{2/3}n)$ suffices for the interpolating through n points $(a_i,y_i,z_i) \in \mathbb{F}^3$. One could perhaps wishfully hope that this can be the basis of an algorithm for list decoding a fraction $\approx 1-R^{2/3}$ of errors, and more generally, by interpolating in $s+1$ dimensions, a fraction $\approx 1-R^{s/(s+1)}$ of errors, which exceeds $1-R-\varepsilon$ for s chosen large enough.

The RS codeword encoding a polynomial $p \in \mathbb{F}[X]$ could naturally be viewed as a subset $\{(a_i,p(a_i)) \mid i=1,2,\ldots,n\} \subseteq \mathbb{F}^2$ on which a bivariate polynomial can be interpolated. To interpolate a trivariate polynomial, we need a set of points in $\mathbb{F}^3$ that correspond in some natural way to the (noisy) codeword. Consider the variant of Reed-Solomon codes with evaluation points $S = \{a_1,\ldots,a_n\} \subseteq \mathbb{F}$ where the message consists of an arbitrary *pair* of polynomials $p_1,p_2 \in \mathbb{F}[X]$, each of degree at most $k-1$, which are encoded into a codeword $c \in (\mathbb{F}^2)^n$ (over alphabet $\mathbb{F}\times\mathbb{F}$) where $c_i = (p_1(a_i),p_2(a_i))$. Note that the rate R of this code is the same as that of the RS code, namely k/n. Now the received word to be decoded consists of $(y_i,z_i) \in \mathbb{F}^2$ for $1 \leqslant i \leqslant n$, and we can interpolate a nonzero $Q \in \mathbb{F}[X,Y,Z]$ of $(1,k-1,k-1)$-weighted degree at most D (for a suitably chosen degree bound D) such that $Q(a_i,y_i,z_i) = 0$ for $i = 1,2,\ldots,n$.

By picking the parameter D appropriately and also enforcing multiplicities in the interpolation step, it is not hard to prove the following claim by proceeding along the lines of Lemma 5 and ensuing arguments in Section 4.2:

$$\begin{gathered}\text{If } (p_1(a_i),p_2(a_i)) = (y_i,z_i) \text{ for at least } t \text{ values of } i, \text{ and} \\ t \geqslant (1+o(1))\sqrt[3]{k^2n} \approx R^{2/3}n, \text{ then } Q(X,p_1(X),p_2(X)) = 0 \end{gathered} \tag{2}$$

If we could efficiently determine all pairs of polynomials (f,g) of degree at most $k-1$ that satisfy $Q(X,f(X),g(X)) = 0$ for a given low-degree $Q \in \mathbb{F}[X,Y,Z]$, we would be done with list decoding all the pairs (p_1,p_2) whose encoding agrees with the received word on $t \approx R^{2/3}n$ positions. Unfortunately, this algebraic task is

impossible in general, as there can be exponentially many (at least $|\mathbb{F}|^k$) solutions (f, g) to $Q(X, f(X), g(X)) = 0$. (As a simple example, consider $Q(X, Y, Z) = Y - Z$; now (f, f) is a solution for every $f \in \mathbb{F}[X]$ of degree at most $k - 1$.)

5.2. Parvaresh-Vardy codes. In the above scheme, we could only obtain one algebraic relation between p_1, p_2, whereas two algebraically independent relations are needed to pin down p_1, p_2 to a small number of possibilities. To circumvent this problem, Parvaresh and Vardy [38] put forth the ingenious idea of obtaining the extra algebraic relation essentially "as default," by enforcing it as an *a priori* condition satisfied at the encoder. Specifically, instead of letting the two message polynomials (p_1, p_2) be independent and uncorrelated, they required them to satisfy an appropriate algebraic condition. For instance, as a concrete choice, one can insist that $p_2(X) = p_1(X)^h \mod E(X)$ for some monic $E \in \mathbb{F}[X]$ of degree k that is irreducible over $\mathbb{F}$, and a suitable exponent h. In this case, the message of the new code is just a degree $k - 1$ polynomial p_1 (as with RS codes), but it is encoded by the evaluations of both p_1, and $p_2 = p_1^h \mod E$, at the points a_i.

Given a nonzero polynomial Q, one can now determine a list of all such messages p_1 for which $Q(X, p_1(X), p_2(X)) = 0$, by recovering the residue $\overline{p_1} = p_1(X) \mod E(X)$ of p_1 in the extension field $K = \mathbb{F}[X]/(E(X))$. The key is that this residue $\overline{p_1}$ is a root of the low-degree polynomial $S \in K[T]$ defined by $S(T) = Q(X, T, T^h) \mod E(X)$. (Some care is required to ensure that S is in fact nonzero; in particular the exponent h must be chosen to be larger than the degree of Q in Y, Z, but these are just technicalities and ignored in this description.)

This transformation of the RS codes is, however, not for free, and costs heavily in terms of the rate. The rate of the Parvaresh-Vardy code is $\frac{k}{2n}$ — half that of the original RS code — since the encoding has twice as many symbols (evaluations of *both* p_1, p_2). Plugging into the Claim (2), Parvaresh and Vardy get codes of rate R list-decodable up to a fraction $1 - (2R)^{2/3}$ errors. For $R < 1/16$, this gives a small improvement over the $1 - \sqrt{R}$ bound for RS codes. The natural extension of this idea to encoding s polynomials and using $(s+1)$-variate interpolation gives a decoding radius $1 - (sR)^{s/(s+1)}$, and (optimizing in s) a decoding radius $1 - O(R \log(1/R))$ for low rates $R \to 0$. Unfortunately, the improvement over RS codes is confined to low rates.

5.3. Folded RS codes. We now turn to the work of the author and Rudra on optimal rate list-decodable codes [25], obtained by a "folding" of the Reed-Solomon code. The big rate loss in the Parvaresh-Vardy code construction was due to the inclusion of the evaluations of a second polynomial p_2 along with those of the message polynomial p_1. Note that the codes by construction can never have a rate exceeding $1/2$, whereas we would like list-decodable codes of rate R for any desired $R \in (0, 1)$.

5.3.1. Algebra behind folding. To motivate the approach, consider the code where instead of picking p_2 as above, we take $p_2(X) = p_1(-X)$ (let us assume

the characteristic of $\mathbb{F}$ is not 2; if the field has characteristic 2 we can take $p_2(X) = p_1(X+a)$ for some $a \in \mathbb{F}^*$). Also assume that $n = |\mathbb{F}^*|$ and the evaluation points are ordered so that $a_{2i} = -a_{2i-1}$ for $1 \leqslant i \leqslant n/2$. In the encoding of p_1 by the evaluations of (p_1, p_2) at $\{a_1, \ldots, a_n\}$, note that $(p_1(a_{2i-1}), p_2(a_{2i-1})) = (p_1(a_{2i-1}), p_1(-a_{2i-1})) = (p_1(a_{2i-1}), p_1(a_{2i})) = (p_2(a_{2i}), p_1(a_{2i}))$. So the codeword symbols at locations $2i-1, 2i$ are the same up to the ordering within the pair. Therefore, we can compress the codeword with no loss in information, by deleting half of the evaluations $(p_1(a_{2i}), p_2(a_{2i}))$. This recovers the factor 2 lost in the Parvaresh-Vardy construction. In fact, note the resulting code is essentially just the RS code, but considered as a code over $\mathbb{F}^2$ of length $n/2$, by "bundling" together the evaluations at $\alpha, -\alpha$ together for each $\alpha \in \mathbb{F}$.

The argument described in Section 5.1 above shows that all polynomials p_1 differing from the received word on at most a fraction $1 - R^{2/3}$ of places must satisfy $Q(X, p_1(X), p_1(-X)) = 0$. So the question once again is how many such polynomials can exist, and whether one can find them all efficiently? Unfortunately, there could still be exponentially many such solutions. For instance, consider $Q(X, Y, Z) = Y - Z$. For every $f \in \mathbb{F}[X]$ of degree at most $(k-1)/2$, $p_1(X) = f(X^2)$ clearly satisfies $Q(X, p_1(X), p_1(-X)) = f(X^2) - f((-X)^2) = 0$.

The above example also shows that there can be $\approx |\mathbb{F}|^{k/r}$ polynomials $p_1 \in \mathbb{F}[X]$ of degree less than k satisfying $Q(X, p_1(X), p_1(\alpha X)) = 0$ for a nonzero α, if the order of α in the multiplicative group $\mathbb{F}^*$ is r. Thus to get a reasonable (polynomial in $n, |\mathbb{F}|$) upper bound on the number of solutions, the multiplicative order of α must be large.

The following lemma shows that a form of converse also holds. It is the key algebraic fact underlying the list decoding algorithm for folded RS codes.

Lemma 8. *Let $Q \in \mathbb{F}_q[X, Y, Z]$ be a nonzero polynomial with degrees d_Y, d_Z in Y, Z less than q. Let γ be a primitive element in $\mathbb{F}_q$. Let $1 \leqslant k < q$. The number of degree $k-1$ polynomials $f \in \mathbb{F}_q[X]$ such that $Q(X, f(X), f(\gamma X)) = 0$ is at most $d_Y + q d_Z < q^2$, and a list of all such f can be found in time polynomial in q.*

For the formal proof, see [25] or [19, Sec. 6.4]. The algebraic crux is the following identity for every polynomial $f \in \mathbb{F}_q[X]$:

$$f(\gamma X) \equiv f(X)^q \pmod{(X^{q-1} - \gamma)} . \tag{3}$$

Further, the polynomial $P(X) = X^{q-1} - \gamma$ is irreducible over $\mathbb{F}_q$. Therefore, the condition $Q(X, f(X), f(\gamma X)) = 0$ implies the equation $T(\bar{f}, \bar{f}^q) = 0$ where $\bar{f} = f \mod P(X)$ and $T(Y, Z) = Q(X, Y, Z) \mod P(X)$ is a polynomial with coefficients from the extension field $\mathbb{F}_q[X]/(P(X))$. This implies that $\bar{f}$, and hence f if its degree is less than $q-1$, can be found amongst the roots of the polynomial $T(Y, Y^q)$, which has degree at most $d_Y + q d_Z$, over the field $\mathbb{F}_q[X]/(P(X))$.

Analogously, the following generalization holds for higher order interpolation:

Lemma 9. *Let $Q \in \mathbb{F}_q[X, Y_1, Y_2, \ldots, Y_s]$ be a nonzero polynomial with the degree in the Y_i's less than q, and let γ be a primitive element in $\mathbb{F}_q$. Then there are at*

most q^s polynomials $f \in \mathbb{F}_q[X]$ of degree less than $(q-1)$ satisfying

$$Q(X, f(X), f(\gamma X), \ldots, f(\gamma^{s-1}X)) = 0 \ .$$

Further, a list of all these polynomials can be found in $q^{O(s)}$ time.

5.3.2. Code description and main result. Unlike the code where we bundled together $f(\alpha), f(-\alpha)$ together, when γ is primitive (or has high order), one cannot bundle together the evaluations of f on an whole orbit of the action by multiplication by γ. The folded RS code proposed in [25] consists of bundling together consecutive m symbols of the RS codeword $(f(1), f(\gamma), \ldots, f(\gamma^{n-1}))$ for some fixed integer constant m called the *folding parameter* (which we assume for convenience divides the block length $n = q-1$). Formally,

Definition 3 (Folded Reed-Solomon Code). The m-folded version of the RS code $\mathrm{RS}_{\mathbb{F}_q, \mathbb{F}_q^*}[n, k]$ is a code of block length $N = n/m$ over the alphabet $\mathbb{F}_q^m$. The encoding of a polynomial $f \in \mathbb{F}_q[X]$ of degree at most $k-1$, has as its j'th symbol, for $0 \leqslant j < n/m$, the m-tuple $(f(\gamma^{jm}), f(\gamma^{jm+1}), \cdots, f(\gamma^{jm+m-1}))$.

The folded version of a RS code thus carries the same information, just "bundled" differently. It is a code of exactly the same rate as the original RS code, but is defined over a larger alphabet. At a high level, folding restricts the flexibility in the subset of evaluation points that an adversary can corrupt. We now state the main result concerning decoding these codes from [25].

Theorem 10. *For every positive integer m and integer s, $1 \leqslant s \leqslant m$, the folded RS code with the parameters q, n, N, k from Definition (3) can be list decoded up to a radius*

$$N - (1 + o(1)) \frac{(k^s n)^{1/(s+1)}}{m - s + 1} \ , \tag{4}$$

in time at most $q^{O(s)}$, and the list size output by the decoder will be at most q^s.

The parameter s corresponds to the number of dimensions in the interpolation, and the stated bound is obtained through $(s+1)$-variate interpolation. Specifically, the decoding algorithm interpolates a low-degree nonzero polynomial $Q \in \mathbb{F}_q[X, Y_1, Y_2, \ldots, Y_s]$ such that any message f whose folded RS encoding is within distance (4) from the received word must satisfy $Q(X, f(X), f(\gamma X), \ldots, f(\gamma^{s-1}X))$. The list of all such degree $< k$ polynomials can be efficiently found by Lemma 9.

By picking m large enough compared to s, and noting that the rate $R = k/n$ and $n = Nm$, the fraction of decoded errors can be made larger than $1-(1+\zeta)R^{s/(s+1)}$ for any desired $\zeta > 0$. In the limit of large s (specifically, for $s = \Theta(\varepsilon^{-1}\log(1/R))$), the fraction of errors corrected approaches $1 - R$, leading to the main conclusion about optimal rate list-decodable codes.

Theorem 11. *For every $\varepsilon > 0$ and $0 < R < 1$, there is a family of folded Reed-Solomon codes which have rate at least R and which can be list decoded up to a fraction $1-R-\varepsilon$ of errors in time $(N/\varepsilon^2)^{O(\varepsilon^{-1}\log(1/R))}$ where N is the block length*

of the code. The alphabet size of the code as a function of the block length N is $(N/\varepsilon^2)^{O(1/\varepsilon^2)}$.

Remark 3. The large alphabet size and decoding complexity in the above result are a shortcoming. Fortunately, the alphabet size can be reduced to a constant depending only on ε, and in fact with a dependence that is not far from optimal. We will sketch this in the next section.

Open Problem 4. *Can one improve the decoding complexity (for list decoding up to a fraction $1 - R - \varepsilon$ of errors) to have a better dependence on $1/\varepsilon$, and/or guarantee that the list size will be bounded by a constant independent of n?*

Recall that by the existential result in Theorem 4, a list size of $O(1/\varepsilon)$ suffices.

Remark 4 (Connection to Artin-Frobenius automorphisms). The algebraic crux of the folded RS decoder, identity (3), was that the automorphism Γ of the function field $K = \mathbb{F}_q(X)$ induced by $X \mapsto \gamma X$ satisfied the identity $\Gamma(f) \equiv f^q \pmod{P(X)}$ for all polynomials f, where $P(X) = X^{q-1} - \gamma$. That is, the automorphism Γ induces a low-degree map w.r.t the evaluations of polynomials at the "place" corresponding to $P(X)$. In general, one can obtain such low-degree relations between residues via the Frobenius automorphisms of places in Galois extensions. Specifically, if a place P of a field K is inert in a finite Galois extension L/K with a place P' in L above it, then the Frobenius automorphism Frob_P satisfies $\mathrm{Frob}_P(x) = x^{\|P\|} \pmod{P'}$ for every $x \in L$ that is regular at P', where $\|P\|$ is the size of the residue field at P. Using this approach, we were able to extend the folded RS code construction to folded versions of certain algebraic-geometric codes based on cyclotomic function fields [20].

6. List-decodable codes over smaller alphabets

6.1. Binary codes. Let us now discuss results on constructing binary list-decodable codes. The existential result of Theorem 3 says that there exist $(\rho, O(1/\varepsilon))$-list decodable codes of rate $1 - h(\rho) - \varepsilon$, for any desired error fraction $\rho \in (0, 1/2)$. However, unlike large alphabets, an explicit construction with rate approaching the optimal bound of $1 - h(\rho)$ is not known, and the following is a *major* open question.

Open Problem 5. *Fix $\rho \in (0, 1/2)$. Can one give an explicit construction of binary codes of rate $1 - h(\rho) - \varepsilon$ for any desired constant $\varepsilon > 0$ that are (ρ, ℓ)-list decodable, even for a list-size ℓ that grows as a polynomially bounded function $\ell(n) = n^{O_\varepsilon(1)}$ of the block length n of the code?*

Can one construct such codes together with an efficient list decoding algorithm that on input $y \in \{0,1\}^n$ outputs the list of codewords within Hamming distance ρn of y in polynomial time.

Though the above challenge is wide open, one can construct binary list-decodable codes achieving a reasonable trade-off, the so-called Zyablov bound, between the

error fraction ρ and rate R, for any $\rho \in (0, 1/2)$. The bound below appears somewhat complicated, and a useful regime to compare it with the optimal rate of $1 - h(\rho)$ is when $\rho = 1/2 - \gamma$ for $\gamma \to 0$: in this case $1 - h(\rho) \approx \gamma^2$ whereas the Zyablov bound is only $\approx \gamma^3$.

Theorem 12 ([25]). *For every $\rho \in (0, 1/2)$ and $\varepsilon > 0$, there is a polynomial time constructible family of binary codes of rate at least*

$$\max_{\substack{\rho_1 \rho_2 = \rho \\ \rho_1 < 1, \rho_2 < 1/2}} (1 - \rho_1)(1 - h(\rho_2)) - \varepsilon$$

that can be list decoded in polynomial time up to a fraction ρ of errors.

The idea behind the above claim is *code concatenation*, where we first encode the message as per an "outer" code over a large alphabet, and then each symbol is further encoded by an "inner" binary code which has the optimal trade-off between rate and list-decodability. The inner code is of small enough length that one can find such a good code essentially by brute-force in time polynomial in the overall code length.

Theorem 12 is proved by concatenating the folded RS codes guaranteed by Theorem 11 of rate $\approx 1 - \rho_1$ with inner codes of rate $\approx 1 - h(\rho_2)$ that are $(\rho_2, \ell = O_\varepsilon(1))$-list decodable (as guaranteed to exist by Theorem 3). The idea behind the decoding is to first list decode the various inner blocks up to fractional radius ρ_2. This gives a list of size at most ℓ for the possible symbols at each position of the outer codeword. Every folded RS codeword that must be output can have at most a fraction ρ_1 of symbols which do not belong to the respective lists. Now while the folded RS code of rate $\approx 1 - \rho_1$ can recover from a fraction ρ_1 of errors, we now have a harder problem, as for each position we do not have a unique symbol but a list of ℓ possible symbols. Somewhat remarkably, folded RS codes can handle such bounded size lists with no loss in rate! (The alphabet size will depend on ℓ; see [25] for exact details.) This extension to list decoding is called "list recovery" and is a powerful primitive that is useful in composing list-decodable codes together.

6.2. Optimal rate list-decodable codes over fixed alphabets.

The powerful list recovery properties offered by folded RS codes, together with techniques based on expander graphs to redistribute symbols of codewords, can also be used to attain the optimal trade-off between error-correction radius and rate of Theorem 11 over an alphabet of fixed size depending only on ε. For details, see [25] or [19, Chap. 7].

Theorem 13. *For every $R \in (0, 1)$ and $\varepsilon > 0$, there is a polynomial time constructible family of codes over an alphabet of size $2^{O(\varepsilon^{-4} \log(1/\varepsilon)}$ that have rate at least R and can be list decoded up to a fraction $(1 - R - \varepsilon)$ of errors in polynomial time.*

Remark 5. $2^{\Omega(1/\varepsilon)}$ is a lower bound on the alphabet size needed for list decoding up to a fraction $1 - R - \varepsilon$ of errors with rate R, so the above alphabet size is in the right ballpark.

7. Alternate bridges between worst-case and random errors

We conclude this survey with a brief discussion of some other models besides list decoding that can be used to handle worst-case errors with rates similar to random errors. One of these is to allow randomized coding strategies where the sender and receiver share secret randomness (hidden from the channel) that is used to pick a coding scheme at random from a family of codes. Using such strategies, one can achieve a rate approaching $1 - h(\rho)$ for communicating on the adversarial channel ADV_ρ (for example, by randomly permuting the symbols and adding a random offset to codes achieving capacity on BSC_ρ).

The amount of shared randomness in the above setting can be reduced if we make computational assumptions on the channel [35] — the encoder and decoder only need to share a private seed for a pseudorandom generator. One can also reduce the shared randomness to logarithmic amounts without computational assumptions by using list-decodable codes together with standard message authentication schemes [46]. It is also possible to eliminate the shared randomness and instead require a public key [37], and this solution also relies on list-decodable codes. If explicit list-decodable codes of rate approaching $1 - h(\rho)$ for correcting a fraction ρ of errors were constructed, these would imply optimal rate explicit codes in these models as well.

In the ADV_ρ model, the channel picks the error vector e after seeing the codeword. A weaker model is that of an oblivious additive channel that can pick any worst-case error vector e (in particular the e need not have any specific distribution such as the binomial distribution), but must do so before seeing the codeword. The following result is known in this model [4]: there *exist* binary codes with encoding function $E : \{0,1\}^k \to \{0,1\}^n$ of rate approaching $1 - h(\rho)$ and a decoding function D such that for *every* error vector e of Hamming weight at most ρn, for *most* messages $m \in \{0,1\}^k$, $D(E(m) + e) = e$. Note the quantifiers are flipped compared to Shannon's result for BSC_ρ: instead of decoding most error vectors for every message, we can decode most messages for every error vector. It was recently shown how one can get codes with this property by combining list-decodable codes with certain algebraic manipulation detection codes [28]. Once again, this shows the versatility of the primitive of list decoding, and highlights the importance of constructing better explicit list-decodable codes.

We have mentioned several open questions in this survey, and we end by reiterating the central Open Problem 5 on the challenge of an explicit construction of binary codes of rate approaching $1 - h(\rho)$ for list decoding up to a fraction ρ of errors.

References

[1] N. Alon and M. Luby. A linear time erasure-resilient code with nearly optimal recovery. *IEEE Transactions on Information Theory*, 42(6):1732–1736, 1996. 6

[2] S. Ar, R. Lipton, R. Rubinfeld, and M. Sudan. Reconstructing algebraic functions from mixed data. *SIAM Journal on Computing*, 28(2):488–511, 1999. 4

[3] E. Ben-Sasson, S. Kopparty, and J. Radhakrishnan. Subspace polynomials and list decoding of Reed-Solomon codes. In *Proceedings of the 47th Annual Symposium on Foundations of Computer Science*, pages 207–216, 2006. 12

[4] I. Csiszár and P. Narayan. The capacity of the arbitrarily varying channel revisited: Positivity, constraints. *IEEE Trans. on Info. Theory*, 34(2):181–193, 1988. 22

[5] Z. Dvir. On the size of Kakeya sets in finite fields. *J. Amer. Math. Soc.*, 22:1093–1097, 2009. 15

[6] Z. Dvir, S. Kopparty, S. Saraf, and M. Sudan. Extensions to the method of multiplicities, with applications to Kakeya sets and mergers. In *Proceedings of the 50th Symposium on Foundations of Computer Science*, pages 181–190, 2009. 15

[7] P. Elias. Coding for two noisy channels. *Information Theory, Third London Symposium*, pages 61–76, September 1955. 7

[8] P. Elias. List decoding for noisy channels. *Technical Report 335, Research Laboratory of Electronics, MIT*, 1957. 4

[9] P. Elias. Error-correcting codes for list decoding. *IEEE Transactions on Information Theory*, 37:5–12, 1991. 9

[10] G. D. Forney. *Concatenated Codes*. MIT Press, Cambridge, MA, 1966. 9

[11] G. D. Forney. Generalized Minimum Distance decoding. *IEEE Transactions on Information Theory*, 12:125–131, 1966. 7

[12] A. Garcia and H. Stichtenoth. A tower of Artin-Schreier extensions of function fields attaining the Drinfeld-Vlădut bound. *Inventiones Math.*, 121:211–222, 1995. 6

[13] P. Gemmell and M. Sudan. Highly resilient correctors for multivariate polynomials. *Information Processing Letters*, 43(4):169–174, 1992. 11

[14] O. Goldreich and L. Levin. A hard-core predicate for all one-way functions. In *Proceedings of the 21st Annual ACM Symposium on Theory of Computing*, pages 25–32, May 1989. 4

[15] O. Goldreich, D. Ron, and M. Sudan. Chinese remaindering with errors. *IEEE Transactions on Information Theory*, 46(5):1330–1338, 2000. 15

[16] V. D. Goppa. Codes on algebraic curves. *Soviet Math. Doklady*, 24:170–172, 1981. 6

[17] V. Guruswami. *List decoding of error-correcting codes*. Number 3282 in Lecture Notes in Computer Science. Springer, 2004. 4, 8, 12, 14, 15, 16

[18] V. Guruswami. Iterative Decoding of Low-Density Parity Check Codes. *Bulletin of the European Association for Theoretical Computer Science*, 90, 2006. 7, 9

[19] V. Guruswami. *Algorithmic Results in List Decoding*, volume 2 of *Foundations and Trends in Theoretical Computer Science*. NOW publishers, 2007. 4, 14, 18, 22

[20] V. Guruswami. Cyclotomic function fields, Artin-Frobenius automorphisms, and list error-correction with optimal rate. *Algebra and Number Theory*, 2010. Accepted for publication. Preliminary version in 41st ACM Symp. on Theory of Computing. 20

[21] V. Guruswami, J. Håstad, and S. Kopparty. On the list-decodability of random linear codes. In *Proceedings of the 42th ACM Symposium on Theory of Computing*, 2010. To appear. Available at http://arxiv.org/abs/1001.1386. 10

[22] V. Guruswami, J. Håstad, M. Sudan, and D. Zuckerman. Combinatorial bounds for list decoding. *IEEE Transactions on Information Theory*, 48(5):1021–1035, 2002. 10

[23] V. Guruswami and P. Indyk. Linear-time encodable/decodable codes with near-optimal rate. *IEEE Trans. on Information Theory*, 51(10):3393–3400, 2005. 6

[24] V. Guruswami and A. Rudra. Limits to list decoding Reed-Solomon codes. *IEEE Transactions on Information Theory*, 52(8):3642–3649, 2006. 12

[25] V. Guruswami and A. Rudra. Explicit codes achieving list decoding capacity: Error-correction with optimal redundancy. *IEEE Transactions on Information Theory*, 54(1):135–150, 2008. 11, 18, 19, 21, 22

[26] V. Guruswami and A. Rudra. Error-correction up to the information-theoretic limit. *Communications of the ACM*, 52(3):87–95, March 2009. 4, 11

[27] V. Guruswami, A. Sahai, and M. Sudan. Soft-decision decoding of Chinese Remainder codes. In *Proceedings of the 41st IEEE Symposium on Foundations of Computer Science*, pages 159–168, 2000. 15

[28] V. Guruswami and A. Smith. Explicit capacity-achieving codes for worst-case additive errors. *CoRR*, abs/0912.0965, 2009. 23

[29] V. Guruswami and M. Sudan. Improved decoding of Reed-Solomon and algebraic-geometric codes. *IEEE Transactions on Information Theory*, 45:1757–1767, 1999. 12, 14, 15, 16

[30] V. Guruswami and M. Sudan. List decoding algorithms for certain concatenated codes. In *32nd ACM Symposium on Theory of Computing*, pages 181–190, 2000. 16

[31] V. Guruswami, C. Umans, and S. Vadhan. Unbalanced expanders and randomness extractors from Parvaresh-Vardy codes. *Journal of the ACM*, 56(4), 2009. 4

[32] R. W. Hamming. Error Detecting and Error Correcting Codes. *Bell System Technical Journal*, 29:147–160, April 1950. 2

[33] E. Kaltofen. Polynomial factorization 1987–1991. *Proceedings of LATIN '92, I. Simon (Ed.), Springer LNCS*, 583:294–313, 1992. 13

[34] R. Koetter and A. Vardy. Algebraic soft-decision decoding of Reed-Solomon codes. *IEEE Transactions on Information Theory*, 49(11):2809–2825, 2003. 15

[35] R. J. Lipton. A new approach to information theory. In *Proceedings of the 11th Annual Symposium on Theoretical Aspects of Computer Science*, pages 699–708, 1994. 22

[36] R. J. McEliece and L. Swanson. On the decoder error probability for Reed-Solomon codes. *IEEE Transactions on Information Theory*, 32(5):701–703, 1986. 3

[37] S. Micali, C. Peikert, M. Sudan, and D. A. Wilson. Optimal error correction against computationally bounded noise. In *Proceedings of the 2nd Theory of Cryptography Conference (TCC)*, pages 1–16, 2005. 22

[38] 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. 17

[39] W. W. Peterson. Encoding and error-correction procedures for Bose-Chaudhuri codes. *IEEE Transactions on Information Theory*, 6:459–470, 1960. 11

[40] T. Richardson and R. Urbanke. *Modern Coding Theory*. Cambridge University Press, 2008. 7

[41] R. Roth and G. Ruckenstein. Efficient decoding of Reed-Solomon codes beyond half the minimum distance. *IEEE Trans. on Info. Theory*, 46(1):246–257, 2000. 14

[42] A. Rudra and S. Uurtamo. Two theorems in list decoding. *CoRR*, abs/1001.1781, 2010. 3

[43] R. Shaltiel and C. Umans. Simple extractors for all min-entropies and a new pseudorandom generator. *Journal of the ACM*, 52(2):172–216, 2005. 4

[44] C. E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423, 623–656, 1948. 2

[45] K. W. Shum, I. Aleshnikov, P. V. Kumar, H. Stichtenoth, and V. Deolalikar. A low-complexity algorithm for the construction of algebraic-geometric codes better than the Gilbert-Varshamov bound. *IEEE Trans. on Information Theory*, 47(6):2225–2241, 2001. 6

[46] A. Smith. Scrambling adversarial errors using few random bits, optimal information reconciliation, and better private codes. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 395–404, 2007. 22

[47] H. Stichtenoth. *Algebraic Function Fields and Codes*. Universitext, Springer-Verlag, Berlin, 1993. 6

[48] M. Sudan. Decoding of Reed-Solomon codes beyond the error-correction bound. *Journal of Complexity*, 13(1):180–193, 1997. 4, 12, 13

[49] M. Sudan. List decoding: Algorithms and applications. *SIGACT News*, 31:16–27, 2000. 4, 12

[50] L. Trevisan. Some applications of coding theory in computational complexity. *Quaderni di Matematica*, 13:347–424, 2004. 4

[51] M. A. Tsfasman, S. G. Vlădut, and T. Zink. Modular curves, Shimura curves, and codes better than the Varshamov-Gilbert bound. *Math. Nachrichten*, 109:21–28, 1982. 6

[52] S. P. Vadhan. The unified theory of pseudorandomness. *SIGACT News*, 38(3):39–54, 2007. 4

[53] L. R. Welch and E. R. Berlekamp. Error correction of algebraic block codes. *US Patent Number 4,633,470*, December 1986. 11

[54] J. M. Wozencraft. List Decoding. *Quarterly Progress Report, Research Laboratory of Electronics, MIT*, 48:90–95, 1958. 4

[55] 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. 4, 9

Computer Science Department, Carnegie Mellon University, Pittsburgh, PA 15213, USA.
E-mail: guruswami@cmu.edu