Electronic Colloquium on Computational Complexity, Report No. 104 (2017)

Local List Recovery of High-rate Tensor Codes & Applications

Brett Hemenway* Noga Ron-Zewi† Mary Wootters‡

June 11, 2017

Abstract

In this work, we give the first construction of *high-rate* locally list-recoverable codes. List-recovery has been an extremely useful building block in coding theory, and our motivation is to use these codes as such a building block. In particular, our construction gives the first *capacity-achieving* locally list-decodable codes (over constant-sized alphabet); the first *capacity achieving* globally list-decodable codes with nearly linear time list decoding algorithm (once more, over constant-sized alphabet); and a randomized construction of binary codes on the Gilbert-Varshamov bound that can be uniquely decoded in near-linear-time, with higher rate than was previously known.

Our techniques are actually quite simple, and are inspired by an approach of Gopalan, Guruswami, and Raghavendra (Siam Journal on Computing, 2011) for list-decoding tensor codes. We show that tensor powers of (globally) list-recoverable codes are 'approximately' locally list-recoverable, and that the 'approximately' modifier may be removed by pre-encoding the message with a suitable locally decodable code. Instantiating this with known constructions of high-rate globally list-recoverable codes and high-rate locally decodable codes finishes the construction.

1 Introduction

List-recovery refers to the problem of decoding error correcting codes from "soft" information. More precisely, given a *code* $C : \Sigma^k \to \Sigma^n$, which maps length-$k$ *messages* to length-n *codewords,* an (α, ℓ, L)-list-recovery algorithm for C is provided with a sequence of lists $S_1, \ldots, S_n \subset \Sigma$ of size at most ℓ each, and is tasked with efficiently returning all messages $x \in \Sigma^k$ so that $C(x)_i \notin S_i$ for at most α fraction of the coordinates i; the guarantee is that there are no more than L such messages. The goal is to design codes C which simultaneously admit such algorithms, and which also have other desirable properties, like high *rate* (that is, the ratio k/n, which captures how much information can be sent using the code) or small *alphabet size* $|\Sigma|$. List-recovery is a generalization of *list-decoding,* which is the situation when the lists S_i have size one: we refer to $(\alpha, 1, L)$-list-recovery as (α, L)-list-decoding.

List recoverable codes were first studied in the context of list-decoding and soft-decoding. The celebrated Guruswami-Sudan list-decoding algorithm [GS99] is in fact a list-recovery algorithm, as are several more recent list-decoding algorithms [GR08b, GW11, Kop15, GX13]. Initially, list

*fbrett@cis.upenn.edu. Department of Computer Science, University of Pennsylvania

†nogazewi@cs.bgu.ac.il. Department of Computer Science, Ben-Gurion University

‡marykw@stanford.edu. Departments of Computer Science and Electrical Engineering, Stanford University

ISSN 1433-8092

recoverable codes were used as stepping stones towards constructions of list decodable and uniquely decodable codes [GI01, GI02, GI03, GI04]. Since then, list recoverable codes have found additional applications in the areas of compressed sensing, combinatorial group testing, and hashing [INR10, NPR12, GNP$^+$13, HIOS15].

Locality is another frequent desideratum in coding theory. Loosely, an algorithm is "local" if information about a single coordinate x_i of a message x of C can be determined locally from only a few coordinates of a corrupted version of $C(x)$. Locality, and in particular local list-decoding, has been implicit in theoretical computer science for decades: for example, local list-decoding algorithms are at the heart of algorithms in cryptography [GL89], learning theory [KM93], and hardness amplification and derandomization [STV01].

The actual definition of a locally list-recoverable (or locally list-decodable) code requires some subtlety: we want to return a list of answers, and we want the algorithm to be local (having to do with a single message coordinate), but returning a list of possible symbols in Σ for a single message coordinate is pretty useless if that was our input to begin with (at least if the code is systematic in which case the message coordinates are a subset of the codeword coordinates). Instead, we require that a local list-recovery algorithm returns a list $A_1, \ldots, A_L$ of randomized local algorithms. Each of these algorithms takes an index $i \in [k]$ as input, and has oracle access to the lists $S_1, \ldots, S_n$. The algorithm then makes at most Q queries to this oracle (that is, it sees at most Q different lists S_i), and must return a guess for x_i, where x is a message whose encoding $C(x)$ agrees with many of the lists. The guarantee is that for all such x—that is, for all x whose encoding $C(x)$ agrees with many of the lists—there exists (with high probability) some A_j so that for all i, $A_j(i) = x_i$ with probability at least $2/3$. The parameter Q is called the *query complexity* of the local list-recovery algorithm.

One reason to study local list-recoverability is that list-recovery is a very useful building block throughout coding theory. In particular, the problem of constructing **high rate locally list-recoverable codes** (of rate arbitrarily close to 1, and in particular non-decreasing as a function of ℓ) has been on the radar for a while, because such codes would have implications in local list-decoding, global list-decoding, and classical unique decoding.

In this work, we give the first constructions of high-rate locally list-recoverable codes. As promised, these lead to several applications throughout coding theory. Moreover, our construction is actually quite simple. Our approach is inspired by the list-decoding algorithm of [GGR11] for tensor codes, and our main observation is that this algorithm—with a few tweaks—can be made local.

1.1 Results

We highlight our main results below—we will elaborate more on these results and their context within related literature next in Section 2.

High-rate local list-recovery. Our main technical contribution is the first constructions of high-rate locally list-recoverable codes: Theorems 5.5 and 5.6 give the formal statements. Theorem 5.5 can guarantee high-rate list recovery with query complexity $n^{1/t}$ (for constant t, say $n^{0.001}$), constant alphabet size and *constant* output list size, although without an explicit construction or an efficient list recovery algorithm. Theorem 5.6 on the other hand gives an explicit and efficient version, at the cost of a slightly super-constant output list size (which depends on $\log^* n$).

For those familiar with the area, it may be somewhat surprising that this was not known before: indeed, as discussed below in Section 2, we know of locally list recoverable codes (of low rate), and we also know of high-rate (globally) list-recoverable codes. One might think that our result is lurking implicitly in those earlier works. However, it turns out that it is not so simple: as discussed below, existing techniques for locally or globally list-recoverable codes do not seem to work for this problem. Indeed, some of those prior works [HW15, KMRS16, GKO$^+$17] (which involve the current authors) began with the goal of obtaining high-rate locally list-recoverable codes and ended up somewhere else.

This raises the question: why might one seek high-rate locally list-recoverable error correcting codes in the first place? The motivation is deeper than a desire to add adjectives in front of "error correcting codes." As we will see below, via a number of reductions that already exist in the literature, such codes directly lead to improvements for several other fundamental problems in coding theory, including fast or local algorithms for list and unique decoding.

Capacity-achieving locally list-decodable codes. The first such reduction is an application of an expander-based technique of Alon, Edmunds, and Luby [AEL95], which allows us to turn the high-rate locally list-recoverable codes into *capacity achieving* locally list-decodable (or more generally, locally list recoverable) codes. Our main results are stated as Theorems 6.1 and 6.2.

As before, Theorem 6.1 gives capacity achieving locally list decodable codes with query complexity $n^{0.001}$ (say), constant alphabet size and constant output list size, although without an explicit construction or efficient list decoding algorithms. Theorem 6.2 on the other hand gives explicit and efficiently list decodable codes, and a trade-off between query complexity, alphabet size, and output list size. Specifically, these codes obtain query complexity $Q = n^{1/t}$ with an output list size and an alphabet size that grow doubly exponentially with t (and output list size depends additionally on $\log^* n$). In particular, if we choose t to be constant, we obtain query complexity $n^{1/t}$, with constant alphabet size and nearly-constant output list size. We may also choose to take t to be very slowly growing, and this yields query complexity $n^{o(1)}$, with output list and alphabet size $n^{o(1)}$ as well. Prior to this work, no construction of capacity achieving locally list decodable codes with query-complexity $o(n)$ was known.

Near-linear time capacity-achieving list-decodable codes. Given an efficiently list decodable capacity achieving locally list-decodable code (as given by Theorem 6.2 mentioned above), it is straightforward to construct fast algorithms for global list-decoding the same code. Indeed, we just repeat the local decoding algorithm (which can be done in time $n^{O(1/t)}$) a few times, for all n coordinates, and take majority vote at each coordinate. Thus, our previous result implies explicit, capacity-achieving, list-decodable codes (or more generally, list recoverable codes) that can be (globally) list-decoded (or list-recovered) in time $n^{1+O(1/t)}$.

This result is reported in Theorem 7.1. As with the previous point, this result actually allows for a trade-off: we obtain either decoding time $N^{1.001}$ (say) with constant alphabet size and near-constant output list size, or decoding time $n^{1+o(1)}$ at the cost of increasing the alphabet and output list size to $n^{o(1)}$. Previous capacity achieving list-decoding algorithms required at least quadratic time for recovery.

Near-linear time unique decoding up to the Gilbert Varshamov bound. Via a technique of Thommesen [Tho83] and Guruswami and Indyk [GI04], our near-linear time capacity-

achieving list-recoverable codes give a randomized construction of low-rate (up to 0.02) binary codes approaching the *Gilbert-Varshamov* (GV) bound, which admit near-linear time ($n^{1+o(1)}$) algorithms for unique decoding up to half their distance. The formal statement is in Theorem 7.2. Previous constructions which could achieve this either required at least quadratic decoding time, or else did not work for rates larger than 10^{-4}.

Our approach (discussed more below) is modular; given as an ingredient any (globally) high-rate list-recoverable code (with a polynomial time recovery algorithm), it yields high-rate (efficiently) locally list-recoverable code. To achieve the results advertised above, we instantiate this with either a random (non-efficient) linear code or with the (efficient) Algebraic Geometry (AG) subcodes of [GK16b]. Any improvements in these ingredient codes (for example, in the output list size of AG codes, which is near-constant but not quite) would translate immediately into improvements in our constructions.

Organization. In Section 2 below, we discuss related work and put our results in context, followed by an overview of our techniques in Section 3. We set up notation and definitions in Section 4. Our main technical contribution is a construction of high-rate locally list-recoverable codes, and this is handled in Section 5. In Section 6, we discuss the implication to capacity-achieving local list decoding. Finally, Section 7 discusses our applications to near-linear-time decoding algorithms: Section 7.1 presents our results for global capacity-achieving list-decoding, and Section 7.2 presents our results for unique decoding up to the Gilbert-Varshamov bound.

2 Related work

As mentioned above, list decoding and recovery, local decoding, and local list decoding and recovery, have a long and rich history in theoretical computer science. We mention here the results that are most directly related to ours mentioned above.

High-rate local list recovery. Our main technical contribution is the construction of high-rate locally list-recoverable codes. There are two lines of work that are most related to this: the first is on local list recovery, and the second on high-rate (globally) list-recoverable codes.

Local list-decoding (which is a special case of local list recovery) first arose outside of coding theory, motivated by applications in complexity theory. For example, the Goldreich-Levin theorem in cryptography and the Kushilevitz-Mansour algorithm in learning theory are a local list-decoding algorithm for Hadamard codes. Later, Sudan, Trevisan and Vadhan [STV01], motivated by applications in pseudorandomness, gave an algorithm for locally list-decoding Reed-Muller codes. Neither Hadamard codes nor the Reed-Muller codes of [STV01] are high-rate. However, similar ideas can be used to locally list-decode *lifted codes* [GK16a], and *multiplicity codes* [Kop15], which can be seen as high-rate variants of Reed-Muller codes. These algorithms work up to the so-called *Johnson bound.*

Briefly, the Johnson bound says that a code of distance δ is (α, L)-list-decodable, for reasonable L, when $\alpha \leq 1 - \sqrt{1-\delta}$. This allows for high rate list decodable codes when δ is small, but there exist codes which are more list-decodable: the *list-decoding capacity theorem* implies that there are codes of distance δ which are (α, L)-list-decodable for α approaching the distance δ. The "capacity-achieving" list-decodable codes that we have been referring to are those which meet this latter result, which turns out to be optimal.

Like many list-decoding algorithms, the algorithms of [STV01, Kop15, GK16a] can be used for list-recovery as well (indeed, this type of approach was recently used in [GKO$^+$17] to obtain a local list-recovery algorithm for Reed-Muller codes.) However, as mentioned above they only work up to the Johnson bound for list-decoding, and this holds for list-recovery as well. However, for list-recovery, the difference between the Johnson bound and capacity is much more stark. Quantitatively, for (α, ℓ, L)-list-recovery, the Johnson bound requires $\alpha \leq 1 - \sqrt{\ell(1-\delta)}$, which is meaningless unless δ is very large; this requires the rate of the code to be small, less than $1/\ell$. In particular, these approaches do not give high-rate codes for list-recovery, and the Johnson bound appears to be a fundamental bottleneck.

The second line of work relevant to high-rate local list-recovery is that on high-rate *global* list-recovery. Here, there are two main approaches. The first is a line of work on capacity achieving list-decodable codes (also discussed more below). In many cases, the capacity achieving list-decoding algorithms for these codes are also high-rate list-recovery algorithms [GR08b, GW11, Kop15, GX13]. These algorithms are very global: they are all based on finding some interpolating polynomial, and finding this polynomial requires querying almost all of the coordinates. Thus, it is not at all obvious how to tweak these sorts of algorithms to achieve locally list-recoverable codes. The other line of work on high-rate global list-recovery is that of [HW15], which studies high-rate list-recoverable expander codes. While that algorithm is not explicitly local, it's not as clearly global as those previously mentioned (indeed, expander codes are known to have some locality properties [HOW15]). However, that work could only handle list-recovery with no errors—that is, it returns codewords that agree with *all* of the lists S_i, rather than a large fraction of them—and adapting it to handle errors seems like a challenging task.

Thus, even with a great deal of work on locally list recoverable codes, and on high-rate globally list-recoverable codes, it was somehow not clear how to follow those lines of work to obtain high-rate locally list-recoverable codes. Our work, which does give high-rate locally list-recoverable codes, follows a different approach, based on the techniques of [GGR11] for list-decoding tensor codes. In fact, given their ideas and a few other ingredients, our solution is actually quite simple! We discuss our approach in more detail in Section 3.

Capacity achieving locally list decodable codes. As mentioned above, one reason to seek high-rate codes is because of a transformation of Alon, Edmunds, and Luby [AEL95], recently highlighted in [KMRS16], which can, morally speaking, turn any high-rate code with a given property into a capacity achieving code with the same property.[1] This allows us to obtain *capacity achieving* locally list-decodable (or more generally, locally list recoverable) codes. This trick has been used frequently over the years [GI01, GI02, GI03, GI04, HW15, KMRS16, GKO$^+$17], and in particular [GKO$^+$17] used it for local list recovery. We borrow this result from them, and this immediately gives our capacity achieving locally list-decodable codes. Once we have these, they straightforwardly extend to near-linear time capacity-achieving (globally) list-decodable (or more generally, locally list recoverable) codes, simply by repeatedly running the local algorithm on each coordinate.

[1] We note however that this transformation does not apply to the property of list decoding, but just list recovery, and therefore we cannot use existing constructions of high-rate locally list decodable codes [Kop15, GK16a] as a starting point for this transformation.

Capacity-achieving list-decodable codes. We defined list-decodability above as a special case of list-recovery, but it is in fact much older. List-decodability has been studied since the work of Elias and Wozencraft [Eli57, Woz58] in the late 1950s, and the combinatorial limits are well understood. The *list-decoding capacity theorem,* mentioned earlier, states that there exist codes of rate approaching $1 - H_q(\alpha)$ which are (α, L)-list-decodable for small list size L, where $H_q(\alpha)$ is the q-ary entropy function (when q is large we have $1 - H_q(\alpha) \approx 1 - \alpha$). Moreover, any code of rate larger than that must have exponentially large list size.

The existence direction of the list-decoding capacity theorem follows from a random coding argument, and it wasn't until the Folded Reed-Solomon Codes of Guruswami and Rudra [GR08b] that we had explicit constructions of codes which achieved list-decoding capacity. Since then, there have been many more constructions [Gur10, GW11, Kop15, DL12, GX12a, GX13, GK16b], aimed at reducing the alphabet size, reducing the list size, and improving the speed of the recovery algorithm. We show the state-of-the-art in Table 1 below, along with our results (Theorem 7.1).

Code	Reference	Construction	Alphabet size	List size	Decoding time
Folded RS codes, derivative codes	[GR08b, GW11, Kop15]	Explicit	$\text{poly}(n)$	$\text{poly}(n)$	$n^{O(1/\varepsilon)}$
Folded RS subcodes	[DL12]	Explicit	$\text{poly}(n)$	$O(1)$	n^2
(Folded) AG subcodes	[GX12a, GX13]	Monte Carlo	$O(1)$	$O(1)$	n^c
AG subcodes	[GK16b]	Explicit	$O(1)$	$\exp(\exp((\log^* n)^2))$	n^c
Tensor codes	Theorem 7.1	Explicit	$O(1)$	$\exp(\exp(\exp(\log^* n)))$	$n^{1.001}$

Table 1: Constructions of list-decodable codes that enable (α, L) list decoding up to rate $\rho = 1 - H_q(\alpha) - \varepsilon$, for constant ε. We have suppressed the dependence on ε, except where it appears in the exponent on n in the decoding time. Above, c is an unspecified constant. In the analysis of these works, it is required to take $c \geq 3$. It may be that these approaches could be adapted (with faster linear-algebraic methods) to use a smaller constant c, but it is not apparent; in particular we cannot see how to take $c < 2$.

Unique decoding up to the Gilbert-Varshamov bound. The *Gilbert-Varshamov* (GV) bound [Gil52, Var57] is a classical achievability result in coding theory. It states that there exist binary codes of relative distance $\delta \in (0, 1)$ and rate ρ approaching $1 - H_2(\delta)$, where H_2 is the binary entropy function. The proof is probabilistic: for example, it is not hard to see that a random linear code will do the trick. However, finding *explicit* constructions of codes approaching the GV bound remains one of the most famous open problems in coding theory. While we cannot find explicit constructions, we may hope for randomized constructions with efficient algorithms, and indeed this was achieved in the low-rate regime through a few beautiful ideas by Thommesen [Tho83] and follow-up work by Guruswami and Indyk [GI04].

Thommesen gave an efficient randomized construction of *concatenated codes* approaching the GV bound. Starting with a Reed-Solomon code over large alphabet, the construction is to concatenate each symbol with an independent random linear code. Later, [GI04] showed that these codes could in fact be efficiently decoded up to half their distance, in polynomial time, up to rates about 10^{-4}.

Their idea was to use the list recovery properties of Reed-Solomon codes. The algorithm is then to list decode the small inner codes by brute force, and to run the efficient list-recovery algorithm for Reed-Solomon on the output lists of the inner codes: the combinatorial result of Thommesen ensures that the output list will contain a message that corresponds to the transmitted codeword.

In their work, [GI04] used the Guruswami-Sudan list recovery algorithm [GS99]. After decades of work [Ale02, BB10, CH11, BHNW13, CJN+15], this algorithm can now be implemented to run in near-linear time, and so we already can achieve near-linear time unique decoding near the GV bound, up to rates about 10^{-4}. The reason for the bound on the rate is that the Guruswami-Sudan algorithm only works up to the aforementioned Johnson bound, which means it cannot tolerate as much error as capacity-achieving list-recoverable codes. It was noted by Rudra [Rud07] that replacing the Reed-Solomon codes with a capacity achieving list recoverable code (such as folded Reed-Solomon codes) can improve this rate limit up to about 0.02. However, those capacity achieiving list recovery algorithms were slower (as in Table 1), and this increases the running time back to at best quadratic.

The recent work [GKO+17] also applied these techniques to give *locally decodable codes* approaching the Gilbert-Varshamov bound. These have query complexity n^β, and so in particular can be easily adapted to give a global decoding algorithm with running time $O(n^{1+\beta})$. However, the rate up to which the construction works approaches zero exponentially quickly in $1/\beta$.

Using exactly the same approach as these previous works, we may plug in our capacity achieving near-linear-time list-recooverable codes to obtain binary codes approaching the GV bound, which are uniquely decodable up to half their distance in time $n^{1+o(1)}$, and which work with rate matching Rudra's, $\rho = 0.02$.

Remark 2.1. It is natural to ask whether our result can, like [GKO+17], give locally decodable codes on the GV bound of higher rate. In fact, we do not know how to do this. The main catch is that our locality guarantees are for local decoding rather than local correction. That is, we can only recover the message symbols and not the codeword symbols, and consequently we do not know how to choose the message from the output list that corresponds to the unique closest codeword. It is an interesting open question whether one can use our techniques to extend the results of [GKO+17] to higher rates.

List-decodability and local properties of tensor codes. As we elaborate on more below in Section 3, our codes are constructed by taking tensor products of existing constructions of globally list-recoverable codes. Our approach is inspired by that of [GGR11], who study the list-decodability of tensor codes, although they do not address locality. It should be noted that the local *testing* properties of tensor codes have been extensively studied [BS06, Val05, CR05, DSW06, GM12, BV09, BV15, Vid11, Mei09, Vid13]. To the best of our knowledge, ours is the first work to study the local (list) *decodability* of tensor codes, rather than local testability.

3 Overview of techniques

Our main technical contribution is the construction of high-rate locally list-recoverable codes. While these are powerful objects, and result in new sub-linear and near-linear time algorithms for fundamental coding theoretic tasks, our techniques are actually quite simple (at least if we take certain previous works as a black box). We outline our approach below.

Our main ingredient is *tensor codes,* and the analysis given by Gopalan, Guruswami, and Raghevendra in [GGR11]. Given a linear code $C : \mathbb{F}^k \to \mathbb{F}^n$, consider the *tensor code* $C \otimes C : \mathbb{F}^{k \times k} \to \mathbb{F}^{n \times n}$; we will define the tensor product formally in Definition 4.8, but for now, we will treat the codewords of $C \otimes C$ as $n \times n$ matrices with the constraints that the rows and columns are all codewords of the original code C.

In [GGR11], it is shown that the tensor code $C \otimes C$ is roughly as list-decodable as C is. That work was primarily focused on combinatorial results, but their techniques are algorithmic, and it is these algorithmic insights that we leverage here. The algorithm is very simple: briefly, we imagine fixing some small combinatorial rectangle $S \times T \subseteq [n] \times [n]$ of "advice." Think of this advice as choosing the symbols of the codeword indexed by those positions. By alternately list decoding rows and columns, it can be shown that this advice uniquely determines a codeword c of $C \otimes C$. Finally, iterating over all possible pieces of advice yields the final list.

Inspired by their approach, our Main Technical Lemma 5.2 says that if C is list-recoverable, then not only $C \otimes C$ is also list-recoverable, but in fact it is (approximately) locally list-recoverable. To understand the intuition, let us describe the algorithm just for $C \otimes C$, although our actual codes will require a higher tensor power $C^{\otimes t}$. Suppose that C is list-recoverable with output list size L. First, imagine fixing some advice $J := (j_1, \ldots, j_m) \in [L]^m$ for some (small integer) parameter m. This advice will determine an algorithm $\widetilde{A}_J$ which attempts to locally decode some message that corresponds to a close-by codeword c of $C \otimes C$, and the list we finally return will be the list of all algorithms $\widetilde{A}_J$ obtained by iterating over all possible advice.

Now, we describe the randomized algorithm $\widetilde{A}_J$, on input $(i, i') \in [n] \times [n]$.[2] Recall, $\widetilde{A}_J$ is allowed to query the input lists at every coordinate, and must produce a guess for the codeword value indexed by (i, i'). First, $\widetilde{A}_J$ chooses m random rows of $[n] \times [n]$. These each correspond to codewords in C, and $\widetilde{A}_J$ runs C's list-recovery algorithm on them to obtain lists $\mathcal{L}_1, \ldots, \mathcal{L}_m$ of size at most L each. Notice that this requires querying mn coordinates, which is roughly the square root of the length of the code (which is n^2). Then, $\widetilde{A}_J$ will use the advice $j_1, \ldots, j_m$ to choose codewords from each of these lists, and we remember the i''th symbol of each of these codewords. Finally, $\widetilde{A}_J$ again runs C's list-recovery algorithm on the i''th column, to obtain another list $\mathcal{L}$. Notice that our advice now has the same form as it does in [GGR11]: we have chosen a few symbols of a codeword of C. Now $\widetilde{A}_J$ chooses the codeword in $\mathcal{L}$ that agrees the most with this advice. The i''th symbol of this codeword is $\widetilde{A}_J$'s guess for the (i, i') symbol of the tensor codeword.

The above idea gives a code of length n which is locally list-recoverable with query complexity on the order of $\sqrt{n}$. This algorithm for $C \otimes C$ extends straightforwardly to $C^{\otimes t}$, with query complexity $n^{1/t}$. The trade-off is that the output list-size also grows with t. Thus, as we continue to take tensor powers, the locality improves, while the output list-size degrades; this allows for the trade-off between locality and output list-size mentioned in the introduction.

One issue with this approach is that this algorithm may in fact fail on a constant fraction of coordinates (i, i') (e.g., when a whole column is corrupted). To get around this, we first encode our message with a high-rate *locally decodable code,* before encoding it with the tensor code. For this, we use the codes of [KMRS16], which have rate that is arbitrarily close to 1, and which are locally decodable with $\exp(\sqrt{\log n})$ queries. This way, instead of directly querying the tensor code (which may give the wrong answer a constant fraction of the time), we instead use the outer locally decodable code to query the tensor code: this still does not use too many queries, but now it is

[2]The algorithm $\widetilde{A}_J$ we describe decodes codeword bits instead of messages bits, but since the codes we use are systematic this algorithm can also decode message bits.

robust to a few errors.

The final question is what to use as an inner code. Because we are after high-rate codes we require C to be high-rate (globally) list recoverable. Moreover, since the tensor operation inflates the output list size by quite a lot, we require C to have small (constant or very slowly growing) output list size. Finally, we need C to be linear to get a handle on the rate of the tensor product. One possible choice is random linear codes, and these give a non-explicit and non-efficient construction with constant output list size. Another possible choice is the Algebraic Geometry subcodes of [GX13, GK16b] which give explicit and efficient construction but with slowly growing output list size. However, we cannot quite use these latter codes as a black box, for two reasons. First, the analysis in [GX13] only establishes list-*decodability*, rather than list-recoverability. Fortunately, list-recoverability follows from exactly the same argument as list-decodability. Second, these codes are linear over a subfield, but are not themselves linear, while our arguments require linearity over the whole alphabet. Fortunately, we can achieve the appropriate linearity by concatenating the AG subcode with a small list-recoverable linear code, which exists by a probabilistic argument. We handle these modifications to the approach of [GX13, GK16b] in Appendix A.

To summarize, our high-rate locally list-recoverable code is given by these ingredients: to encode a message x, we first encode it with the [KMRS16] locally decodable code. Then we encode this with a t-fold tensor product of a random linear code or a modified AG subcode and we are done. We go through the details of the argument sketched above in Section 5; but first, we introduce some notation and formally define the notions that we will require.

4 Definitions and preliminaries

For a prime power q we denote by $\mathbb{F}_q$ the finite field of q elements. For any finite alphabet Σ and for any pair of strings $x, y \in \Sigma^n$, the relative distance between x and y is the fraction of coordinates $i \in [n]$ on which x and y differ, and is denoted by $\mathrm{dist}(x, y) := |\{i \in [n] : x_i \neq y_i\}| / n$. For a positive integer ℓ we denote by $\binom{\Sigma}{\ell}$ the set containing all subsets of Σ of size ℓ, and for any pair of strings $x \in \Sigma^n$ and $S \in \binom{\Sigma}{\ell}^n$ we denote by $\mathrm{dist}(x, S)$ the fraction of coordinates $i \in [n]$ for which $x_i \notin S_i$, that is, $\mathrm{dist}(x, S) := |\{i \in [n] : x_i \notin S_i\}| / n$. Throughout the paper, we use $\exp(n)$ to denote $2^{\Theta(n)}$. Whenever we use log, it is to the base 2.

4.1 Error-correcting codes

Let Σ be an alphabet and k, n be positive integers (the message length and the block length, respectively). A code is an injective map $C : \Sigma^k \to \Sigma^n$. The elements in the domain of C are called messages and the elements in the image of C are called codewords. If $\mathbb{F}$ is a finite field and Σ is a vector space over $\mathbb{F}$, we say that C is $\mathbb{F}$-linear if it is a linear transformation over $\mathbb{F}$ between the $\mathbb{F}$-vector spaces Σ^k and Σ^n. If $\Sigma = \mathbb{F}$ and C is $\mathbb{F}$-linear, we simply say that C is linear. The generating matrix of a linear code $C : \mathbb{F}^k \to \mathbb{F}^n$ is the matrix $G \in \mathbb{F}^{n \times k}$ such that $C(x) = G \cdot x$ for any $x \in \mathbb{F}^k$. We say that a code $C : \Sigma^k \to \Sigma^n$ is systematic if any message is the prefix of its image, that is, for any $x \in \Sigma^k$ there exists $y \in \Sigma^{n-k}$ such that $C(x) = (x, y)$.

The rate of a code $C : \Sigma^k \to \Sigma^n$ is the ratio $\rho := \frac{k}{n}$. The relative distance $\mathrm{dist}(C)$ of C is the minimum $\delta > 0$ such that for every pair of distinct messages $x, y \in \Sigma^k$ it holds that $\mathrm{dist}(C(x), C(y)) \geq \delta$. For a code $C : \Sigma^k \to \Sigma^n$ of relative distance δ, a given parameter $\alpha < \delta/2$, and a string $w \in \Sigma^n$, the problem of decoding from α fraction of errors is the task of finding the unique message $x \in \Sigma^k$

(if any) which satisfies $\mathrm{dist}(C(x), w) \leq \alpha$.

The best known general trade-off between rate and distance of codes is the Gilbert-Varshamov bound, attained by random (linear) codes. For $x \in [0,1]$ let

$$H_q(x) = x \log_q(q-1) + x \log_q(1/x) + (1-x) \log_q(1/(1-x))$$

denote the q-ary entropy function.

Theorem 4.1 (Gilbert-Varshamov (GV) bound, [Gil52, Var57])**.** *For any prime power q, $0 \leq \delta < 1 - \frac{1}{q}$, $0 \leq \rho < 1 - H_q(\delta)$, and sufficiently large n, a random linear code $C : \mathbb{F}_q^{\rho n} \to \mathbb{F}_q^n$ of rate ρ has relative distance at least δ with probability at least $1 - \exp(-n)$.*

4.2 List decodable and list recoverable codes

List decoding is a paradigm that allows one to correct more than $\delta/2$ fraction of errors by returning a small list of messages that correspond to close-by codewords. More formally, for $\alpha \in [0,1]$ and an integer L we say that a code $C : \Sigma^k \to \Sigma^n$ is (α, L)-list decodable if for any $w \in \Sigma^n$ there are at most L different messages $x \in \Sigma^k$ which satisfy that $\mathrm{dist}(C(x), w) \leq \alpha$.

For list decoding concatenated codes it is useful to consider the notion of list recovery where one is given as input a small list of candidate symbols for each of the codeword coordinates, and is required to output a list of messages such that the corresponding codewords are consistent with the input lists. More concretely, for $\alpha \in [0,1]$ and integers ℓ, L we say that a code $C : \Sigma^k \to \Sigma^n$ is (α, ℓ, L)-list recoverable if for any $S \in \binom{\Sigma}{\ell}^n$ there are at most L different messages $x \in \Sigma^k$ which satisfy that $\mathrm{dist}(C(x), S) \leq \alpha$.

It is well-known that $1 - H_q(\alpha)$ is the list decoding capacity, that is, any q-ary code of rate above $1 - H_q(\alpha)$ cannot be list decoded from α fraction of errors with list size polynomial in the block length, and on the other hand, a random q-ary (linear) code of rate below $1 - H_q(\alpha)$ can be list decoded from α fraction of errors with small list size.

Theorem 4.2 ([Gur01], Theorem 5.3)**.** *For any prime power q, $0 \leq \alpha < 1 - \frac{1}{q}$,*

$$0 \leq \rho < 1 - H_q(\alpha) - \frac{1}{\log_q(L+1)},$$

and sufficiently large n, a random linear code $C : \mathbb{F}_q^{\rho n} \to \mathbb{F}_q^n$ of rate ρ is (α, L)-list decodable with probability at least $1 - \exp(-n)$.

The following is a generalization of the above theorem to the setting of list recovery.

Theorem 4.3 ([Gur01], Lemma 9.6)**.** *For any prime power q, integers $1 \leq \ell \leq q$ and $L > \ell$, $0 \leq \alpha \leq 1$,*

$$0 \leq \rho < \frac{1}{\log q} \cdot \left[(1-\alpha) \cdot \log(q/\ell) - H_2(\alpha) - H_2(\ell/q) \cdot \frac{q}{\log_q(L+1)} \right],$$

and sufficiently large n, a random linear code $C : \mathbb{F}_q^{\rho n} \to \mathbb{F}_q^n$ of rate ρ is (α, ℓ, L)-list recoverable with probability at least $1 - \exp(-n)$.

Over large alphabet the above theorem yields the following.

Corollary 4.4. *There is a constant c so that the following holds. Choose $\rho \in [0,1]$, $\varepsilon > 0$, and a positive integer ℓ. Suppose that q is a prime power which satisfies*

$$q \geq \max\{(1-\rho-\varepsilon)^{-c(1-\rho-\varepsilon)/\varepsilon}, (\rho+\varepsilon)^{-c(\rho+\varepsilon)/\varepsilon}, \ell^{c/\varepsilon}\}.$$

Then for sufficiently large n, a random linear code $C : \mathbb{F}_q^{\rho n} \to \mathbb{F}_q^n$ of rate ρ is $(1-\rho-\varepsilon, \ell, q^{c\ell/\varepsilon})$-list recoverable with probability at least $1 - \exp(-n)$.

Proof. Follows by observing that in the above setting of parameters,

$$\begin{aligned} & \frac{1}{\log q} \cdot \left[(\rho+\varepsilon) \cdot \log(q/\ell) - H_2(1-\rho-\varepsilon) - H_2(\ell/q) \cdot \frac{q}{c\ell/\varepsilon} \right] \\ \geq \;& \rho + \varepsilon - \frac{\log \ell}{\log q} - \frac{(1-\rho-\varepsilon)\log(1/(1-\rho-\varepsilon))}{\log q} - \frac{(\rho+\varepsilon)\log(1/(\rho+\varepsilon))}{\log q} - O(\varepsilon/c) \\ \geq \;& \rho + \varepsilon - O(\varepsilon/c), \end{aligned}$$

so the corollary holds for sufficiently large constant c. □

4.3 Locally decodable codes

Intuitively, a code C is said to be locally decodable if, given a codeword $C(x)$ that has been corrupted by some errors, it is possible to decode any coordinate of the corresponding message x by reading only a small part of the corrupted version of $C(x)$. Formally, it is defined as follows.

Definition 4.5 (Locally decodable code (LDC))**.** We say that a code $C : \Sigma^k \to \Sigma^n$ is (Q, α)-locally decodable if there exists a randomized algorithm A that satisfies the following requirements:

- **Input:** A takes as input a coordinate $i \in [k]$, and also gets oracle access to a string $w \in \Sigma^n$ that is α-close to some codeword $C(x)$.
- **Query complexity:** A makes at most Q queries to the oracle w.
- **Output:** A outputs x_i with probability at least $\frac{2}{3}$.

Remark 4.6. By definition it holds that $\alpha < \mathrm{dist}(C)/2$. The above success probability of $\frac{2}{3}$ can be amplified using sequential repetition, at the cost of increasing the query complexity. Specifically, amplifying the success probability to $1 - e^{-t}$ requires increasing the query complexity by a multiplicative factor of $O(t)$.

Locally list decodable and list recoverable codes. The following definition generalizes the notion of locally decodable codes to the setting of list decoding / recovery. In this setting the algorithm A is required to find all messages that correspond to nearby codewords in an implicit sense.

Definition 4.7 (Locally list recoverable code)**.** We say that a code $C : \Sigma^k \to \Sigma^n$ is (Q, α, ℓ, L)-locally list recoverable if there exists a randomized algorithm A that satisfies the following requirements:

- **Preprocessing:** A outputs L randomized algorithms $A_1, \ldots, A_L$.

- **Input:** Each A_j takes as input a coordinate $i \in [k]$, and also gets oracle access to a string $S \in \binom{\Sigma}{\ell}^n$.
- **Query complexity:** Each A_j makes at most Q queries to the oracle S.
- **Output:** For every codeword $C(x)$ that is α-close to S, with probability at least $\frac{2}{3}$ over the randomness of A the following event happens: there exists some $j \in [L]$ such that for all $i \in [k]$,
$$\Pr[A_j(i) = x_i] \geq \frac{2}{3},$$
where the probability is over the internal randomness of A_j.

We say that A has preprocessing time T_{pre} if A outputs the description of the algorithms $A_1, \ldots, A_L$ in time at most T_{pre}, and has running time T if each A_j has running time at most T. Finally, we say that C is (Q, α, L)-locally list decodable if it is $(Q, \alpha, 1, L)$-locally list recoverable.

4.4 Tensor codes

A main ingredient in our constructions is the tensor product operation, defined as follows.

Definition 4.8 (Tensor codes)**.** Let $C_1 : \mathbb{F}^{k_1} \to \mathbb{F}^{n_1}$, $C_2 : \mathbb{F}^{k_2} \to \mathbb{F}^{n_2}$ be linear codes, and let $G_1 \in \mathbb{F}^{n_1 \times k_1}, G_2 \in \mathbb{F}^{n_2 \times k_2}$ be the generating matrices of C_1, C_2 respectively. Then the tensor code $C_1 \otimes C_2 : \mathbb{F}^{k_1 \times k_2} \to \mathbb{F}^{n_1 \times n_2}$ is defined as $(C_1 \otimes C_2)(M) = G_1 \cdot M \cdot G_2^T$.

Note that the codewords of $C_1 \otimes C_2$ are $n_1 \times n_2$ matrices over $\mathbb{F}$ whose columns belong to the code C_1 and whose rows belong to the code C_2. The following effects of the tensor product operation on the classical parameters of the code are well known (see e.g. [Sud01, DSW06]).

Fact 4.9. *Suppose that $C_1 : \mathbb{F}^{k_1} \to \mathbb{F}^{n_1}$, $C_2 : \mathbb{F}^{k_2} \to \mathbb{F}^{n_2}$ are linear codes of rates ρ_1, ρ_2 and relative distances δ_1, δ_2 respectively. Then the tensor code $C_1 \otimes C_2$ has rate $\rho_1 \cdot \rho_2$ and relative distance $\delta_1 \cdot \delta_2$.*

For a linear code C, let $C^{\otimes 1} := C$ and $C^{\otimes t} := C \otimes C^{\otimes (t-1)}$. Then by the above, if C has rate ρ and relative distance δ then $C^{\otimes t}$ has rate ρ^t and relative distance δ^t.

5 High-rate locally list recoverable codes

We start by showing the existence of high-rate locally list recoverable codes. For this we first show in Section 5.1 below that high-rate tensor codes are *approximately locally list recoverable*, namely there exists a short list of local algorithms that can recover *most* of the coordinates of messages that correspond to near-by codewords. We then observe in Section 5.2 that by pre-encoding the message with a locally decodable code, the former codes can be turned into *locally list recoverable codes* for which the local algorithms can recover *all* the coordinates of messages that correspond to near-by codewords. Finally, we show in Section 5.3 how to instantiate the codes used in the process in order to obtain high-rate locally list recoverable codes with good performance.

5.1 Approximate local list recovery

We start by showing that high-rate tensor codes are approximately locally list recoverable as per Definition 5.1 below. As noted above, the main difference between approximately locally list recoverable codes and locally list recoverable codes (Definition 4.7) is that in the former we only require that the local algorithms recover *most* of the coordinates. A simple averaging argument then shows that in the case of approximate local list recovery each of the local algorithms in the output list can be assumed to be *deterministic*. Finally, to describe our approximate local list recovery algorithm it will be more convenient to require that the local algorithms recover *codeword bits* as opposed to *message bits*[3].

Definition 5.1 (Approximately locally list recoverable code)**.** We say that a code $C : \Sigma^k \to \Sigma^n$ is $(Q, \alpha, \varepsilon, \ell, L)$-approximately locally list recoverable if there exists a randomized algorithm A that satisfies the following requirements:

- **Preprocessing:** A outputs L deterministic algorithms $A_1, \ldots, A_L$.
- **Input:** Each A_j takes as input a coordinate $i \in [n]$, and also gets oracle access to a string $S \in \binom{\Sigma}{\ell}^n$.
- **Query complexity:** Each A_j makes at most Q queries to the oracle S.
- **Output:** For every codeword $C(x)$ that is α-close to S, with probability at least $1 - \varepsilon$ over the randomness of A the following event happens: there exists some $j \in [L]$ such that

$$\Pr_{i \in [n]} [A_j(i) = C(x)_i] \geq 1 - \varepsilon,$$

  where the probability is over the choice of uniform random $i \in [n]$.

As before, we say that A has preprocessing time T_{pre} if A outputs the description of the algorithms $A_1, \ldots, A_L$ in time at most T_{pre}, and has running time T if each A_j has running time at most T.

Our main technical lemma is the following.

Lemma 5.2 (Main technical)**.** *Suppose that $C : \mathbb{F}^k \to \mathbb{F}^n$ is a linear code of relative distance δ that is (α, ℓ, L)-(globally) list recoverable. Then for any $\widetilde{\varepsilon} > 0$, the tensor product code $\widetilde{C} := C^{\otimes t} : \mathbb{F}^{k^t} \to \mathbb{F}^{n^t}$ is $(\widetilde{Q}, \widetilde{\alpha}, \widetilde{\varepsilon}, \ell, \widetilde{L})$-approximately locally list recoverable for $\widetilde{\alpha} = \alpha \cdot \widetilde{\varepsilon} \cdot \delta^{O(t)}$,*

$$\widetilde{Q} = n \cdot \frac{\log^t L}{(\alpha \cdot \widetilde{\varepsilon})^{O(t)} \cdot \delta^{O(t^2)}},$$

and

$$\widetilde{L} = \exp\left(\frac{\log^t L}{(\alpha \cdot \widetilde{\varepsilon})^{O(t)} \cdot \delta^{O(t^2)}}\right).$$

Moreover, the approximate local list recovery algorithm for $\widetilde{C}$ has preprocessing time

$$\widetilde{T}_{pre} = \log n \cdot \exp\left(\frac{\log^t L}{(\alpha \cdot \widetilde{\varepsilon})^{O(t)} \cdot \delta^{O(t^2)}}\right),$$

[3]In our constructions we shall use systematic codes and so recovery of codeword bits will imply also recovery of message bits.

and if the (global) list recovery algorithm for C runs in time T then the approximate local list recovery algorithm for $\widetilde{C}$ runs in time

$$\widetilde{T} = T \cdot \frac{\log^t L}{(\alpha \cdot \widetilde{\varepsilon})^{O(t)} \cdot \delta^{O(t^2)}}.$$

The proof of the above lemma will follow from repeated application of the following technical lemma.

Lemma 5.3. *Suppose that $C : \mathbb{F}^k \to \mathbb{F}^n$ is a linear code of relative distance δ that is (α, ℓ, L)-(globally) list recoverable, and $C' : \mathbb{F}^{k'} \to \mathbb{F}^{n'}$ is a linear code that is $(Q', \alpha', \varepsilon', \ell, L')$-approximately locally list recoverable. Then for any $\widetilde{\varepsilon} \geq 100\varepsilon'/\delta$, the tensor product code $\widetilde{C} := C \otimes C' : \mathbb{F}^{k \times k'} \to \mathbb{F}^{n \times n'}$ is $(\widetilde{Q}, \widetilde{\alpha}, \widetilde{\varepsilon}, \ell, \widetilde{L})$-approximately locally list recoverable for $\widetilde{\alpha} = \frac{1}{10} \cdot \min\{\alpha' \cdot \delta, \alpha \cdot \widetilde{\varepsilon}\}$,*

$$\widetilde{Q} = O\left(\frac{\log(L/\widetilde{\varepsilon})}{(\delta \cdot \alpha' \cdot \widetilde{\varepsilon})^2}\right) \cdot Q' + n,$$

and

$$\widetilde{L} = \exp\left(\frac{\log L' \cdot \log(L/\widetilde{\varepsilon})}{(\delta \cdot \alpha' \cdot \widetilde{\varepsilon})^2}\right).$$

Moreover, if the (global) list recovery algorithm for C runs in time T, and the approximate local list recovery algorithm for C' has preprocessing time T'_{pre} and runs in time T', then the approximate local list recovery algorithm for $\widetilde{C}$ has preprocessing time

$$\widetilde{T}_{pre} = O\left(\frac{\log(L/\widetilde{\varepsilon})}{(\delta \cdot \alpha' \cdot \widetilde{\varepsilon})^2}\right) \cdot (\log n + T'_{pre}) + \exp\left(\frac{\log L' \cdot \log(L/\widetilde{\varepsilon})}{(\delta \cdot \alpha' \cdot \widetilde{\varepsilon})^2}\right),$$

and runs in time

$$\widetilde{T} = O\left(\frac{\log(L/\widetilde{\varepsilon})}{(\delta \cdot \alpha' \cdot \widetilde{\varepsilon})^2}\right) \cdot T' + T.$$

Before we prove the above lemma we show how it implies the Main Technical Lemma 5.2.

Proof of Lemma 5.2. The proof proceeds by repeated application of Lemma 5.3 to the code C. For a fixed $\widetilde{\varepsilon} > 0$, our goal is to find parameters $Q_{(t)}$, $\alpha_{(t)}$, $L_{(t)}$ such that $C^{\otimes t}$ is a $(Q_{(t)}, \alpha_{(t)}, \widetilde{\varepsilon}, \ell, L_{(t)})$-approximately locally list recoverable code.

We begin by defining

$$\varepsilon_{(i)} := \left(\frac{\delta}{100}\right)^{t-i} \widetilde{\varepsilon} \qquad \text{for } i = 1, \ldots, t\ ,$$

where the factor $\frac{\delta}{100}$ comes from Lemma 5.3. With this definition, we have $\widetilde{\varepsilon} = \varepsilon_{(t)} > \varepsilon_{(t-1)} > \cdots > \varepsilon_{(1)}$, and

$$\widetilde{\varepsilon} = \varepsilon_{(t)} = \left(\frac{100}{\delta}\right)^{t-1} \varepsilon_{(1)}.$$

When $t = 1$, we have $C^{\otimes t} = C$, which means $Q_{(1)} = n$, $\alpha_{(1)} = \alpha$, $L_{(1)} = L$, and this holds for any $\varepsilon_{(1)} > 0$, and hence for any $\widetilde{\varepsilon} > 0$, since C is actually list recoverable (not just approximately).

Since $C^{\otimes i} = C \otimes C^{\otimes(i-1)}$, by Lemma 5.3, we have the following recursive relationships

$$
\begin{aligned}
\alpha_{(i)} &= \frac{1}{10}\min(\alpha_{(i-1)}\delta, \alpha\varepsilon_{(i)}) \\
m_{(i)} &:= \frac{\log(L/\varepsilon_{(i)})}{\left(\delta\alpha_{(i-1)}\varepsilon_{(i)}\right)^2} \\
Q_{(i)} &= m_{(i)}Q_{(i-1)} + n \\
L_{(i)} &= L_{(i-1)}^{m_{(i)}} \\
T_{(i)} &= m_{(i)} \cdot T_{(i-1)} + T \\
(T_{\mathrm{pre}})_{(i)} &= m_{(i)} \cdot \left(\log n + (T_{\mathrm{pre}})_{(i-1)}\right) + L_{(i-1)}^{m_{(i)}}
\end{aligned}
$$

Solving these recursions gives the following bounds on the parameters of interest. The distance parameter, $\alpha_{(t)}$, satisfies

$$
\begin{aligned}
\alpha_{(t)} &= \frac{1}{10}\min(\alpha_{(t-1)}\delta, \alpha\varepsilon_{(t)}) \\
&\geq \frac{1}{10}\min(\alpha_{(t-1)}\delta, \alpha\varepsilon_{(1)}) \\
&\geq \left(\frac{\delta}{10}\right)^{t-1}\alpha\varepsilon_{(1)} \\
&= \left(\frac{\delta}{10}\right)^{t-1}\alpha\left(\frac{\delta}{100}\right)^{t-1}\widetilde{\varepsilon} \\
&= \left(\frac{\delta^2}{1000}\right)^{t-1}\alpha\widetilde{\varepsilon}.
\end{aligned}
$$

Notice that as i increases, the parameters $\varepsilon_{(i)}$ are increasing, whereas the parameters $\alpha_{(i)}$ are decreasing. Nevertheless, $\alpha_{(i)}\varepsilon_{(i)} > \alpha_{(i-1)}\varepsilon_{(i-1)}$. To see this note that $\alpha_{(i)} < \frac{1}{10}\min(\alpha_{(i-1)}, \varepsilon_{(i)}) < \varepsilon_{(i)}$ for all i, which means

$$
\frac{100}{\delta}\alpha_{(i)} = \frac{10}{\delta}\min(\alpha_{(i-1)}\delta, \varepsilon_{(i)}) > \alpha_{(i-1)},
$$

where the last inequality follows since $\alpha_{i-1} < \varepsilon_{i-1} < \varepsilon_i$. Multiplying both sides by $\varepsilon_{(i)}$, we conclude that $\alpha_{(i)}\varepsilon_{(i+1)} > \alpha_{(i-1)}\varepsilon_{(i)}$, as desired.

Now, notice that since $\varepsilon_{(i-1)} < \varepsilon_{(i)}$ and $\alpha_{(i)}\varepsilon_{(i+1)} > \alpha_{(i-1)}\varepsilon_{(i)}$, we have

$$
m_{(i-1)} = \frac{\log(L/\varepsilon_{(i-1)})}{\left(\delta\alpha_{(i-2)}\varepsilon_{(i-1)}\right)^2} \geq \frac{\log(L/\varepsilon_{(i)})}{\left(\delta\alpha_{(i-1)}\varepsilon_{(i)}\right)^2} = m_{(i)}
$$

Thus

$$
m_{(2)} \geq m_{(3)} \geq \cdots \geq m_{(t)}
$$

Notice that

$$
m_{(2)} = \frac{\log(L/\varepsilon_{(2)})}{\left(\delta\alpha_{(1)}\varepsilon_{(2)}\right)^2} = \frac{\log\left(\left(\frac{100}{\delta}\right)^{t-2}L/\varepsilon_{(t)}\right)}{\left(\delta\alpha\left(\frac{\delta}{100}\right)^{t-2}\varepsilon_{(t)}\right)^2} \leq \frac{\log\left(\left(\frac{100}{\delta}\right)^{t-2}L/\varepsilon_{(t)}\right)}{\left(\frac{\delta}{100}\right)^{2t}\alpha^2\varepsilon_{(t)}^2} \leq \frac{\log(L/\varepsilon_{(t)})}{\left(\frac{\delta}{100}\right)^{3t}\alpha^2\varepsilon_{(t)}^2}
$$

Where last inequality holds since $\log(ax) \le a\log(x)$ when $\log(x) \ge \log(a)/(a-1)$.

The output list size, $L_{(t)}$, satisfies

$$\begin{aligned} L_{(t)} &= L^{\prod_{i=2}^{t-1} m_{(i)}} \\ &\le L^{m_{(2)}^{t-1}} \\ &= \exp\left((\log L) \cdot m_{(2)}^{t-1}\right) \\ &\le \exp\left(\frac{\log^t(L/\widetilde{\varepsilon})}{\left(\frac{\delta}{100}\right)^{3t^2} \alpha^{2t}\widetilde{\varepsilon}^{2t}}\right). \end{aligned}$$

The query complexity, $Q_{(t)}$, satisfies

$$\begin{aligned} Q_{(t)} &= n\left(1 + \sum_{i=2}^{t-1}\prod_{j=i}^{t-1} m_{(j)}\right) \\ &\le (t-1)nm_{(2)}^{t-1} \\ &\le n\left(\frac{\log^t(L/\widetilde{\varepsilon})}{\left(\frac{\delta}{100}\right)^{3t^2} \alpha^{2t}\widetilde{\varepsilon}^{2t}}\right). \end{aligned}$$

The running time, $T_{(t)}$, satisfies

$$\begin{aligned} T_{(t)} &= T\prod_{i=2}^{t-1} m_{(i)} + T\left(1 + \sum_{i=2}^{t-1}\prod_{j=i}^{t-1} m_{(j)}\right) \\ &\le T\left(\frac{\log^t(L/\widetilde{\varepsilon})}{\left(\frac{\delta}{100}\right)^{3t^2} \alpha^{2t}\widetilde{\varepsilon}^{2t}}\right). \end{aligned}$$

The pre-processing time, $(T_{\text{pre}})_{(t)}$, satisfies

$$\begin{aligned} (T_{\text{pre}})_{(t)} &= m_{(t)} \cdot \left(\log n + (T_{\text{pre}})_{(t-1)}\right) + \exp\left(m_{(t)} \log L_{(t-1)}\right) \\ &= \log n \cdot \left(1 + \sum_{i=2}^{t}\prod_{j=i}^{t-1} m_{(j)}\right) + \sum_{i=1}^{t-1} L_{(i)}\prod_{j=i}^{t-1} m_{(j)} \\ &\le m_{(2)}^{t-1} t\log n + t m_{(2)}^{t-1} L_{(t)} \\ &\le (t\log n + tL_{(t)}) m_{(2)}^{t-1} \\ &\le \left(t\log n + t\exp\left(\frac{\log^t(L/\widetilde{\varepsilon})}{\left(\frac{\delta}{100}\right)^{3t^2} \alpha^{2t}\widetilde{\varepsilon}^{2t}}\right)\right) \frac{\log^t(L/\widetilde{\varepsilon})}{\left(\frac{\delta}{100}\right)^{3t^2} \alpha^{2t}\widetilde{\varepsilon}^{2t}} \\ &\le \log n \cdot \exp\left(\frac{\log^t L}{(\alpha\cdot\widetilde{\varepsilon})^{O(t)} \cdot \delta^{O(t^2)}}\right) \end{aligned}$$

□

We proceed to the proof of Lemma 5.3.

Proof of Lemma 5.3. Our goal is to find a randomized algorithm $\widetilde{A}$ that outputs a list of (deterministic) local algorithms $\widetilde{A}_1, \ldots, \widetilde{A}_{\widetilde{L}}$ such that for any codeword $\widetilde{c} \in C \otimes C'$ that is consistent with most of the input lists, with high probability over the randomness of A, there exists some $\widetilde{A}_i$ in the output list that computes correctly most of the coordinates of $\widetilde{c}$.

We first describe the algorithm $\widetilde{A}$. The algorithm $\widetilde{A}$ first chooses a uniform random subset $R \subseteq [n]$ of rows of size $m := O\left(\frac{\log(L/\widetilde{\varepsilon})}{(\delta \cdot \alpha' \cdot \widetilde{\varepsilon})^2}\right)$. It then runs for each row $r \in R$, independently, the approximate local list recovery algorithm A' for C', let $A'^r_1, \ldots, A'^r_{L'}$ denote the output algorithms on row r. Finally, for every possible choice of a single local algorithm $A'^r_{j_r}$ per each of the rows $r \in R$, the algorithm $\widetilde{A}$ outputs a local algorithm denoted $\widetilde{A}_J$ where $J := (j_r)_{r \in R} \in [L']^R$. The formal definition of the algorithm $\widetilde{A}_J$ is given below, followed by an informal description.

Algorithm 1 The approximate local list recovery algorithm for $C \otimes C'$.

function $\widetilde{A}_J((i, i') \in [n] \times [n'])$
- ▷ $\widetilde{A}_J$ receives oracle access to a matrix of lists $S \in \binom{\mathbb{F}}{\ell}^{n \times n'}$
- ▷ $J = (j_r)_{r \in R} \in [L']^R$
- **for** $r \in R$ **do**
  - Run $A'^r_{j_r}$ on input i' and oracle access to the rth row $S|_{\{r\} \times [n']}$.
  - Let $c'_r \leftarrow A'^r_{j_r}(i')$.
  - ▷ c'_r is a candidate for the symbol at position $(r, i') \in [n] \times [n']$.
- **end for**
- ▷ At this point, we have candidate symbols for every position in $R \times \{i'\}$.
- Run the (global) list recovery algorithm for C on the i'th column $S|_{[n] \times \{i'\}}$.
- Let $\mathcal{L} \subseteq \mathbb{F}^n$ denote the output list.
- Choose a codeword $c \in \mathcal{L}$ such that $c|_R$ is closest to $(c'_r)_{r \in R}$ (breaking ties arbitrarily).
- **Return:** c_i

end function

Recall, that the algorithm $\widetilde{A}_J$ is given as input a codeword coordinate $(i, i') \in [n] \times [n']$ in the tensor product code $C \otimes C'$, is allowed to query the input lists at every coordinate, and must produce a guess for the codeword value indexed by (i, i'). To this end, the algorithm $\widetilde{A}_J$ first runs on each row $r \in R$ the local recovery algorithm $A'^r_{j_r}$ for C' that is specified by the choice of $J = (j_r)_{r \in R}$ on input i' and oracle access to the rth row. For each row $r \in R$ let c'_r be the guess for the symbol at position $(r, i') \in [n] \times [n']$ produced by $A'^r_{j_r}$. At this point we have candidate symbols for all positions in $R \times \{i'\}$. Now, $\widetilde{A}_J$ runs the global list recovery algorithm for C on the i'th column and chooses a codeword $c \in C$ from the output list that agrees the most with the candidate symbols $(c'_r)_{r \in R}$ on this column. Finally, the ith symbol of c is $\widetilde{A}_J$'s guess for the (i, i') symbol of the tensor codeword. Next we prove the correctness of the purposed local list recovery algorithm, followed by an analysis of its performance.

Correctness: Let $\widetilde{c}$ be a codeword of $\widetilde{C} = C \otimes C'$ such that $\mathrm{dist}(\widetilde{c}, S) \leq \widetilde{\alpha}$. Our goal is to show that with probability at least $1 - \widetilde{\varepsilon}$ over the randomness of $\widetilde{A}$, there exists some local algorithm $\widetilde{A}_J$ in the output list of $\widetilde{A}$ that computes correctly at least $1 - \widetilde{\varepsilon}$ fraction of the coordinates of $\widetilde{c}$.

We first explain how the algorithm $\widetilde{A}_J$ above is obtained. Recall that $A^r_{j_r}$ is the local algorithm that computes the j_rth "guess" for the codeword on the rth row. For every row $r \in R$ let $j_r \in [L']$ be such that $A^r_{j_r}$ agrees the most with $\widetilde{c}$ on the rth row among all local algorithms $A^r_1, \ldots, A^r_{L'}$ (breaking ties arbitrarily), and let $J = (j_r)_{r \in R}$. We shall show below that with probability at least $1 - \widetilde{\varepsilon}$ over the randomness of $\widetilde{A}$, it holds that $\widetilde{A}_J$ computes correctly at least $1 - \widetilde{\varepsilon}$ fraction of the coordinates of $\widetilde{c}$.

The high level idea of the proof is as follows. First, we observe that since the rows in R are chosen uniformly at random, and by averaging, with high probability for most rows $r \in R$ it holds that $\widetilde{c}$ is consistent with most of the input lists on the row. Let us denote by $R_{\text{good}} \subseteq R$ the subset of these 'good' rows. By the local list recovery guarantee for C', with high probability on each such good row $r \in R_{\text{good}}$ the algorithm $A^r_{j_r}$ computes correctly most of the coordinates of $\widetilde{c}$ on this row. Now, by another averaging argument this implies in turn that for most columns $i' \in [n']$ it holds that both $\widetilde{c}$ is consistent with most of the input lists on the columnn and additionally, most of the guesses $(c'_r)_{r \in R}$ for this column are correct. As above, let us denote by $\text{Col}_{\text{good}} \subseteq [n']$ the subset of these 'good' columns. Finally, by the list recovery guarantee for C, on any good column $i' \in \text{Col}_{\text{good}}$ the i'th column $\widetilde{c}|_{[n] \times \{i'\}}$ of $\widetilde{c}$ is present in the output list $\mathcal{L}$, and we further show that with high probability $\widetilde{c}|_{[n] \times \{i'\}}$ is closest to $(c'_r)_{r \in R}$ on R among all codewords in $\mathcal{L}$, in which case $\widetilde{A}_J$ outputs the correct (i, i') symbol of $\widetilde{c}$. Details follow.

We start by showing the existence of a large subset of good rows $R_{\text{good}} \subseteq R$. For this observe that since the rows in R are chosen uniformly at random, and since $\widetilde{\alpha} \leq \frac{1}{10} \cdot \alpha' \cdot \delta$, by Chernoff bound (without replacement, see e.g. Lemma 5.1 in [GGR11]), with probability at least $1 - \exp(-(\alpha' \cdot \delta)^2 m) \geq 1 - \frac{\widetilde{\varepsilon}}{3}$ over the choice of R, it holds that $\text{dist}(\widetilde{c}|_{R \times [n']}, S|_{R \times [n']}) \leq \frac{\alpha' \cdot \delta}{8}$. If this is the case, then by averaging, for at least $1 - \frac{\delta}{8}$ fraction of the rows $r \in R$ it holds that $\text{dist}\left(\widetilde{c}|_{\{r\} \times [n']}, S|_{\{r\} \times [n']}\right) \leq \alpha'$. Let

$$R_{\text{good}} = \left\{ r \in R \mid \text{dist}\left(\widetilde{c}|_{\{r\} \times [n']}, S|_{\{r\} \times [n']}\right) \leq \alpha' \right\}.$$

Now we observe that for each good row $r \in R_{\text{good}}$, with high probability over the randomness of A' the algorithm $A^r_{j_r}$ computes correctly most of the coordinates of $\widetilde{c}$ on this row. For this note that since $1 - \varepsilon' \geq 1 - \frac{1}{100} \cdot \delta \cdot \widetilde{\varepsilon}$, with probability at least $1 - \frac{1}{100} \cdot \delta \cdot \widetilde{\varepsilon}$ over the randomness of A', independently for each row r, it holds that $A^r_{j_r}$ computes correctly at least $1 - \frac{1}{100} \cdot \delta \cdot \widetilde{\varepsilon}$ fraction of the coordinates of $\widetilde{c}|_{\{r\} \times [n']}$. Consequently, by Chernoff bound with probability at least $1 - \exp(-(\delta \cdot \widetilde{\varepsilon})^2 \cdot m) \geq 1 - \frac{\widetilde{\varepsilon}}{3}$ over the randomness of $\widetilde{A}$, it holds that $(A^r_{j_r})_{r \in R_{\text{good}}}$ compute correctly at least $1 - \frac{1}{24} \cdot \delta \cdot \widetilde{\varepsilon}$ fraction of the coordinates of $\widetilde{c}|_{R_{\text{good}} \times [n']}$. Finally, if this is the case then for at least $1 - \frac{\widetilde{\varepsilon}}{3}$ fraction of the columns $i' \in [n']$ it holds that $\text{dist}\left(\widetilde{c}|_{R_{\text{good}} \times \{i'\}}, (A^r_{j_r}(i'))_{r \in R_{\text{good}}}\right) \leq \frac{\delta}{8}$.

Next we show the existence of a large subset of good columns $\text{Col}_{\text{good}} \subseteq [n']$. So far we obtained that with probability at least $1 - \frac{2}{3} \cdot \widetilde{\varepsilon}$ over the randomness of $\widetilde{A}$, for at least $1 - \frac{\widetilde{\varepsilon}}{3}$ fraction of the columns $i' \in [n']$ it holds that $\text{dist}\left(\widetilde{c}|_{R \times \{i'\}}, (A^r_{j_r}(i'))_{r \in R}\right) \leq \frac{\delta}{4}$. Moreover, note that since $\widetilde{\alpha} \leq \frac{1}{10} \cdot \alpha \cdot \widetilde{\varepsilon}$, for at least $1 - \frac{\widetilde{\varepsilon}}{3}$ fraction of the columns $i' \in [n']$ it holds that $\text{dist}(\widetilde{c}|_{[n] \times \{i'\}}, S|_{[n] \times \{i'\}}) \leq \alpha$. Let

$$\text{Col}_{\text{good}} = \left\{ i' \in [n'] \mid \text{dist}(\widetilde{c}|_{[n] \times \{i'\}}, S|_{[n] \times \{i'\}}) \leq \alpha \text{ and } \text{dist}\left(\widetilde{c}|_{R \times \{i'\}}, (A^r_{j_r}(i'))_{r \in R}\right) \leq \frac{\delta}{4} \right\}.$$

Then by the above, with probability at least $1 - \frac{2}{3} \cdot \widetilde{\varepsilon}$ over the randomness of $\widetilde{A}$ it holds that $|\text{Col}_{\text{good}}| \geq (1 - \frac{2}{3} \cdot \widetilde{\varepsilon})|n'|$.

Now for each column $i' \in \mathrm{Col}_{\mathrm{good}}$ it holds that $\mathrm{dist}(\widetilde{c}|_{[n]\times\{i'\}}, S|_{[n]\times\{i'\}}) \leq \alpha$, and so $\widetilde{c}|_{[n]\times\{i'\}}$ must be in the output list $\mathcal{L}$ of the i'th column. Moreover, since the code C has relative distance δ, any codeword $\hat{c} \in \mathcal{L}$ other than $\widetilde{c}|_{[n]\times\{i'\}}$ must differ from $\widetilde{c}|_{[n]\times\{i'\}}$ by at least δ fraction of the coordinates. Furthermore, since R is chosen uniformly at random, by Chernoff bound, with probability at least $1-\exp(-\delta^2 m)$ over the choice of R it holds that $\mathrm{dist}(\hat{c}|_R, \widetilde{c}|_{R\times\{i'\}}) \geq 3\delta/4$. By union over all codewords in $\mathcal{L}$ this implies in turn that with probability at least $1-L\cdot\exp(-\delta^2 m) \geq 1-\frac{1}{9}\cdot\widetilde{\varepsilon}^2$ over the choice of R, for all codewords $\hat{c} \in \mathcal{L}\setminus\{\widetilde{c}|_{[n]\times\{i'\}}\}$ it holds that $\mathrm{dist}(\hat{c}|_R, \widetilde{c}|_{R\times\{i'\}}) \geq 3\delta/4$.

Next observe that if the above holds then for any column $i' \in \mathrm{Col}_{\mathrm{good}}$ we have on the one hand that $\mathrm{dist}\left(\widetilde{c}|_{R\times\{i'\}}, (c'_r)_{r\in R}\right) \leq \frac{\delta}{4}$, and on the other hand that

$$\mathrm{dist}\left(\hat{c}|_R, (c'_r)_{r\in R}\right) \geq \mathrm{dist}(\hat{c}|_R, \widetilde{c}|_{R\times\{i'\}}) - \mathrm{dist}\left(\widetilde{c}|_{R\times\{i'\}}, (c'_r)_{r\in R}\right) \geq \frac{\delta}{2}$$

for any $\hat{c} \in \mathcal{L}\setminus\{\widetilde{c}|_{[n]\times\{i'\}}\}$. So $\widetilde{c}|_{[n]\times\{i'\}}$ will be the codeword in $\mathcal{L}$ that is closest to $(c'_r)_{r\in R}$ on R, and consequently we will have $c = \widetilde{c}|_{[n]\times\{i'\}}$ and $c_i = \widetilde{c}_{i,i'}$. So we obtained that for each column $i' \in \mathrm{Col}_{\mathrm{good}}$, with probability at least $1-\frac{1}{9}\cdot\widetilde{\varepsilon}^2$ over the randomness of $\widetilde{A}$, the algorithm $\widetilde{A}_J$ computes the entire column i' correctly. By averaging, this implies in turn that with probability at least $1-\frac{\widetilde{\varepsilon}}{3}$ over the randomness of $\widetilde{A}$ the algorithm $\widetilde{A}_J$ computes correctly at least $1-\frac{\widetilde{\varepsilon}}{3}$ fraction of the coordinates of $\widetilde{c}|_{[n]\times\mathrm{Col}_{\mathrm{good}}}$.

Concluding, we obtained that with probability at least $1-\widetilde{\varepsilon}$ over the randomness of $\widetilde{A}$, the algorithm $\widetilde{A}_J$ computes correctly at least $1-\widetilde{\varepsilon}$ fraction of the coordinates of $\widetilde{c}$, so the algorithm $\widetilde{A}$ satistifes the local list recovery requirement. Next we analyze the performance of the algorithm.

Output list size: The resulting output list size equals the number of different strings $(j_r)_{r\in R} \in [L']^R$ which is

$$\widetilde{L} = (L')^m = \exp\left(\frac{\log(L')\cdot\log(L/\widetilde{\varepsilon})}{(\delta\cdot\alpha'\cdot\widetilde{\varepsilon})^2}\right).$$

Query complexity: The query complexity is

$$\widetilde{Q} = m\cdot Q' + n = O\left(\frac{\log(L/\widetilde{\varepsilon})}{(\delta\cdot\alpha'\cdot\widetilde{\varepsilon})^2}\right)\cdot Q' + n,$$

since Q' queries are needed in order to list recover each of the rows in R, and n additional queries are needed to globally list recover the i''th column.

Pre-Processing Time: The pre-processing algorithm generates a random set R, of size m, which takes $m\log n$ time to generate and store. The pre-processing algorithm then runs m independent copies of A' (once for each row in R), and this takes time $m\cdot T'_{\mathrm{pre}}$. Finally, the pre-processing algorithm generates the set J of size $(L')^m$. Thus the total pre-processing time is

$$m\left(\log n + T'_{\mathrm{pre}}\right) + \exp\left(m\log L'\right).$$

Running Time: Each local recovery algorithm runs m copies of the local recovery algorithm for C', which takes time $m \cdot T'$. Then it runs the global list recovery algorithm for C once, which takes time T, thus the total running time of each local recovery algorithm is

$$m \cdot T' + T.$$

□

5.2 Local list recovery

Next we show that the approximately locally list recoverable codes of Lemma 5.2 can be turned into locally list recoverable codes by pre-encoding the message with a locally decodable code.

Lemma 5.4. *Suppose that $C : \mathbb{F}^{\rho n} \to \mathbb{F}^n$ is a* systematic *linear code of rate ρ and relative distance δ that is (α, ℓ, L)-(globally) list recoverable, and $\widehat{C} : \mathbb{F}^{\widehat{k}} \to \mathbb{F}^{(\rho n)^t}$ is $(\widehat{Q}, \widehat{\alpha})$-locally decodable. Then $\widetilde{C} := C^{\otimes t}\big(\widehat{C}\big) : \mathbb{F}^{\widehat{k}} \to \mathbb{F}^{n^t}$ is $(\widetilde{Q}, \widetilde{\alpha}, \ell, \widetilde{L})$-locally list recoverable for $\widetilde{\alpha} = \alpha \cdot \widehat{\alpha} \cdot \rho^t \cdot \delta^{O(t)}$,*

$$\widetilde{Q} = \widehat{Q} \cdot n \cdot \frac{\log^t L}{(\alpha \cdot \widehat{\alpha})^{O(t)} \cdot (\rho \cdot \delta)^{O(t^2)}},$$

and

$$\widetilde{L} = \exp\left(\frac{\log^t L}{(\alpha \cdot \widehat{\alpha})^{O(t)} \cdot (\rho \cdot \delta)^{O(t^2)}}\right).$$

Moreover, the local list recovery algorithm for $\widetilde{C}$ has preprocessing time

$$\widetilde{T}_{pre} = \exp\left(\frac{\log^t L}{(\alpha \cdot \widehat{\alpha})^{O(t)} \cdot (\rho \cdot \delta)^{O(t^2)}}\right) \cdot \log n,$$

and if the (global) list recovery algorithm for C runs in time T and the local decoding algorithm for $\widehat{C}$ runs in time $\widehat{T}$ then the local list recovery algorithm for $\widetilde{C}$ runs in time

$$\widetilde{T} = \widehat{T} + \widehat{Q} \cdot T \cdot \frac{\log^t L}{(\alpha \cdot \widehat{\alpha})^{O(t)} \cdot (\rho \cdot \delta)^{O(t^2)}}.$$

Proof. Setting $\widetilde{\varepsilon} = \widehat{\alpha}\rho^t$ in Lemma 5.2, we conclude that the tensor code $\overline{C} := C^{\otimes t} : \mathbb{F}^{(\rho n)^t} \to \mathbb{F}^{n^t}$ is $(\overline{Q}, \overline{\alpha}, \widetilde{\varepsilon}, \ell, \overline{L})$-approximately locally list recoverable for $\overline{\alpha} = \alpha\widetilde{\varepsilon}\delta^{O(t)}$. Note furthermore that since C is systematic then so is $\overline{C}$.

Intuitively, the proof works as follows: to recover the ith message symbol, x_i, run the local decoder of the inner code $\widehat{C}$ to obtain a set of $\widehat{Q}$ indices in $\mathbb{F}^{(\rho n)^t}$ that, if queried, would allow you to recover x_i. Since the code $\overline{C}$ is systematic, those symbols correspond to symbols in the big code $\widetilde{C}$. Use the approximate local list recovery algorithm for $\overline{C}$ to obtain $\overline{L}$ guesses for each of these $\widehat{Q}$ symbols. Finally, for each of these $\overline{L}$ sets of $\widehat{Q}$ "guesses" run the local decoding algorithm for $\widehat{C}$ to obtain $\overline{L}$ guesses for x_i. Since $\overline{C}$ is only approximately locally list recoverable, there will be a subset of symbols on which the approximate local list decoder fails, but by carefully choosing parameters, these errors can be handled by the local decoding procedure of $\widehat{C}$.

It is not hard to see that the query complexity of this algorithm will be $\widehat{Q}$ times the query complexity of $C^{\otimes t}$, and the output list size will be the same as that of $C^{\otimes t}$.

Below, we describe the algorithm and proof of correctness in more detail. Let $(\overline{A}_1, \cdots, \overline{A}_{\overline{L}}) \leftarrow \overline{A}(\cdot)$ be the approximate local list recovery algorithms for $\overline{C}$. In Algorithm 2 we describe the local list recovery algorithms $\widetilde{A}_1, \ldots, \widetilde{A}_{\overline{L}}$ for the code $\widetilde{C} := \overline{C}(\widehat{C})$.

Algorithm 2 The local list recovery algorithm for $\widetilde{C} := C^{\otimes t}(\widehat{C})$.

function $\widetilde{A}_j(i \in [\widehat{k}])$

$\triangleright$ $\widetilde{A}_j$ receives oracle access to lists $S \in \binom{\mathbb{F}}{\ell}^{n^t}$

Run the local decoding algorithm for $\widehat{C}$ on input i to obtain a set of $\widehat{Q}$ indices that the local decoder would query.

Let $R \subseteq [(\rho n)^t]$ be the subset of indices that would be queried.

Let $\overline{R} \subseteq [n^t]$ be the indices in $\overline{C}$ encoding the indices of R.

$\triangleright$ $\overline{R}$ exists and $|\overline{R}| = |R| = \widehat{Q}$ because $\overline{C}$ is systematic.

for $\overline{r} \in \overline{R}$ **do**

Let $c_j^{(\overline{r})} \leftarrow \overline{A}_j(\overline{r})$ (on oracle access to S)

end for

Run the local decoder for $\widehat{C}$ on input $\{c_j^{(\overline{r})}\}_{\overline{r} \in \overline{R}}$ to obtain a guess $x_i^{(j)}$ for the ith symbol of the message

Return: $x_i^{(j)}$

end function

Correctness: Given a string of lists $S \in \binom{\mathbb{F}}{\ell}^{n^t}$, suppose $\widetilde{c} = \widetilde{C}(x) = \overline{C}(\widehat{C}(x))$ is a codeword of $\widetilde{C}$ that is $\widetilde{\alpha}$-close to S. We need to show that with probability at least $\frac{2}{3}$ over the randomness of $\widetilde{A}$ there is a $j \in [\overline{L}]$ such that for all $i \in [\widehat{k}]$, $\Pr[\widetilde{A}_j(i) = x_i] \geq \frac{2}{3}$.

Since $\widetilde{\varepsilon} = \widehat{\alpha}\rho^t$ the tensor code $\overline{C} = C^{\otimes t}$ is $(\overline{Q}, \overline{\alpha}, \widetilde{\varepsilon}, \ell, \overline{L})$-approximately locally list recoverable, for $\overline{\alpha} = \alpha\widetilde{\varepsilon}\delta^{O(t)}$. Thus if $\widetilde{c}$ is $\widetilde{\alpha}$-close to S, then with probability $1 - \widetilde{\varepsilon}$, over the randomness of $\overline{A}$, there is a $j \in [\overline{L}]$ such that

$$\Pr_{i \in [n^t]}[\overline{A}_j(i) = \overline{C}(x)_i] \geq 1 - \widetilde{\varepsilon}.$$

This means that with probability $1 - \widetilde{\varepsilon}$ over the randomness of $\overline{A}$, there is a j such that $\overline{A}_j(\cdot)$ is $\widetilde{\varepsilon}$-close to the codeword $\overline{C}(\widehat{C}(x))$. Since $\overline{C}$ is systematic, the inner codeword $\widehat{C}(x)$ must be $\widetilde{\varepsilon}\rho^{-t}$-close to the restriction of S to the information symbols. Since $\widehat{C}$ is $(\widehat{Q}, \widehat{\alpha})$-locally decodable, and $\widehat{\alpha} = \widetilde{\varepsilon}\rho^{-t}$, then by the local decoding property of the inner code, $\widehat{C}$, given any $i \in [\widehat{k}]$, and oracle access to $\overline{A}_j(\cdot)$, the local decoding algorithm for $\widehat{C}$ will recover x_i with probability at least $\frac{2}{3}$.

Output list size and query complexity: The query complexity is

$$\widetilde{Q} = \widehat{Q} \cdot \overline{Q} = \widehat{Q} \cdot n \cdot \frac{\log^t L}{(\alpha \cdot \widehat{\alpha} \cdot \rho^t)^{O(t)} \cdot \delta^{O(t^2)}},$$

and the output list size is

$$\widetilde{L} = \overline{L} = \exp\left(\frac{\log^t L}{(\alpha \cdot \widehat{\alpha} \cdot \rho^t)^{O(t)} \cdot \delta^{O(t^2)}}\right).$$

It can also be verified that the running times are as required.

□

5.3 Instantiations

In what follows we shall instantiate Lemma 5.4 in two ways. For both, we will use the high-rate LDCs of [KMRS16] (Theorem 5.8 below) as the code $\widehat{C}$. In the first instantiation, which is more straightforward, we just use a random linear code (via Corollary 4.4) as the code C. This yields a code $\widetilde{C}$ that is not efficiently encodable, nor efficiently list recoverable, but it does achieve small locality together with constant alphabet size and constant output list size. The second instantiation, which does yield efficinetly encodable (in nearly-linear time) and efficiently list recoverable (in sub-linear time) codes, uses a modification of the Algebraic Geometry subcodes studied in [GX13, GK16b] (Theorem A.1) as the code C. These latter codes have constant alphabet size, but slightly super-constant output list size, which means that our efficient construction will as well. In more detail, we obtain the following pair of theorems.

Theorem 5.5 (High-rate locally list recoverable codes, non-efficient)**.** *There is a constant c so that the following holds. Choose $\varepsilon > 0$ and positive integers ℓ, t. Suppose that $s \geq \max\{1/\varepsilon^c, c(\log \ell)t/\varepsilon\}$. Then there exists an infinite family of $\mathbb{F}_2$-linear codes $\{C_n\}_n$ such that the following holds.*

1. *$C_n : \mathbb{F}_{2^s}^{(1-\varepsilon)n} \to \mathbb{F}_{2^s}^n$ has rate $1-\varepsilon$ and relative distance at least $(\varepsilon/(16t))^t$.*
2. *C_n is (Q, α, ℓ, L)-locally list recoverable for $\alpha = (\varepsilon/t)^{O(t)}$,*

$$Q = n^{1/t} \cdot 2^{O(\sqrt{\log n \cdot \log\log n})} \cdot (s\ell)^t \cdot (t/\varepsilon)^{O(t^2)},$$

and

$$L = \exp\left((s\ell)^t \cdot (t/\varepsilon)^{O(t^2)}\right).$$

In particular, when ε, ℓ, t, s are constant we get that $\alpha = \Omega(1)$, $Q = n^{1/t+o(1)}$, and $L = O(1)$.

Theorem 5.6 (High-rate locally list recoverable codes, efficient)**.** *There is a constant c so that the following holds. Choose $\varepsilon > 0$ and a positive integer ℓ. Let $\{t_n\}_n$ be a sequence of positive integers, non-decreasing with n, so that t_0 is sufficiently large and*

$$t_n \leq \sqrt{\frac{\varepsilon \log_q(n)}{c\ell}}.$$

For each choice of t choose $s = s(t)$ so that $s \geq \max\{1/\varepsilon^c, c(\log \ell)t/\varepsilon\}$ is even. Then there exists an infinite family of $\mathbb{F}_2$-linear codes $\{C_n\}_n$ such that the following holds. Below, to simplify notation we use t instead of t_n and s instead of $s(t_n)$.

1. *$C_n : \mathbb{F}_{2^s}^{(1-\varepsilon)n} \to \mathbb{F}_{2^s}^n$ has rate $1-\varepsilon$ and relative distance at least $(\Omega(\varepsilon/t))^{2t}$.*
2. *C_n is (Q, α, ℓ, L)-locally list recoverable for $\alpha = (\varepsilon/t)^{O(t)}$,*

$$Q = n^{1/t} \cdot 2^{O(\sqrt{\log n \cdot \log\log n})} \cdot \exp\left(\frac{t^2 \ell s}{\varepsilon} \cdot \exp(\log^* n) + t \log s\right),$$

and

$$L = \exp\left(\exp\left(\frac{t^2 \ell s}{\varepsilon} \cdot \exp(\log^* n) + t \log s\right)\right).$$

3. *The local list recovery algorithm for C_n has preprocessing time*

$$T_{pre} = \exp\left(\exp\left(\frac{t^2 \ell s}{\varepsilon} \cdot \exp(\log^* n) + t \log s\right)\right) \cdot \log n,$$

and running time

$$T = n^{O(1/t)} \cdot 2^{O(\sqrt{\log n \cdot \log\log n})} \cdot \exp\left(\exp\left(\frac{t\ell s}{\varepsilon} \cdot \exp(\log^* n) + \log s\right)\right).$$

4. *C_n can be encoded in time*

$$n \cdot 2^{O(\sqrt{\log n \cdot \log\log n})} + t \cdot n^{1+O(1/t)}.$$

In particular, when $\varepsilon, \ell, t_n = t, s$ are constant we get that $\alpha = \Omega(1)$, $Q = n^{1/t+o(1)}$, $L = \exp(\exp(\exp(\log^* n)))$, $T_{pre} = \log^{1+o(1)} n$, $T = n^{O(1/t)}$, and encoding time is $n^{1+O(1/t)}$.

Remark 5.7 (Super-constant t)**.** Theorem 5.6 is interesting even when $t_n = t$ is a sufficiently large constant that does not depend on n. For our applications, we will need to take t_n to be slightly super-constant, so we allow for that in the statement of Theorem 5.6.

To prove the above theorems we use the following result from [KMRS16, KMRS15] that shows the existence of high-rate LDCs with sub-polynomial query complexity.

Theorem 5.8 ([KMRS15], Theorem 1.3)**.** *There is a constant c so that the following holds. Choose $\rho \in [0,1]$ and $\varepsilon > 0$. Then there exists an infinite sequence $\{n_i\}_i$ such that for any $n = n_i$ in the sequence and for any $s \geq 1/\varepsilon^c$ there exists an $\mathbb{F}_2$-linear code C_n satisfying:*

1. *$C_n : \mathbb{F}_{2^s}^{\rho n} \to \mathbb{F}_{2^s}^n$ has rate ρ and relative distance at least $1 - \rho - \varepsilon$.*
2. *C_n is $(2^{O(\sqrt{\log n \cdot \log\log n})}, \frac{1-\rho-\varepsilon}{2})$-locally decodable in time $s \cdot 2^{O(\sqrt{\log n \cdot \log\log n})}$.*
3. *C_n can be encoded in time $n \cdot 2^{O(\sqrt{\log n \cdot \log\log n})}$.*

Remark 5.9. We remark about a few differences between the above theorem and Theorem 1.3 in [KMRS15]:

1. Theorem 1.3 in [KMRS15] talks about locally correctable codes (LCCs) instead of locally decodable codes (LDCs). The difference between LCCs and LDCs is that for LCCs the local correction algorithm is required to decode codeword bits as opposed to message bits. However, it can be shown that the result holds for LDCs as well (see discussion at end of Section 1.1 in [KMRS15]).
2. Theorem 1.3 in [KMRS15] only states the existence of *some* $s_0 \leq 1/\varepsilon^c$ for which the above holds, however it can be verified that the result holds for *any* $s \geq s_0$ as well.
3. Encoding time is not stated explicitly in [KMRS15], Lemma 3.2 and [Kop15], Appendix A.

We proceed to the proof of Theorem 5.5.

Proof of theorem 5.5. Let c' be a sufficiently large constant for which both Corollary 4.4 and Theorem 5.8 hold, and suppose that $s \geq \max\{(4/\varepsilon)^{c'}, 16c'(\log \ell)t/\varepsilon\}$.

Let $C : \mathbb{F}_{2^s}^{((1-\varepsilon/2)n)^{1/t}} \to \mathbb{F}_{2^s}^{n^{1/t}}$ be a linear code of rate $(1-\varepsilon/2)^{1/t} \leq 1-\varepsilon/(8t)$ (the inequality holds since $(1-x)^y \leq 1-xy/4$ for $x, y \in [0,1]$, see e.g. Fact 2.1 in [KMRS15]) that is $(\frac{\varepsilon}{16t}, \ell, 2^{O(s\ell t/\varepsilon)})$-list recoverable whose existence is guaranteed by Corollary 4.4 for sufficiently large n (depending on ε, ℓ, t, s). Note furthermore that by Theorem 4.1 we may assume that C has relative distance at least $\varepsilon/(16t)$. Finally, note that one may assume that the code C is systematic by performing Gaussian elimination on the generating matrix of C. Let $\widehat{C} : \mathbb{F}_{2^s}^{(1-\varepsilon)n} \to \mathbb{F}_{2^s}^{(1-\varepsilon/2)n}$ be an $\mathbb{F}_2$-linear code of rate $\frac{1-\varepsilon}{1-\varepsilon/2} \leq 1 - \frac{\varepsilon}{2}$ that is $\left(2^{O(\sqrt{\log n \cdot \log\log n})}, \frac{\varepsilon}{4}\right)$-locally decodable given by Theorem 5.8 for infinite values of n (depending on ε).

Let $C_n := C^{\otimes t}(\widehat{C})$ for any n for which both C and $\hat{C}$ exist. Then $C_n : \mathbb{F}_{2^s}^{(1-\varepsilon)n} \to \mathbb{F}_{2^s}^{n}$ is an $\mathbb{F}_2$-linear code of rate $1-\varepsilon$ and relative distance at least $(\varepsilon/(16t))^t$. Moreover, by Lemma 5.4 the code C_n is (Q, α, ℓ, L)-locally list recoverable for $\alpha = (\varepsilon/t)^{O(t)}$,

$$Q = n^{1/t} \cdot 2^{O(\sqrt{\log n \cdot \log\log n})} \cdot \left(\frac{s\ell t}{\varepsilon}\right)^t \cdot (t/\varepsilon)^{O(t^2)} = n^{1/t} \cdot 2^{O(\sqrt{\log n \cdot \log\log n})} \cdot (s\ell)^t \cdot (t/\varepsilon)^{O(t^2)},$$

and

$$L = \exp\left((s\ell)^t \cdot (t/\varepsilon)^{O(t^2)}\right).$$

□

Next we prove Theorem 5.6.

Proof of Theorem 5.6. Fix any $n \in \mathbb{N}$ so that Theorem 5.8 may be instantiated with block length $(1-\varepsilon/2)n$, and so that

$$n^{1/t_n} \geq q^{8c_0\ell t_n/\varepsilon}, \tag{1}$$

where c_0 is the constant from the statement of Theorem A.1. By Theorem 5.8 and the assumption on t_n, there are infinitely many such n. For the rest of the proof, we will denote t_n by t, since n is now fixed.

Let c' be a sufficiently large constant (independent of n) for which both Theorem A.1 and Theorem 5.8 hold, and suppose that $s \geq \max\{(4/\varepsilon)^{c'}, 8c'(\log \ell)t/\varepsilon\}$ is even. Then the code C_n is constructed as follows.

Let $C : \mathbb{F}_{2^s}^{((1-\varepsilon/2)n)^{1/t}} \to \mathbb{F}_{2^s}^{n^{1/t}}$ be a linear code of rate $1-\varepsilon/(8t) \geq (1-\varepsilon/2)^{1/t}$ and relative distance $\Omega((\varepsilon/8t)^2)$, that is $(\Omega((\varepsilon/8t)^2), \ell, L_0)$-list recoverable whose existence is guaranteed by Theorem A.1, where

$$\begin{aligned} L_0 &= \exp_{2^s}\left(\exp_{2^s}\left(\frac{\ell t}{\varepsilon} \cdot \exp(\log^*(n))\right)\right) \\ &= \exp\left(\exp\left(\frac{\ell s t}{\varepsilon}\exp(\log^*(n))\right) + \log(s)\right). \end{aligned}$$

Here, we are using (1) to ensure that we may choose the block length $n^{1/t}$ and rate $1-\varepsilon/(8t)$ in Theorem A.1.

Once more, one may further assume that the code C is systematic by performing Gaussian elimination on the generating matrix of C. Let $\widehat{C} : \mathbb{F}_{2^s}^{(1-\varepsilon)n} \to \mathbb{F}_{2^s}^{(1-\varepsilon/2)n}$ be an $\mathbb{F}_2$-linear code of rate $\frac{1-\varepsilon}{1-\varepsilon/2} \leq 1 - \frac{\varepsilon}{2}$ that is $\left(2^{O(\sqrt{\log n \cdot \log\log n})}, \frac{\varepsilon}{4}\right)$-locally decodable given by Theorem 5.8.

Let $C_n := C^{\otimes t}(\widehat{C})$. Then $C_n : \mathbb{F}_{2^s}^{(1-\varepsilon)n} \to \mathbb{F}_{2^s}^n$ is an $\mathbb{F}_2$-linear code of rate $1-\varepsilon$ and relative distance at least $(\Omega(\varepsilon/t))^{2t}$. Moreover, by Lemma 5.4 the code C_n is (Q, α, ℓ, L)-locally list recoverable for $\alpha = (\varepsilon/t)^{O(t)}$,

$$Q = n^{1/t} \cdot 2^{O(\sqrt{\log n \cdot \log\log n})} \cdot \exp\left(\frac{t^2 \ell s}{\varepsilon} \cdot \exp(\log^* n) + t \log s\right),$$

and

$$L = \exp\left(\exp\left(\frac{t^2 \ell s}{\varepsilon} \cdot \exp(\log^* n) + t \log s\right)\right).$$

Next observe that since C can be list recovered in time $\mathrm{poly}(n^{1/t}, L_0)$, and $\widehat{C}$ can be locally decoded in time $2^{O(\sqrt{\log n \cdot \log\log n})}$, the local list recovery algorithm for C_n has preprocessing time

$$T_{\mathrm{pre}} = \exp\left(\exp\left(\frac{t^2 \ell s}{\varepsilon} \cdot \exp(\log^* n) + t \log s\right)\right) \cdot \log n,$$

and running time

$$T = n^{O(1/t)} \cdot 2^{O(\sqrt{\log n \cdot \log\log n})} \cdot \exp\left(\exp\left(\frac{t \ell s}{\varepsilon} \cdot \exp(\log^* n) + \log s\right)\right).$$

Finally, since C can be encoded in time $\mathrm{poly}(n^{1/t})$ then $C^{\otimes t}$ can be encoded in time $t \cdot n \cdot \mathrm{poly}(n^{1/t})$, and consequently the encoding time of $C_n := C^{\otimes t}(\widehat{C})$ is

$$n \cdot 2^{O(\sqrt{\log n \cdot \log\log n})} + t \cdot n^{1+O(1/t)}.$$

□

6 Capacity achieving locally list decodable codes

Next we show the existence of capacity achieving locally list decodable codes over large (but constant) alphabet. As before we exhibit two instantiations of this result: In the first instantiation we obtain capacity achieving locally list decodable codes that have constant output list size, however these codes cannot be efficiently encoded or list decoded; In the second instantiation we obtain capacity achieving locally list decodable codes that are efficiently encodable (in nearly-linear time) and efficiently list decodable (in sub-linear time), however these codes have slightly super-constant output list size. These latter codes can also achieve $n^{o(1)}$ query complexity at the cost of increasing the alphabet and output list sizes to $n^{o(1)}$. Towards our GV bound application, we actually present a stronger version that applies also to list recovery, where list decoding corresponds to the special case in which the input list size ℓ equals 1.

Theorem 6.1 (Capacity achieving locally list decodable / recoverable codes, non-efficient)**.** *There is a constant c so that the following holds. Choose $\rho \in [0,1]$, $\varepsilon > 0$, a positive integer ℓ, and sufficiently large integer t. Let*

$$s_0 := \max\left\{c(1-\rho-\varepsilon/2)\log(1/(1-\rho-\varepsilon/2))/\varepsilon, c(\rho+\varepsilon/2)\log(1/(\rho+\varepsilon/2))/\varepsilon, c(\log \ell)/\varepsilon\right\},$$

and suppose that $s \geq \max\left\{(t/\varepsilon)^{ct}, c\ell t/\varepsilon^2\right\} \cdot s_0/\rho$.

Then there exists an infinite family of $\mathbb{F}_2$-linear codes $\{C_n\}_n$ such that the following holds.

1. $C_n : \mathbb{F}_{2^s}^{\rho n} \to \mathbb{F}_{2^s}^n$ *has rate* ρ *and relative distance at least* $1 - \rho - \varepsilon$.

2. C_n *is* $(Q, 1-\rho-\varepsilon, \ell, L)$*-locally list recoverable for*

$$Q = n^{1/t} \cdot 2^{O(\sqrt{\log n \cdot \log\log n})} \cdot s^{t+O(1)} \cdot 2^{O(s_0 \ell t/\varepsilon)} \cdot (t/\varepsilon)^{O(t^2)},$$

and

$$L = \exp\left(s^t \cdot 2^{O(s_0 \ell t/\varepsilon)} \cdot (t/\varepsilon)^{O(t^2)}\right).$$

In particular, when $\rho, \varepsilon, \ell, t, s$ *are constant we get that* $Q = n^{1/t+o(1)}$ *and* $L = O(1)$.

Theorem 6.2 (Capacity achieving locally list decodable / recoverable codes, efficient)**.** *There is a constant* c *so that the following holds. Choose* $\rho \in [0,1]$, $\varepsilon > 0$, *and a positive integer* ℓ. *Let* $\{t_n\}_n$ *be a sequence of positive integers, non-decreasing with* n, *so that* t_0 *is sufficiently large, and so that*

$$t_n \leq \sqrt{\frac{\varepsilon \log_q(n)}{c\ell}}.$$

Let

$$s_0 := \max\left\{c(1-\rho-\varepsilon/2)\log(1/(1-\rho-\varepsilon/2))/\varepsilon, c(\rho+\varepsilon/2)\log(1/(\rho+\varepsilon/2))/\varepsilon, c(\log \ell)/\varepsilon\right\},$$

and for each choice of t *choose* $s = s(t)$ *so that* $s \geq \max\left\{(t/\varepsilon)^{ct}, c\ell t/\varepsilon^2\right\} \cdot s_0/\rho$ *is even.*

Then there exists an infinite family of $\mathbb{F}_2$*-linear codes* $\{C_n\}_n$ *such that the following holds. Below, we use* t *to denote* t_n *and* s *to denote* $s(t_n)$ *to simplify notation.*

1. $C_n : \mathbb{F}_{2^s}^{\rho n} \to \mathbb{F}_{2^s}^n$ *has rate* ρ *and relative distance at least* $1 - \rho - \varepsilon$.

2. C_n *is* $(Q, 1-\rho-\varepsilon, \ell, L)$*-locally list recoverable for*

$$Q = n^{1/t} \cdot 2^{O(\sqrt{\log n \cdot \log\log n})} \cdot \exp\left(t^2 s \cdot 2^{O(s_0 \ell/\varepsilon)} \cdot \exp(\log^* n) + t \log s\right)$$

and

$$L = \exp\left(\exp\left(t^2 s \cdot 2^{O(s_0 \ell/\varepsilon)} \cdot \exp(\log^* n) + t \log s\right)\right).$$

3. *The local list recovery algorithm for* C_n *has preprocessing time*

$$T_{pre} = \exp\left(\exp\left(t^2 s \cdot 2^{O(s_0 \ell/\varepsilon)} \cdot \exp(\log^* n) + t \log s\right)\right) \cdot \log n,$$

and running time

$$T = n^{O(1/t)} \cdot 2^{O(\sqrt{\log n \cdot \log\log n})} \cdot \exp\left(\exp\left(ts \cdot 2^{O(s_0 \ell/\varepsilon)} \cdot \exp(\log^* n) + \log s\right)\right).$$

4. C_n *can be encoded in time*

$$n \cdot 2^{O(\sqrt{\log n \cdot \log\log n})} + t \cdot n^{1+O(1/t)} + O(n \cdot 2^{s^2}) + s^{O(1)} \cdot n \cdot \mathrm{polylog} n.$$

In particular,

- *When $\rho, \varepsilon, \ell, t, s$ are constant we see that $Q = n^{1/t+o(1)}$, $L = \exp(\exp(\exp(\log^* n)))$, $T_{pre} = \log^{1+o(1)} n$, $T = n^{O(1/t)}$, and encoding time is $n^{1+O(1/t)}$.*
- *Alternatively, when ρ, ε, ℓ are constant, $t = t_n = O\left(\frac{\log\log\log(n)}{(\log\log\log\log(n))^2}\right)$, and $s = t^{O(t)}$ we see that Q, L, T_{pre}, T are of the form $n^{o(1)}$ and encoding time is $n^{1+O(1/t)} = n^{1+o(1)}$.*

To prove the above theorems we shall use the following lemma from [GKO+17] that gives a distance amplification procedure for local list recovery.

Lemma 6.3 ([GKO+16], Lemma 5.4)**.** *There is a constant c so that the following holds. For any $\delta_{out}, \alpha_{out}, \varepsilon > 0$ there exists an integer $d \leq (\delta_{out} \cdot \alpha_{out} \cdot \varepsilon)^{-c}$ such that the following holds.*

- *Let $C_{out} : (\Sigma_{out})^{\rho_{out} \cdot n_{out}} \to (\Sigma_{out})^{n_{out}}$ be an $\mathbb{F}$-linear code of rate ρ_{out} and relative distance δ_{out} that is $(Q, \alpha_{out}, \ell_{out}, L_{out})$-locally list recoverable.*
- *Let $C_{in} : (\Sigma_{in})^{\rho_{in} \cdot n_{in}} \to (\Sigma_{in})^{n_{in}}$ be an $\mathbb{F}$-linear code of rate ρ_{in} and relative distance δ_{in} that is $(\alpha_{in}, \ell_{in}, L_{in})$-(globally) list recoverable.*
- *Additionally, suppose that $n_{in} \geq d$, $|\Sigma_{out}| = |\Sigma_{in}|^{\rho_{in} \cdot n_{in}}$ and $L_{in} \leq \ell_{out}$.*

Then there exists an $\mathbb{F}$-linear code $C : (\Sigma_{in}^{n_{in}})^{(\rho_{in} \cdot \rho_{out}) \cdot n_{out}} \to (\Sigma_{in}^{n_{in}})^{n_{out}}$ of rate $\rho_{in} \cdot \rho_{out}$ and relative distance at least $\delta_{in} - \varepsilon$ that is $(O(Q \cdot n_{in}^2 \cdot \log n_{in}), \alpha_{in} - \varepsilon, \ell_{in}, L_{out})$-locally list recoverable.

Moreover,

- *If the local list recovery algorithm for C_{out} has preprocessing time $T_{pre,out}$ and running time T_{out}, and the global list recovery algorithm for C_{in} has running time T_{in}, then the local list recovery algorithm for C has preprocessing time $T_{pre,out}$ and running time*

$$O(T_{out}) + O(Q \cdot T_{in}) + \mathrm{poly}(Q, n_{in}, \ell_{in}).$$

- *If the encoding times of C_{out}, C_{in} are $\widehat{T}_{out}, \widehat{T}_{in}$ respectively then the encoding time of C is*

$$O(\widehat{T}_{out}) + O(n_{out} \cdot \widehat{T}_{in}) + n_{out} \cdot \mathrm{poly}(n_{in}, \log n_{out}).$$

Remark 6.4. The definition of locally list recoverable codes in [GKO+17] is stronger than ours since it requires an additional soundness property which guarantees that with high probability, all local algorithms in the output list compute an actual codeword. This requirement is not needed in our setting, and it can be verified that the proof of the above lemma goes through also with our weaker definition. Also, the encoding time was not mentioned explicitly in [GKO+17] but it can be deduced from the proof of the lemma.

We proceed to the proof of Theorem 6.1.

Proof of Theorem 6.1. Let c' be a sufficiently large constant for which both Corollary 4.4, Theorem 5.5 and Lemma 6.3 hold, and suppose that $s \geq \max\left\{(t/\varepsilon)^{c't}, 32(c')^2 \ell t/\varepsilon^2\right\} \cdot s_0/\rho$ for

$$s_0 := \max\left\{4c'(1-\rho-\varepsilon/2)\log(1/(1-\rho-\varepsilon/2))/\varepsilon, 4c'(\rho+\varepsilon/2)\log(1/(\rho+\varepsilon/2))/\varepsilon, 4c'(\log \ell)/\varepsilon\right\}.$$

Let $C_{in} : \mathbb{F}_{2^{s_0}}^{(\rho+\varepsilon/4)\cdot(s/s_0)} \to \mathbb{F}_{2^{s_0}}^{s/s_0}$ be a linear code of rate $\rho+\varepsilon/4$ that is $\left(1-\rho-\varepsilon/2, \ell, 2^{4c's_0\ell/\varepsilon}\right)$-list recoverable whose existence is guaranteed by Corollary 4.4 for sufficiently large t (depending on

$\rho, \varepsilon, \ell)$. Note furthermore that by Theorem 4.1 we may assume that C_{in} has relative distance at least $1 - \rho - \varepsilon/2$.

Let $C_{out} : \mathbb{F}_{2^{(\rho+\varepsilon/4)s}}^{\rho n/(\rho+\varepsilon/4)} \to \mathbb{F}_{2^{(\rho+\varepsilon/4)s}}^{n}$ be an $\mathbb{F}_2$-linear code of rate $\frac{\rho}{\rho+\varepsilon/4} \leq 1-\varepsilon/8$ and relative distance at least $(\varepsilon/(128t))^t$, given by Theorem 5.5 for infinite values of n (depending on $\rho, \varepsilon, \ell, t, s$), that is $(Q_{out}, (\varepsilon/t)^{O(t)}, 2^{4c's_0\ell/\varepsilon}, L_{out})$-locally list recoverable for

$$Q_{out} = n^{1/t} \cdot 2^{O(\sqrt{\log n \cdot \log\log n})} \cdot s^t \cdot 2^{O(s_0\ell t/\varepsilon)} \cdot (t/\varepsilon)^{O(t^2)}$$

and

$$L_{out} = \exp\left(s^t \cdot 2^{O(s_0\ell t/\varepsilon)} \cdot (t/\varepsilon)^{O(t^2)}\right).$$

Then by Lemma 6.3 for any n for which C_{out} exists there exists an $\mathbb{F}_2$-linear code $C_n : \mathbb{F}_{2^s}^{\rho n} \to \mathbb{F}_{2^s}^{n}$ of rate ρ and relative distance at least $1-\rho-\varepsilon$ that is $(Q, 1-\rho-\varepsilon, \ell, L)$-locally list recoverable for

$$Q = n^{1/t} \cdot 2^{O(\sqrt{\log n \cdot \log\log n})} \cdot s^{t+O(1)} \cdot 2^{O(s_0\ell t/\varepsilon)} \cdot (t/\varepsilon)^{O(t^2)},$$

and

$$L = \exp\left(s^t \cdot 2^{O(s_0\ell t/\varepsilon)} \cdot (t/\varepsilon)^{O(t^2)}\right).$$

□

Next we prove Theorem 6.2.

Proof of theorem 6.2. Let c' be a sufficiently large constant for which both Corollary 4.4, Theorem 5.6 and Lemma 6.3 hold. Fix $n \in \mathbb{N}$ so that Theorem 5.6 guarantees the existence of a code with block length n and using parameter $t = t_n$. (By Theorem 5.6, there are infinitely many such n). Now that n is fixed, we will use t to denote t_n. Suppose that $s = s(t) \geq \max\left\{(t/\varepsilon)^{c't}, 32(c')^2\ell t/\varepsilon^2\right\} \cdot s_0/\rho$ is even for

$$s_0 := \max\left\{4c'(1-\rho-\varepsilon/2)\log(1/(1-\rho-\varepsilon/2))/\varepsilon, 4c'(\rho+\varepsilon/2)\log(1/(\rho+\varepsilon/2))/\varepsilon, 4c'(\log\ell)/\varepsilon\right\}.$$

The code C_n is constructed as follows.

Let $C_{in} : \mathbb{F}_{2^{s_0}}^{(\rho+\varepsilon/4)\cdot(s/s_0)} \to \mathbb{F}_{2^{s_0}}^{s/s_0}$ be a linear code of rate $\rho+\varepsilon/4$ that is $\left(1-\rho-\varepsilon/2, \ell, 2^{4c's_0\ell/\varepsilon}\right)$-list recoverable whose existence is guaranteed by Corollary 4.4. Note furthermore that by Theorem 4.1 we may assume that C_{in} has relative distance at least $1-\rho-\varepsilon/2$.

Let $C_{out} : \mathbb{F}_{2^{(\rho+\varepsilon/4)s}}^{\rho n/(\rho+\varepsilon/4)} \to \mathbb{F}_{2^{(\rho+\varepsilon/4)s}}^{n}$ be an $\mathbb{F}_2$-linear code of rate $\frac{\rho}{\rho+\varepsilon/4} \leq 1-\varepsilon/8$ and relative distance at least $(\Omega(\varepsilon/t))^{2t}$, given by Theorem 5.6, that is $(Q_{out}, (\varepsilon/t)^{O(t)}, 2^{4c's_0\ell/\varepsilon}, L_{out})$-locally list recoverable for

$$Q_{out} = n^{1/t} \cdot 2^{O(\sqrt{\log n \cdot \log\log n})} \cdot \exp\left(t^2 s \cdot 2^{O(s_0\ell/\varepsilon)} \cdot \exp(\log^* n) + t\log s\right)$$

and

$$L_{out} = \exp\left(\exp\left(t^2 s \cdot 2^{O(s_0\ell/\varepsilon)} \cdot \exp(\log^* n) + t\log s\right)\right).$$

Then by Lemma 6.3 there exists an $\mathbb{F}_2$-linear code $C_n : \mathbb{F}_{2^s}^{\rho n} \to \mathbb{F}_{2^s}^{n}$ of rate ρ and relative distance at least $1-\rho-\varepsilon$ that is $(Q, 1-\rho-\varepsilon, \ell, L)$-locally list recoverable for

$$Q = n^{1/t} \cdot 2^{O(\sqrt{\log n \cdot \log\log n})} \cdot \exp\left(t^2 s \cdot 2^{O(s_0\ell/\varepsilon)} \cdot \exp(\log^* n) + t\log s\right)$$

and

$$L = \exp\left(\exp\left(t^2 s \cdot 2^{O(s_0 \ell/\varepsilon)} \cdot \exp(\log^* n) + t \log s\right)\right).$$

The stated running and encoding times follow similarly.

□

7 Nearly-linear time capacity achieving list decodable codes

In this section we show how to construct capacity achieving list decodable codes that can be encoded and list decoded (probabilistically) in nearly-linear time. These codes are presented in Section 7.1 below. We then show (in Section 7.2) how these codes can be used to probabilistically construct codes of rate up to 0.02 that can be uniquely decoded (probabilistically) up to half the Gilbert-Varshamov (GV) bound in nearly-linear time.

7.1 Nearly-linear time capacity achieving list decodable codes

Our nearly-linear time capacity achieving list decodable codes follow as a consequence of our efficient capacity achieving locally list decodable code construction (Theorem 6.2). Once more, we show a stronger version that applies also to list recovery.

Theorem 7.1 (Nearly-linear time capacity achieving list decodable / recoverable codes)**.** *There is a constant c so that the following holds. Choose $\rho \in [0,1]$, $\varepsilon > 0$, and a positive integer ℓ. Let $\{t_n\}_n$ be a sequence of positive integers, non-decreasing with n, so that t_0 is sufficiently large, and so that*

$$t_n \leq \sqrt{\frac{\varepsilon \log_q(n)}{c\ell}}.$$

Let

$$s_0 := \max\left\{c(1-\rho-\varepsilon/2)\log(1/(1-\rho-\varepsilon/2))/\varepsilon, c(\rho+\varepsilon/2)\log(1/(\rho+\varepsilon/2))/\varepsilon, c(\log \ell)/\varepsilon\right\},$$

and for each choice of t, suppose that $s = s(t)$ is such that $s \geq \max\left\{(t/\varepsilon)^{ct}, c\ell t/\varepsilon^2\right\} \cdot s_0/\rho$ is even.

Then there exists an infinite family of $\mathbb{F}_2$-linear codes $\{C_n\}_n$ such that the following holds. Below, to simplify notation we use t instead of t_n and s instead of $s(t_n)$.

1. *$C_n : \mathbb{F}_{2^s}^{\rho n} \to \mathbb{F}_{2^s}^n$ has rate ρ and relative distance at least $1-\rho-\varepsilon$.*
2. *C_n is $(1-\rho-\varepsilon, \ell, L)$-list recoverable for*

$$L = \exp\left(\exp\left(t^2 s \cdot 2^{O(s_0 \ell/\varepsilon)} \cdot \exp(\log^* n) + t \log s)\right)\right).$$

3. *C_n can be list recovered probabilistically (with success probability $2/3$)[4] in time*

$$T = n^{1+O(1/t)} \cdot 2^{O(\sqrt{\log n \cdot \log\log n})} \cdot \exp\left(\exp\left(t^2 s \cdot 2^{O(s_0 \ell/\varepsilon)} \cdot \exp(\log^* n) + t \log s\right)\right).$$

[4]More precisely, there exists a randomized algorithm A, running in time T, that outputs a list of L messages, and the guarantee is that with probability at least $2/3$ the output list contains all messages that correspond to close-by codewords. Note that the success probability can be amplified to $1-e^{-t}$, at the cost of increasing the output list size by a multiplicative factor of $O(t)$, by repeating the algorithm independently $O(t)$ times and returning the union of all output lists.

4. C_n can be encoded in time

$$n \cdot 2^{O(\sqrt{\log n \cdot \log\log n})} + t \cdot n^{1+O(1/t)} + O(n \cdot 2^{s^2}) + s^{O(1)} \cdot n \cdot \mathrm{polylog} n.$$

In particular,

- *When $\rho, \varepsilon, \ell, t, s$ are constant we get that $L = \exp(\exp(\exp(\log^* n)))$ and list recovery and encoding times are $n^{1+O(1/t)}$.*
- *Alternatively, when ρ, ε, ℓ are constant, $t = t_n = O\left(\frac{\log\log\log(n)}{(\log\log\log\log(n))^2)}\right)$ and $s = t^{O(t)}$ we get that $L = n^{o(1)}$ and list recovery and encoding times are $n^{1+O(1/t)} = n^{1+o(1)}$.*

Proof. We use the same codes as in Theorem 6.2 so it only remains to prove the second and third bullets.

The global list recovery algorithm A for C_n first runs the local list recovery algorithm A' for C_n guaranteed by Theorem 6.2 for $O(\log L')$ times independently where L' is the output list size given by Theorem 6.2. Let $A'_1, A'_2, \ldots, A'_{O(L' \log L')}$ denote the local algorithms in the output lists. Then for each A'_j the algorithm A includes in the output list a message $x^{(j)} \in \mathbb{F}_{2^s}^{\rho n}$ which results by applying A'_j on each of the ρn message coordinates $O(\log(nL'))$ times independently and taking majority vote for each of the coordinates.

We claim that with probability at least $2/3$ the list output by A includes all messages that correspond to close-by codewords. To see this note first that since any close-by codeword is represented in the output list of A' with probability at least $2/3$, it must hold that the number of close-by codewords is at most $3L'/2$. Consequently, running A' for $O(\log L')$ times guarantees that with probability at least $5/6$ all close-by codewords will be represented in the output list. Moreover, repeating the decoding of each message coordinate for $O(\log(nL'))$ times guarantees that each of these coordinates is decoded correctly with probability at least $1 - 1/(10nL')$, and so by union bound with probability at least $5/6$ all messages are decoded correctly. So we obtained that with probability at least $1-1/6-1/6 = 2/3$ the output list of A will include all messages that correspond to close-by codewords.

Finally note that the output list size of the algorithm A is

$$O(L' \log L') = \exp\left(\exp\left(t^2 s \cdot 2^{O(s_0 \ell/\varepsilon)} \cdot \exp(\log^* n) + t \log s\right)\right),$$

and its running time is

$$\begin{aligned} &O(T'_{\mathrm{pre}} \cdot \log L') + O(L' \log L' \cdot n \log(nL') \cdot T') \\ &= n^{1+O(1/t)} \cdot 2^{O(\sqrt{\log n \cdot \log\log n})} \cdot \exp\left(\exp\left(t^2 s \cdot 2^{O(s_0 \ell/\varepsilon)} \cdot \exp(\log^* n) + t \log s\right)\right), \end{aligned}$$

where T'_{pre}, T' denote the preprocessing and running times of the local list recovery algorithm for C_n, respectively. □

7.2 Nearly-linear time decoding up to half the GV bound

Next we show how the nearly-linear time capacity achieving list recoverable codes of Theorem 7.1 can be used to obtain codes of rate up to 0.02 that are uniquely decodable up to half the Gilbert-Varshamov (GV) bound in nearly-linear time. Let $H_2^{-1} : [0,1] \to [0, \frac{1}{2}]$ be the inverse of the binary entropy function H_2 in the domain $[0, \frac{1}{2}]$.

Theorem 7.2 (Nearly-linear time decoding up to half GV bound)**.** *Choose constants $\rho \in [0, 0.02]$ and $\varepsilon > 0$. Then there exists a randomized polynomial time algorithm which for infinitely many n, given an input string 1^n, outputs a description of a code C_n that satisfies the following properties with probability at least $1 - \exp(-n)$.*[5]

1. *$C_n : \mathbb{F}_2^{\rho n} \to \mathbb{F}_2^n$ is a linear code of rate ρ and relative distance at least $\delta := H_2^{-1}(1-\rho) - \varepsilon$.*
2. *C_n can be uniquely decoded probabilistically (with success probability $2/3$) from $\delta/2$ fraction of errors in time $n^{1+O(1/t)} = n^{1+o(1)}$ for $t = O\left(\frac{\log\log\log(n)}{(\log\log\log\log(n))^2}\right)$.*
3. *C_n can be encoded in time $n^{1+O(1/t)} = n^{1+o(1)}$ with the same choice of t.*

To prove the above theorem we rely on the following lemma which says that one can turn a code that approximately satisfies the Singleton bound into one that approximately satisfies the GV bound via random concatenation. The proof is similar to that of Thommesen [Tho83]. In what follows let $\theta(x) := 1 - H_2(1 - 2^{x-1})$ for $x \in [0, 1]$.

Lemma 7.3. *There is a constant c so that the following holds. Choose $\varepsilon > 0$, $\rho_{in} \in [0, 1]$, and $\rho_{out} \in \left[0, \frac{\theta(\rho_{in}) - \varepsilon/2}{\rho_{in}}\right]$. Suppose that $s \geq \frac{c \cdot \rho_{in}}{\varepsilon^2 \cdot (1-\rho_{out})}$. Then the following holds for sufficiently large n. Let $C_{out} : \mathbb{F}_{2^s}^{\rho_{out} \cdot n} \to \mathbb{F}_{2^s}^n$ be an $\mathbb{F}_2$-linear code of rate ρ_{out} and relative distance at least $1 - \rho_{out} - \frac{\varepsilon^2}{c}$. Let $C : \mathbb{F}_2^{s \cdot \rho_{out} \cdot n} \to \mathbb{F}_2^{sn/\rho_{in}}$ be a code obtained from C_{out} by applying a random linear code $C^{(i)} : \mathbb{F}_2^s \to \mathbb{F}_2^{s/\rho_{in}}$ on each coordinate $i \in [n]$ of C_{out} independently (where we identify the field $\mathbb{F}_{2^s}$ with the vector space $\mathbb{F}_2^s$ via the usual $\mathbb{F}_2$-linear transformation). Then C has relative distance at least $H_2^{-1}(1 - \rho_{out} \cdot \rho_{in}) - \varepsilon$ with probability at least $1 - exp(-n)$.*

We shall also use the following lemma that states the effect of concatenation on list recovery properties.

Lemma 7.4. *Let $C_{out} : \mathbb{F}_{q^s}^{\rho_{out} \cdot n} \to \mathbb{F}_{q^s}^n$ be an $(\alpha_{out}, \ell_{out}, L_{out})$-list recoverable code, with a list recovery algorithm running in time T_{out}. Let $C : \mathbb{F}_q^{s \cdot \rho_{out} \cdot n} \to \mathbb{F}_q^{sn/\rho_{in}}$ be the code obtained from C_{out} by applying a code $C^{(i)} : \mathbb{F}_q^s \to \mathbb{F}_q^{s/\rho_{in}}$ on each coordinate $i \in [n]$ of C_{out}. Suppose furthermore that at least $1 - \varepsilon$ fraction of the codes $C^{(i)}$ are $(\alpha_{in}, \ell_{in}, L_{in})$-list recoverable for $L_{in} = \ell_{out}$, with a list recovery algorithm running in time T_{in}. Then C is $((\alpha_{out} - \varepsilon) \cdot \alpha_{in}, \ell_{in}, L_{out})$-list recoverable in time $T_{out} + O(n \cdot T_{in})$.*

Before we prove the above pair of lemmas we show how they imply Theorem 7.2.

Proof of Theorem 7.2. Let c be a sufficiently large constant for which both Theorem 7.1 and Lemma 7.3 hold. Let $\rho_{in} := \theta^{-1}(\rho + \varepsilon/2)$ and $\rho_{out} := \frac{\rho}{\rho_{in}} = \frac{\rho}{\theta^{-1}(\rho+\varepsilon/2)}$. Define

$$t = t_n = O\left(\frac{\log\log\log(n)}{(\log\log\log\log(n))^2}\right)$$

[5]The randomized algorithm can output different codes under different random choices, we are only guaranteed that the output code has the required properties with high probability. Also the encoding and decoding algorithms need access to the random choices made during the construction of the code.

as in the statement of Theorem 7.2, and let

$$s = \max\left\{(ct/\varepsilon^2)^{ct}, c^3 2^{1/\varepsilon} t/\varepsilon^4\right\} \cdot s_0/(\rho_{out}(1-\rho_{out}))$$

for even

$$s_0 := \max\left\{\frac{c^2}{\varepsilon^2}\left(1-\rho_{out}-\frac{\varepsilon^2}{2c}\right)\log\left(\frac{1}{1-\rho_{out}-\varepsilon^2/(2c)}\right), \right.$$
$$\left. \frac{c^2}{\varepsilon^2}\left(\rho_{out}+\frac{\varepsilon^2}{2c}\right)\log\left(\frac{1}{\rho_{out}+\varepsilon^2/(2c)}\right), \frac{c^2}{\varepsilon^3}\right\}.$$

Notice that chosing when $t = t_n$ is super-constant, as above, and ε, ρ are constant (as in the theorem statement) that we always have $s = t^{O(t)}$.

Choose any $n \in \mathbb{N}$ so that Theorem 7.1 guarantees that codes of block length n exist (there are infinitely many of these). Then the code C_n is constructed as follows.

Let $C_{out} : \mathbb{F}_{2^s}^{\rho_{out}\cdot(\rho_{in}n/s)} \to \mathbb{F}_{2^s}^{\rho_{in}n/s}$ be the $\mathbb{F}_2$-linear code of rate ρ_{out} and relative distance at least $1-\rho_{out}-\frac{\varepsilon^2}{c}$ that is $\left(1-\rho_{out}-\frac{\varepsilon^2}{c}, 2^{1/\varepsilon}, n^{O(1/t)}\right)$-list recoverable in time $n^{1+O(1/t)}$ whose existence is guaranteed by Theorem 7.1. Notice that here t is growing with n, but slowly enough that we may apply Theorem 7.1.

Let $C_n : \mathbb{F}_2^{\rho_{out}\cdot\rho_{in}n} \to \mathbb{F}_2^n$ be a binary linear code of rate $\rho_{in}\cdot\rho_{out} = \rho$ obtained from C_{out} by applying a random linear code $C^{(i)} : \mathbb{F}_2^s \to \mathbb{F}_2^{s/\rho_{in}}$ on each coordinate $i \in [\rho_{in}n/s]$ of C_{out} independently.

Then by Lemma 7.3 the code C_n has relative distance at least $H_2^{-1}(1-\rho)-\varepsilon$ with probability at least $1-\exp(-n)$. Moreover, by Theorem 4.2 we also have that each $C^{(i)}$ is $(H_2^{-1}(1-\rho_{in}-\varepsilon), 2^{1/\varepsilon})$-list decodable with probability at least $1-\exp(-s) = 1-o(1)$, so with probability at least $1-\exp(-n)$ it holds that at least $1-\varepsilon^2/c$ fraction of the $C^{(i)}$'s are $(H_2^{-1}(1-\rho_{in}-\varepsilon), 2^{1/\varepsilon})$-list decodable. Lemma 7.4 then implies that in this case the code C_n is $((1-\rho_{out}-2\varepsilon^2/c)\cdot H_2^{-1}(1-\rho_{in}-\varepsilon), n^{O(1/t)})$-list decodable (probabilistically) in time $n^{1+O(1/t)} + n\cdot 2^{O(s)} = n^{1+O(1/t)}$.

Now observe that by Theorem 7.1 the code C_{out} is encodable in time $n^{1+O(1/t)}$, and so encoding time of C_n is $n^{1+O(1/t)} + n\cdot 2^{O(s^2)} = n^{1+O(1/t)}$. Consequently, whenever the list decoding radius of C_n exceeds half the minimum distance, one can uniquely decode C_n up to half the minimum distance in time $n^{1+O(1/t)}$ by list decoding C_n, computing the codewords that correspond to the messages in the output list, and returning the (unique) closest codeword.

So we obtained that the code C_n is uniquely decodable up to half the minimum distance in time $n^{1+O(1/t)}$ whenever

$$\left(1-\rho_{out}-2\varepsilon^2/c\right)\cdot H_2^{-1}(1-\rho_{in}-\varepsilon) \geq \frac{H_2^{-1}(1-\rho)-\varepsilon}{2}.$$

Finally, it is shown in [Rud07], Section 4.4 that the inequality above holds for sufficiently small constant $\varepsilon > 0$ by choice of $\rho = \rho_{in}\cdot\rho_{out} \leq 0.02$. □

It remains to prove Lemmas 7.3 and 7.4. We start with Lemma 7.3.

Proof of Lemma 7.3. The proof follows the arguments of [Tho83]. For a string x of length n let the *relative weight* $\mathrm{wt}(x)$ of x denote the fraction of non-zero coordinates of x. It is well known that the relative distance of an $\mathbb{F}$-linear code equals the minimum relative weight $\mathrm{wt}(c)$ of a non-zero codeword $c \in C$.

Fix a codeword $c' \in C_{out}$ with $\mathrm{wt}(c') = \gamma \geq 1 - \rho_{out} - \varepsilon^2/c$, and let $c \in \mathbb{F}_2^{sn/\rho_{in}}$ be a word obtained from c' by applying a random linear code $C^{(i)} : \mathbb{F}_2^s \to \mathbb{F}_2^{s/\rho_{in}}$ on each coordinate $i \in [n]$ of c' independently. Then for each non-zero coordinate i of c' it holds that the i-th block of c of length s/ρ_{in} is distributed uniformly over $\mathbb{F}_2^{s/\rho_{in}}$, and so $\gamma sn/\rho_{in}$ coordinates of c are uniformly distributed (while the rest equal zero). Consequently, we have that

$$\Pr[\mathrm{wt}(c) < \delta] \leq \binom{\gamma sn/\rho_{in}}{\leq \delta sn/\rho_{in}} 2^{-\gamma sn/\rho_{in}} \leq 2^{H_2(\delta/\gamma)\gamma sn/\rho_{in}} \cdot 2^{-\gamma sn/\rho_{in}}.$$

Next we apply a union bound over all codewords $c' \in C_{out}$. For this fix $\gamma > 0$ such that $\gamma \geq 1 - \rho_{out} - \varepsilon^2/c$ and $\gamma n \in \mathbb{N}$. Then it holds that the number of codewords in C_{out} of relative weight γ is at most

$$\binom{n}{\gamma n} \cdot (2^s)^{\gamma n - (1-\rho_{out}-\varepsilon^2/c)n} \leq 2^n \cdot 2^{(\gamma - (1-\rho_{out}-\varepsilon^2/c))sn},$$

where the above bound follows since there are at most $\binom{n}{\gamma n}$ choices for the location of the non-zero coordinates, and for any such choice fixing the value of the first $\gamma n - (1 - \rho_{out} - \varepsilon^2/c)n$ non-zero coordinates determines the value of the rest of the non-zero coordinates (since two different codewords cannot differ on less than $(1 - \rho_{out} - \varepsilon^2/c)n$ coordinates).

Consequently, we have that

$$\Pr[\mathrm{dist}(C) < \delta] \leq \sum_{1-\rho_{out}-\varepsilon^2/c \leq \gamma \leq 1,\ \gamma n \in \mathbb{N}} 2^n \cdot 2^{(\gamma-(1-\rho_{out}-\varepsilon^2/c))sn} \cdot 2^{H_2(\delta/\gamma)\gamma sn/\rho_{in}} \cdot 2^{-\gamma sn/\rho_{in}}$$

$$= \sum_{1-\rho_{out}-\varepsilon^2/c \leq \gamma \leq 1,\ \gamma n \in \mathbb{N}} \exp\left[-\gamma sn/\rho_{in}\left(1 - \rho_{in} \cdot \left(1 - \frac{1-\rho_{out}-\varepsilon^2/c}{\gamma}\right) - \frac{\rho_{in}}{\gamma s} - H_2\left(\frac{\delta}{\gamma}\right)\right)\right].$$

Finally, Lemma 8 in [GR08a] implies that in our setting of parameters, and for choice of $\delta := H_2^{-1}(1 - \rho_{in} \cdot \rho_{out}) - \varepsilon$, the right hand side of the above inequality is at most $\exp(-n)$, which completes the proof of the lemma. The running time follows. □

Next we prove Lemma 7.4.

Proof of Lemma 7.4. For $i \in [n]$ and $j \in [s/\rho_{in}]$, let $S_{i,j} \subseteq \mathbb{F}_q$ be a list of at most ℓ_{in} possible symbols for the coordinate $C(x)_{i,j} := C(x)_{(i-1)\cdot(s/\rho_{in})+j}$, which is the j-th coordinate of $C^{(i)}(C_{out}(x)_i)$.

Suppose that for at most a $(\alpha_{out} - \varepsilon) \cdot \alpha_{in}$ fraction of coordinates (i, j), $C(x)_{i,j} \notin S_{i,j}$. Then by Markov's inequality, for at most a $\alpha_{out} - \varepsilon$ fraction of $i \in [n]$, the blocks $C^{(i)}(C_{out}(x)_i)$ have more than α_{in} fraction of the $j \in [s/\rho_{in}]$ so that $C(x)_{i,j} \notin S_{i,j}$. Thus, we may list recover each block $C^{(i)}(C_{out}(x)_i)$ which is list recoverable to obtain a list $S_i \subseteq \mathbb{F}_{q^s}$ of at most $L_{in} = \ell_{out}$ possible symbols for $C_{out}(x)_i$, and the above reasoning shows that $C_{out}(x)_i \notin S_i$ for at most $\alpha_{out} n$ values of i. Now we may run C_{out}'s list recovery algorithm to obtain a final list of size L_{out}. □

Acknowledgements. The second author would like to thank Swastik Kopparty for raising the question of obtaining capacity achieving locally list decodable codes which was a direct trigger for this work as well as previous work [KMRS16, GKO$^+$17], and Sivakanth Gopi, Swastik Kopparty, Rafael Oliveira and Shubhangi Saraf for many discussions on this topics. The current collaboration began during the Algorithmic Coding Theory Workshop at ICERM, we thank ICERM for their hospitality.

References

[AEL95] Noga Alon, Jeff Edmonds, and Michael Luby. Linear time erasure codes with nearly optimal recovery. In *proceedings of the 36th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 512–519. IEEE Computer Society, 1995.

[Ale02] Michael Alekhnovich. Linear diophantine equations over polynomials and soft decoding of reed-solomon codes. In *Foundations of Computer Science, 2002. Proceedings. The 43rd Annual IEEE Symposium on*, pages 439–448. IEEE, 2002.

[BB10] Peter Beelen and Kristian Brander. Key equations for list decoding of Reed Solomon codes and how to solve them. *Journal of Symbolic Computation*, 45(7):773–786, 2010.

[BHNW13] Peter Beelen, Tom Hoholdt, Johan S.R. Nielsen, and Yingquan Wu. On rational interpolation-based list-decoding and list-decoding binary goppa codes. *Information Theory, IEEE Transactions on*, 59(6):3269–3281, 2013.

[BS06] Eli Ben-Sasson and Madhu Sudan. Robust locally testable codes and products of codes. *Random Struct. Algorithms*, 28(4):387–402, 2006.

[BV09] Eli Ben-Sasson and Michael Viderman. Tensor products of weakly smooth codes are robust. *Theory of Computing*, 5(1):239–255, 2009.

[BV15] Eli Ben-Sasson and Michael Viderman. Composition of semi-LTCs by two-wise tensor products. *Computational Complexity*, 24(3):601–643, 2015.

[CH11] Henry Cohn and Nadia Heninger. Ideal forms of Coppersmith's theorem and Guruswami-Sudan list decoding. In *ICS*, pages 298–308, 2011.

[CJN+15] Muhammad F.I. Chowdhury, Claude-Pierre Jeannerod, Vincent Neiger, Eric Schost, and Gilles Villard. Faster algorithms for multivariate interpolation with multiplicities and simultaneous polynomial approximations. *Information Theory, IEEE Transactions on*, 61(5):2370–2387, 2015.

[CR05] Don Coppersmith and Atri Rudra. On the robust testability of tensor products of codes. ECCC TR05-104, 2005.

[DL12] Zeev Dvir and Shachar Lovett. Subspace Evasive Sets. In *STOC '12*, pages 351–358, October 2012.

[DSW06] Irit Dinur, Madhu Sudan, and Avi Wigderson. Robust local testability of tensor products of LDPC codes. In *proceedings of the 9th International Workshop on Randomization and Computation (RANDOM)*, pages 304–315. Springer, 2006.

[Eli57] Peter Elias. List decoding for noisy channels. Technical Report 335, MIT, September 1957.

[GGR11] Parikshit Gopalan, Venkatesan Guruswami, and Prasad Raghavendra. List decoding tensor products and interleaved codes. *SIAM Journal on Computing*, 40(5):1432–1462, 2011.

[GI01] Venkatesan Guruswami and Piotr Indyk. Expander-based constructions of efficiently decodable codes. In *Foundations of Computer Science, 2001. Proceedings. 42nd IEEE Symposium on*, pages 658–667. IEEE, October 2001.

[GI02] Venkatesan Guruswami and Piotr Indyk. Near-optimal linear-time codes for unique decoding and new list-decodable codes over smaller alphabets. In *Proceedings of the Thiry-fourth Annual ACM Symposium on Theory of Computing*, STOC '02, pages 812–821, New York, NY, USA, 2002. ACM.

[GI03] Venkatesan Guruswami and Piotr Indyk. Linear time encodable and list decodable codes. In *Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, STOC '03, pages 126–135, New York, NY, USA, 2003. ACM.

[GI04] Venkatesan Guruswami and Piotr Indyk. Efficiently decodable codes meeting Gilbert-Varshamov bound for low rates. In *SODA '04: Proceedings of the fifteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 756–757, Philadelphia, PA, USA, 2004. Society for Industrial and Applied Mathematics.

[Gil52] Edgar N. Gilbert. A comparision of signalling alphabets. *Bell System Technical Journal*, 31:504–522, 1952.

[GK16a] Alan Guo and Swastik Kopparty. List-decoding algorithms for lifted codes. *IEEE Transactions on Information Theory*, 62(5):2719 – 2725, 2016.

[GK16b] Venkatesan Guruswami and Swastik Kopparty. Explicit subspace designs. *Combinatorica*, 36(2):161–185, 2016.

[GKO+16] Sivakanth Gopi, Swastik Kopparty, Rafael Oliveira, Noga Ron-Zewi, and Shubhangi Saraf. Locally testable and locally correctable codes approaching the gilbert-varshamov bound. *Electronic Colloquium on Computational Complexity (ECCC)*, 23:122, 2016.

[GKO+17] Sivakanth Gopi, Swastik Kopparty, Rafael Oliveira, Noga Ron-Zewi, and Shubhangi Saraf. Locally testable and locally correctable codes approaching the gilbert-varshamov bound. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2073–2091. SIAM, 2017.

[GL89] Oded Goldreich and Leonid A Levin. A hard-core predicate for all one-way functions. In *Proceedings of the twenty-first annual ACM symposium on Theory of computing*, pages 25–32. ACM, 1989.

[GM12] Oded Goldreich and Or Meir. The tensor product of two good codes is not necessarily locally testable. *Inf. Proces. Lett.*, 112(8-9):351–355, 2012.

[GNP+13] Anna C. Gilbert, Hung Q. Ngo, Ely Porat, Atri Rudra, and Martin J. Strauss. ℓ_2/ℓ_2-foreach sparse recovery with low risk. In *Automata, Languages, and Programming*, volume 7965 of *Lecture Notes in Computer Science*, pages 461–472. Springer Berlin Heidelberg, 2013.

[GR08a] Venkatesan Guruswami and Atri Rudra. Concatenated Codes Can Achieve List-decoding Capacity. In *Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '08, pages 258–267, Philadelphia, PA, USA, 2008. Society for Industrial and Applied Mathematics.

[GR08b] Venkatesan Guruswami and Atri Rudra. Explicit codes achieving list decoding capacity: Error-correction with optimal redundancy. *IEEE Transactions on Information Theory*, 54(1):135–150, 2008.

[GS99] Venkatesan Guruswami and Madhu Sudan. Improved decoding of Reed-Solomon and algebraic-geometry codes. *IEEE Transactions on Information Theory*, 45(6), 1999.

[Gur01] Venkatesan Guruswami. *List decoding of error-correcting codes*. PhD thesis, MIT, 2001.

[Gur10] Venkatesan Guruswami. Cyclotomic function fields, artin–frobenius automorphisms, and list error correction with optimal rate. *Algebra & Number Theory*, 4(4):433–463, 2010.

[Gur11] Venkatesan Guruswami. Linear-algebraic list decoding of folded Reed-Solomon codes. In *IEEE Conference on Computational Complexity*, pages 77–85, 2011.

[GW11] Venkatesan Guruswami and Carol Wang. Optimal rate list decoding via derivative codes. In *RANDOM '11*, 2011.

[GX12a] Venkatesan Guruswami and Chaoping Xing. Folded codes from function field towers and improved optimal rate list decoding. In *Proceedings of the 44th annual ACM symposium on Theory of computing (STOC)*, pages 339–350. ACM, 2012.

[GX12b] Venkatesan Guruswami and Chaoping Xing. List decoding reed-solomon, algebraic-geometric, and gabidulin subcodes up to the singleton bound. *Electronic Colloquium on Computational Complexity (ECCC)*, 2012.

[GX13] Venkatesan Guruswami and Chaoping Xing. List decoding reed-solomon, algebraic-geometric, and gabidulin subcodes up to the singleton bound. In *Proceedings of the 45th annual ACM symposium on Theory of Computing (STOC)*, pages 843–852. ACM, 2013.

[GX14] Venkatesan Guruswami and Chaoping Xing. Optimal rate list decoding of folded algebraic-geometric codes over constant-sized alphabets. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1858–1866. SIAM, 2014.

[HIOS15] Iftach Haitner, Yuval Ishai, Eran Omri, and Ronen Shaltiel. Parallel hashing via list recoverability. In Rosario Gennaro and Matthew Robshaw, editors, *CRYPTO*, volume 9216 of *Lecture Notes in Computer Science*, pages 173–190. Springer, 2015.

[HOW15] Brett Hemenway, Rafail Ostrovsky, and Mary Wootters. Local correctability of expander codes. *Inf. Comput*, 243:178–190, 2015.

[HW15] Brett Hemenway and Mary Wootters. Linear-time list recovery of high-rate expander codes. In *ICALP '15*, volume 9134, pages 701–712, 2015.

[INR10] Piotr Indyk, Hung Q. Ngo, and Atri Rudra. Efficiently decodable non-adaptive group testing. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '10, pages 1126–1142, Philadelphia, PA, USA, 2010. Society for Industrial and Applied Mathematics.

[KM93] Eyal Kushilevitz and Yishay Mansour. Learning decision trees using the fourier spectrum. *SIAM Journal on Computing*, 22(6):1331–1348, 1993.

[KMRS15] Swastik Kopparty, Or Meir, Noga Ron-Zewi, and Shubhangi Saraf. High-rate locally-correctable and locally-testable codes with sub-polynomial query complexity. *Electronic Colloquium on Computational Complexity (ECCC)*, 2015.

[KMRS16] Swastik Kopparty, Or Meir, Noga Ron-Zewi, and Shubhangi Saraf. High rate locally-correctable and locally-testable codes with sub-polynomial query complexity. In *Journal of the ACM, to appear. Preliminary version in proceedings of the 48th Annual Symposium on Theory of Computing (STOC)*, pages 25–32. ACM Press, 2016.

[Kop15] Swastik Kopparty. List-decoding multiplicity codes. *Theory OF Computing*, 11(5):149–182, 2015.

[Mei09] Or Meir. Combinatorial construction of locally testable codes. *SIAM J. Comput.*, 39(2):491–544, 2009.

[NPR12] Hung Q. Ngo, Ely Porat, and Atri Rudra. Efficiently Decodable Compressed Sensing by List-Recoverable Codes and Recursion. In Christoph Dürr and Thomas Wilke, editors, *29th International Symposium on Theoretical Aspects of Computer Science (STACS 2012)*, volume 14 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 230–241, Dagstuhl, Germany, 2012. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[Rud07] Atri Rudra. *List Decoding and Property Testing of Error Correcting Codes.* PhD thesis, University of Washington, 2007.

[Sti09] Henning Stichtenoth. *Algebraic function fields and codes*, volume 254. Springer Science & Business Media, 2009.

[STV01] Madhu Sudan, Luca Trevisan, and Salil Vadhan. Pseudorandom generators without the XOR lemma. *Journal of Computer and System Sciences*, 62(2):236–266, 2001.

[Sud01] Madhu Sudan. Algorithmic introduction to coding theory (lecture notes), 2001.

[Tho83] Christian Thommesen. The existence of binary linear concatenated codes with reed - solomon outer codes which asymptotically meet the gilbert- varshamov bound. *IEEE Trans. Information Theory*, 29(6):850–853, 1983.

[Val05] Paul Valiant. The tensor product of two codes is not necessarily robustly testable. In *proceedings of the 9th International Workshop on Randomization and Computation (RANDOM)*, pages 472–481. Springer, 2005.

[Var57] R. R. Varshamov. Estimate of the number of signals in error correcting codes. *Doklady Akadamii Nauk*, pages 739–741, 1957.

[Vid11] Michael Viderman. A combination of testability and decodability by tensor products. *Electronic Colloquium on Computational Complexity (ECCC)*, 18:87, 2011.

[Vid13] Michael Viderman. Strong LTCs with inverse poly-log rate and constant soundness. In *proceedings of the 54th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 330–339. IEEE Computer Society, 2013.

[Woz58] J.M. Wozencraft. List decoding. Quarterly progress report, MIT, 1958.

A List-recovery of algebraic geometry codes

In this appendix, we outline how the approach of [GX13] needs to be changed in order to obtain linear list-recoverable codes. The main theorem is as follows.

Theorem A.1. *There are constants c, c_0 so that the following holds. Choose $\varepsilon > 0$ and a positive integer ℓ. Suppose that $q \geq \ell^{c/\varepsilon}$ is an even power of a prime. Let $N_0 = q^{c_0 \ell/\varepsilon}$.*

Then for all $N \geq N_0$, there is a deterministic polynomial-time construction of an $\mathbb{F}_q$-linear code $C : \mathbb{F}_q^{(1-\varepsilon)N} \to \mathbb{F}_q^N$ of rate $1-\varepsilon$ and relative distance $\Omega(\varepsilon^2)$ which is $(\Omega(\varepsilon^2), \ell, L)$-list-recoverable in time $\mathrm{poly}(N, L)$*, returning a list that is contained in a subspace over $\mathbb{F}_q$ of dimension at most*

$$\left(\frac{q^{c\ell/\varepsilon}}{\varepsilon}\right)^{2^{\log^*(N)}}.$$

In particular, when ε, ℓ, q are constant, the ouput list size L is $\exp(\exp(\exp(\log^* N)))$.

We remark that the list size is very slowly growing (although admittedly with extremely large constants).

We follow the approach of [GX13, GK16b]. In [GX13], Guruswami and Xing show how to construct high-rate list-decodable codes over a constant alphabet, modulo a construction of *explicit subspace designs*. In [GK16b], Guruswami and Kopparty gave such constructions and used them to construct high-rate list-decodable codes over constant-sized alphabets with small list-sizes. We would like to use these codes here. However, there are two things which must be modified. First, the guarantees of [GX13, GK16b] are for list-decodability, and we are after list-recoverability. Fortunately, this follows from a standard modification of the techniques that they use. Second, the codes that they obtain are not linear, but rather are linear over a subfield of the alphabet. To correct this, we concatenate these codes with list-recoverable linear codes of a constant length. A random linear code has this property, and since we only require them to be of constant length, we may find such a code, and run list-recovery algorithms on it, in constant time.

We begin by addressing the leap from list-decodability to list-recovery, and then discuss the code concatenation step. We refer the reader to [GX13, GK16b] for the details (and, indeed, for several definitions); here we just outline the parts which are important for list-recovery. The basic outline of the construction (and the argument) is as follows:

Step 1. Show that AG codes are list-decodable, with large but very structured lists. We will extend this to list-recoverability with structured lists.

Step 2. Show that one can efficiently find a subcode of the AG code which will avoid this sort of structure: this reduces the list size. This part of the argument goes through unchanged, and will yield a list-recoverable code over $\mathbb{F}_{q^m}$ with small list size.

Once we have $\mathbb{F}_q$-linear codes over $\mathbb{F}_q^m$ that are list-recoverable, we discuss the third step:

Step 3. The code produced is $\mathbb{F}_q$-linear (rather than $\mathbb{F}_{q^m}$-linear). This was fine for [GX13, GK16b], but for us we require a code which is linear over the alphabet it is defined over. To get around this we concatenate the codes above with a random linear code of length m over $\mathbb{F}_q$. This will result in an $\mathbb{F}_q$-linear code over $\mathbb{F}_q$ that is list-recoverable with small list sizes.

We briefly go through the details. First we give a short refresher/introduction to the notation. Then we handle the three steps above, in order. We note that throughout this appendix we will refer to Theorem and Lemma numbers in the extended version [GX12b] rather than the conference version [GX13].

Step 0. Algebraic Geometry Codes and basic notation. Since we do not need to open up the AG code machinery very much in order to extend the results of [GX13] to list-recovery, we do not go into great detail here, and we refer the reader to [GX13] and the references therein for the technical details, and to [Sti09] for a comprehensive treatment of AG codes. However, for the ease of exposition here (for the reader unfamiliar with AG codes), we will introduce some notation and explain the intuitive definitions of these notions. In particular, we will use the running example of a rational function field. We stress that this is *not* the final function field used; thus the intuition should be taken as intuition only.

Let $F/\mathbb{F}_q$ be a function field of genus g. One example (which may be helpful to keep in mind) of a genus-0 function field is the rational function field $\mathbb{F}_q(X)/\mathbb{F}_q$, which may be thought of as rational functions $f(X)/g(X)$, where $f, g \in \mathbb{F}_q[X]$ are irreducible polynomials. For the code construction, we will use a function field of larger genus (given by the Garcia-Stichtenoth tower, as in [GX13]), but we will use this example to intuitively define the algebraic objects that we need.

Let $P_\infty, P_1, \ldots, P_n$ be $n+1$ distinct $\mathbb{F}_q$-rational places (that is, of degree 1). Formally, these are ideals, but they are in one-to-one correspondence with $\mathbb{F}_q \cup \{\infty\}$, and let us think of them that way. For each such place P, there is a map (the *residue class map* with respect to P) which maps $F/\mathbb{F}_q$ to $\mathbb{F}_q$; we may think of this as function evaluation, and in our example of $\mathbb{F}_q(X)/\mathbb{F}_q$, if P is a place associated with a point $\alpha \in \mathbb{F}_q$, then indeed this maps $f(X)/g(X)$ to $f(\alpha)/g(\alpha)$.

Let $\mathcal{L}(lP_\infty)$ be the Riemann-Roch space over $\mathbb{F}_q$. Formally, this is

$$\mathcal{L}(lP_\infty) = \{h \in F \setminus \{0\} : \nu_{P_\infty}(h) \geq -l\} \cup \{0\},$$

where ν_{P_∞} is the discrete valuation of P_∞. Informally (in our running example), this should be thought of as the set of rational functions $f(X)/g(X)$ so that $\deg(g(X)) - \deg(f(X)) \geq -l$. In particular, the number of poles of f/g is at least the number of roots, minus l. It would be tempting, in this example, to think of these as degree $\leq l$ polynomials; all but at most l of the powers of X in the numerator are "canceled" in the denominator. Of course, there are many problems with this intuition, but it turns out that this indeed works out in some sense. In particular, it can be shown that the dimension of this space is at least $l - g + 1$. When $g = 0$ (as in our running example), it is exactly $l + 1$, the same as the dimension of the space of degree-$\leq l$ polynomials.

More generally (whatever the genus), for any rational place P, we may write a function $h \in \mathcal{L}_m(lP_\infty)$ as

$$h = \sum_{j=0}^{\infty} h_j T^j, \tag{2}$$

where T is a local parameter of P, and it turns out that h is uniquely determined by the first $l+1$ coefficients $h_0, h_1, \ldots, h_{l+1}$.

Now let F_m be the constant extension $\mathbb{F}_{q^m} \cdot F$, and let $\mathcal{L}_m(lP_\infty)$ be the corresponding Riemann-Roch space. This has the same dimension over $\mathbb{F}_{q^m}$ as $\mathcal{L}(lP_\infty)$ does over $\mathbb{F}_q$. Now we consider the algebraic geometry code defined by

$$C(m; l) := \{(h(P_1), \ldots, h(P_n)) \,:\, h \in \mathcal{L}_m(lP_\infty)\}.$$

Following the intuition that $h(P_i)$ denotes function evaluation, this definition looks syntactically the same as a standard polynomial evaluation code, and should be thought of that way. This is an $\mathbb{F}_{q^m}$-linear code over $\mathbb{F}_{q^m}$, with block length n and dimension at least $l - g + 1$.

Step 1. List-decoding with structured lists to list-recovery with structured lists. With the preliminaries (and some basic, if possibly misleading, intuition for the reader unfamiliar with AG codes) out of the way, we press on with the argument.

Fix a parameter k, and consider a general AG code $C(m; k+2g-1)$, with the notation above. (We will fix a particular AG (sub)code later, by choosing a function field and by choosing a subcode). Let $S_1, \ldots, S_n \subset \mathbb{F}_{q^m}$ be lists of size at most ℓ corresponding to each coordinate. We first show that $C(m; k + 2g - 1)$ is $(1 - \alpha, \ell, L)$-list-recoverable for some α to be chosen below, where the list size is very large, but the list is structured. In [GX13], the approach (similar to that in [Gur11] or [GW11]) is as follows.

1. We will first find a low-degree interpolating linear polynomial (whose coefficients live in Riemann-Roch spaces)

   $$Q(Y_1, \ldots, Y_s) = A_0 + A_1 Y + \cdots + A_s Y_s$$

   so that $A_i \in \mathcal{L}_m(DP_\infty)$ and $A_0 \in \mathcal{L}_m((D+k+2g-1)P_\infty)$, for some parameter k to be chosen later, for

   $$D = \lfloor \frac{\ell n - k + (s-1)g + 1}{s+1} \rfloor,$$

   and subject to ℓn linear constraints over $\mathbb{F}_{q^m}$. Before we list the constraints, notice that the number of degrees of freedom in Q is

   $$s(D - g + 1) + D + k + g,$$

   because the $\mathbb{F}_{q^m}$-dimension of $\mathcal{L}_m((D + k + 2g - 1)P_\infty)$ is at least $D + k + g$, and the $\mathbb{F}_{q^m}$-dimension of $\mathcal{L}_m((DP_\infty))$ is at least $D-g+1$. Thus, the choice of D shows that the dimension of this space of interpolating polynomials is greater than ℓn. Thus, we will be able to find such a Q that satisfies the ℓn following ℓn constraints. For each $i \in [n]$ and for all $y \in S_i$, we have the constraint that

   $$A_0(P_i) + A_1(P_i)y + A_2(P_i)y^q + \cdots + A_s(P_i)y^{q^{s-1}} = 0.$$

2. With this polynomial Q in hand, we observe that if $h \in \mathcal{L}_m((k+2g-1)P_\infty)$ whose encoding has $h(P_i) \in S_i$ for at least αn positions i, for $\alpha n > D+k+2g-1$, then $Q(h, h^\sigma, \ldots, h^{\sigma^{s-1}}) = 0$, where h^σ denotes the extension of the Frobenius automorphism $\alpha \mapsto \alpha^q$ on $\mathbb{F}_{q^m}$ to $\mathcal{L}_m(lP_\infty)$. This proof (Lemma 4.7 in [GX12b]) remains unchanged when we pass to list-recovery from list-decoding. Briefly, this agreement means that

$$Q(h, \ldots, h^{\sigma^{s-1}})(P_i) = A_0(P_i) + A_1(P_i)h(P_i) + \cdots + A_s(P_i)h(P_i)^{q^{s-1}} = 0$$

for at least αn values of i, and so the function $Q(h, h^\sigma, \ldots, h^{\sigma^{s-1}})$ (which lies in $\mathcal{L}_m((D + k + 2g - 1)P_\infty)$; as per the intuition above, we are thinking of these as roughly analogous to degree-$(D + k + 2g - 1)$ polynomials) has at least $\alpha n \geq D + k + 2g - 1$ roots, and hence is the zero function.

3. Thus, any element $h \in \mathcal{L}_m((k + 2g - 1)P_\infty)$ that agrees with at least αn lists also satisfies $Q(h, \ldots, h^{\sigma^{s-1}}) = 0$. It remains to analyze the space of these solutions, and to show that they are nicely structured. This requires one more step, which goes through without change. More precisely, [GX13] takes a subcode of $C(m; k+2g-1)$; this subcode will still have a large list size, but the list will be structured. This resulting code, denoted $C(m; k + 2g - 1|\mathbb{F}_{q^m}^k)$, has dimension k. (Recall that $C(m; k + 2g - 1)$ has dimension $k + g$, so we have reduced the dimension by g.) We refer the reader to [GX13] for the details, as they do not matter for us. At the end of the day, the analysis of [GX13] (Lemma 4.8 in the full version [GX12b]) applies unchanged to show that the set of messages h in this new code that are solutions to this equation lie in a structured space: more precisely, the coefficients $(h_0, h_1, \ldots, h_{k+2g-1})$ as in (2) belong to an $(s-1, m)$-*ultra-periodic subspace* of $\mathbb{F}_q^{m(k+2g-1)}$. For us, the precise definition of this does not matter, as we may use the rest of [GX13] as a black box.

4. Before we move on, we summarize parameters. We have so far established that there is a code $C(m; k + 2g - 1|\mathbb{F}_{q^m}^k)$ that is list-recoverable with agreement parameter α and inner list sizes ℓ, resulting in a structured list. The requirement on α is:

$$\begin{aligned} \alpha n &> D + k + 2g - 1 \\ &= \left\lfloor \frac{\ell n - k + (s-1)g + 1}{s+1} \right\rfloor + k + 2g - 1, \end{aligned}$$

and so it suffices to take

$$\begin{aligned} \alpha n &> \frac{\ell n - k + (s-1)g + 1}{s+1} + k + 2g - 1 \\ &= \frac{1}{s+1}\left(\ell n + s(k-1) + g(3s+1)\right). \end{aligned}$$

Again, the dimension of the code is k and the length is n. It is $\mathbb{F}_{q^m}$-linear over $\mathbb{F}_{q^m}$.

Step 2. Taking a subcode. For this step, we may follow the argument of [GX13] without change. Briefly, to instantiate the AG code we use a function field from a *Garcia-Stichtenoth tower*. The parameters of this are as follows: we choose a prime power r, and let $q = r^2$. Then we choose an integer $e > 0$. There is a function field $F = K_e$ so that K_e has at least $n = r^{e-1}(r^2 - r) + 1$ rational places, and genus g_e bounded by r^e. This is the function field we will use. We remark

that [GX13] has to do a bit of work here to show that one can actually find a description of the structured list efficiently, but it can be done. We plug in parameters to obtain the following Lemma, which is analogous to Theorem 4.14 in [GX12b].

Lemma A.2. *Let q be the even power of a prime, and choose $\ell, \varepsilon > 0$. There is a parameter $s = O(\ell/\varepsilon)$ so that the following holds. Let $m \geq s$ and let $R \in (0,1)$. Suppose that $\alpha \geq R + \varepsilon + 3/\sqrt{q}$. Then for infinitely many n (all integers of the form $n = q^{e/2}(\sqrt{q}-1)$), there is a deterministic polynomial-time construction of an $\mathbb{F}_{q^m}$-linear code C of block length n, dimension $k = Rn$, so that the following holds: for any sets $S_1, \ldots, S_n \subseteq \mathbb{F}_{q^m}$ with $|S_i| \leq \ell$ for all i, the set of messages leading to codewords $c \in C$ so that $c_i \in S_i$ for at least αn coordinates i is contained in one of $q^{O(mn)}$ possible $(s-1, m)$-ultra periodic $\mathbb{F}_q$-affine subspaces of $\mathbb{F}_q^{mk}$. Further, this collection of subspaces can be described in time* $\mathrm{poly}(n, m)$.

Proof. Our condition on α is that it is at least

$$\begin{aligned} \frac{\ell n + s(k-1) + g_e(3s+1)}{n(s+1)} &\leq \frac{\ell n + s(k-1) + n(3s+1)/(r-1)}{n(s+1)} \qquad \text{Using } g_e \leq n/(r-1) \\ &= \frac{\ell + s(R - 1/n) + (3s+1)/(r-1)}{s+1}. \end{aligned}$$

Choosing $s = O(\ell/\varepsilon)$ and using the fact that $r = \sqrt{q}$ gives the conclusion. □

With this lemma in hand, we may proceed exactly as the proof in [GX13]; indeed, it is exactly the same code, and we exactly the same conclusion on the structure of the candidate messages. The basic idea is to choose a subset of messages carefully via a *cascaded subspace design.* This ensures that the number of legitimate messages remaining in the list is small, and further that they can be found efficiently.

We briefly go through parameters, again referring the reader to the discussion in [GX13, GK16b] for details. We will fix

$$s = O(\ell/\varepsilon), \qquad \text{and} \qquad m = O\left(\frac{\ell}{\varepsilon^2} \cdot \log_q(\ell/\varepsilon)\right). \tag{3}$$

We now trace these choices through the analysis of [GX13, GX14].

Remark A.3. The reader familiar with these sorts of arguments might expect us to set $m = \ell/\varepsilon^2$, and indeed this would be sufficient if we could allow q to be sufficiently large. However, in this case, setting m this way would result in a requirement that $q \geq \ell/\varepsilon^2$. We would like q to be independent of ℓ for the next concatenation step to work (of course, the alphabet size q^m must be larger than ℓ), and this requires us to take m slightly larger. This loss comes out in the final list size.

Without defining a cascaded subspace design, we will just mention that it is a sequence of T subspace designs (which we will not define either); a cascaded subspace design comes with vectors of parameters $(r_0, \ldots, r_T)$, $(m_0, \ldots, m_T)$, and $(d_0, \ldots, d_{T_1})$. For $i = 1, \ldots, T$, the i'th subspace design in this sequence is a (r_{i-1}, r_i)-strong-subspace design in $\mathbb{F}_q^{m_{i-1}}$, of cardinality m_i/m_{i-1}, and dimension d_{i-1}. Again, for us it does not matter what a strong subspace design is, only that we may find explicit ones:

Theorem A.4 (Follows from Theorem 6 in [GK16b]). *For all $\zeta \in (0,1)$ and for all r, m with $r \leq \zeta m/4$, and for all prime powers q so that $2r/\zeta < q^{\zeta m/(2r)}$, there exists an explicit collection of $M \geq q^{\Omega(\zeta m/r)}/(2r)$ subspaces in $\mathbb{F}_q^m$, each of codimension at most ζm, which form a $(r, r^2/\zeta)$-strong subspace design.*

Remark A.5. In [GK16b], the theorem is stated for $(r, r/\zeta)$-weak subspace designs; however, as is noted in that work, a (A, B)-weak subspace design is also a (A, AB)-strong subspace design, which yields our version of the theorem.

Below, we will use Theorem A.4 in order to instantiate a cascaded subspace design. The reason we want to do this is because of Lemma 5.6 in [GX12b]:

Lemma A.6 (Lemma 5.6 in [GX12b]). *Let $\mathcal{M}$ be a $(r_0, r_1, \ldots, r_T)$-cascaded subspace design with length-vector $(m_0, m_1, \ldots, m_T)$. Let A be an (r, m)-ultra periodic affine subspace of $\mathbb{F}_q^{m_T}$. Then the dimension of the affine space $A \cap U(\mathcal{M})$ is at most r_T, where $U(\mathcal{M})$ denotes the* canonical subspace *of $\mathcal{M}$.*

We have not defined a canonical subspace, and we refer the reader to [GX12b] for details; the important thing for us is that we wish to construct a cascaded subspace design $\mathcal{M}$ so that r_T is small, m_T is equal to mk, and so that $r_0 = s - 1$ and $m_0 = m$. This will allow us to choose a subcode of the code from Lemma A.2 by restricting the space of messages to the canonical subspace $U(\mathcal{M})$, and this will be the $\mathbb{F}_q$-linear code (over $\mathbb{F}_q^m$) that we are after.

We may use Theorem A.4 to instantiate such a cascaded subspace design as follows (the derivation below follows the proof of Lemma 5.7 in [GX12b]). We choose $\zeta_i = \varepsilon/2^i$, $r_0 = s-1$, and $r_i = r_{i-1}^2/\zeta_i$. We choose $m_0 = m$ and we will define $m_i = m_{i-1} \cdot q^{\sqrt{m_{i-1}}}$. We will continue up to $i = T$, choosing T so that $m_T = mk$. At this point, we must deal with the detail that there may be no such T; to deal with this we do exactly as in the proof of Lemma 5.7 in [GX12b] and modify our last two choices of m_{T-1}, m_T so that $m_T \leq mk$ but is close (within an additive $\log_q^2(km)$); for our final subspace, we will pad the m_T-dimensional vectors with 0's in order to form a subspace in $\mathbb{F}_q^{mk}$ with the same dimension. Choosing $m_T \approx mk$ puts $T = O(\log^*(mk))$, and $r_T = O(s/\varepsilon)^{2^T} \leq (\ell/\varepsilon)^{O(\log^*(mk))}$.

With these choices, we instantiate T subspace designs via Theorem A.4, with $m \leftarrow m_i, r \leftarrow r_i$, and $\zeta \leftarrow \zeta_i$. We check that the requirements of Theorem A.4 are satisfied, beginning with the requirement that $r_i \leq \zeta_i m_i/4$. Since $m_i \zeta_i$ grows much faster than r_i as i increases, it suffices to check this for $i = 0$, when we require $r_0 \leq \zeta_0 m_0$, or $s - 1 \leq m\varepsilon/8$. Our choices of m and s in (3) satisfy this.

The next requirement is that $2r_i/\zeta_i \leq q^{\zeta_i m_i/(2r_i)}$ for all i. Again, the right hand side grows much faster than the left, and so we establish this for $i = 0$, requiring that

$$\frac{4(s-1)}{\varepsilon} \leq q^{\varepsilon m/4(s-1)}.$$

With our choices of m and s, this requirement is that

$$\frac{\ell}{\varepsilon^2} \leq q^{O(\log_q(\ell/\varepsilon))},$$

which is true.

Thus, Theorem A.4 provides us with a cascaded subspace design with the given parameters. As mentioned above, we may then use Lemma A.6 to choose an appropriate subcode of our AG

code from Lemma A.2. We have chosen the parameters above so that $(r_0, m_0) = (s-1, m)$, precisely the guarantee of Lemma A.2. Thus, the final bound on the dimension of this intersection is $r_T \le (\ell/\varepsilon)^{O(\log^*(mk))}$, which gives our final list size. Finally, we observe (as in Observation 5.5 of [GX12b]) that the dimension of the resulting subcode is at least $(1 - \sum_i \zeta_i)m_T = (1-\varepsilon)mk$. Thus the final code has dimension at least $(1-\varepsilon)mk$ over $\mathbb{F}_q^{km}$, and hence the final rate is at least $(1-\varepsilon)R$. Observing that q must be at least ε^{-2} for the $1/\sqrt{q}$ term in Lemma A.2 to be absorbed into the additive ε factor, we arrive at the following theorem.

Theorem A.7. *Let q be an even power of a prime, and choose $\ell, \varepsilon > 0$, so that $q \ge \varepsilon^{-2}$. Choose $\rho \in (0,1)$. There is an $m_{min} = O(\ell \log_q(\ell/\varepsilon)/\varepsilon^2)$ so that the following holds for all $m \ge m_{min}$. For infinitely many n (all n of the form $q^{e/2}(\sqrt{q}-1)$ for any integer e), there is a deterministic polynomial-time construction of an $\mathbb{F}_q$-linear code $C : \mathbb{F}_{q^m}^{\rho n} \to \mathbb{F}_{q^m}^n$ of rate ρ and relative distance $1 - \rho - O(\varepsilon)$ that is $(1-\rho-\varepsilon, \ell, L)$-list-recoverable in time $\mathrm{poly}(n, L)$, returning a list that is contained in a subspace over $\mathbb{F}_q$ of dimension at most*

$$\left(\frac{\ell}{\varepsilon}\right)^{2^{\log^*(mk)}}.$$

We note that the distance of the code comes from the fact that it is a subcode of $C(m; k+3g_e-1)$, which has distance at least $n - (k+2g-1) = n - 2g - k + 1$. In the above parameter regime, the genus g_e satisfies $g_e \le n/(r-1) = n/(\sqrt{q}-1) = O(\varepsilon n)$. Thus, the relative distance of the final code is at least $(n - 2g_e - k + 1)/n \ge 1 - O(\varepsilon) - \rho$.

Step 3. Concatenating to obtain $\mathbb{F}_q$-linear codes over $\mathbb{F}_q$. Theorem A.7 gives codes over $\mathbb{F}_{q^m}$ which are $\mathbb{F}_q$-linear. For our purposes, to prove Theorem A.1, we require codes over $\mathbb{F}_q$ which are $\mathbb{F}_q$-linear. Thus, we will concatenate these codes with random $\mathbb{F}_q$-linear codes from Corollary 4.4 and apply Lemma 7.4 about the concatenation of list-recoverable codes. In more detail, we choose parameters as follows.

Let $\varepsilon > 0$ and let $\varepsilon' = \varepsilon/2$, and choose any integer ℓ and any block length N. Fix a constant c and parameters m and e which will be determined below. Choose an even prime power q so that

$$q \ge \max\left\{\ell^{c/\varepsilon}, \varepsilon^{-c}\right\}.$$

Let C_{in} be a random q-ary linear code of rate $\rho_{in} = 1 - \varepsilon'$ of length m/ρ_{in}. By Corollary 4.4, there exists an $\mathbb{F}_q$ linear code C_{in} with rate $\rho_{in} = 1-\varepsilon'$ and block length m/ρ_{in} which is $(\alpha_{in}, \ell_{in}, L_{in})$-list-recoverable, for $\alpha_{in} = \varepsilon'/2$, $\ell_{in} = \ell$, and $L_{in} = q^{2c\ell/\varepsilon'}$. We note that we can choose c large enough to ensure that the hypothesis of Corollary 4.4 hold.

Let C_{out} be the codes from Theorem A.7, instantiated with rate $\rho_{in} = 1-\varepsilon'$, $\varepsilon \leftarrow \varepsilon'/2$ and $\ell \leftarrow L_{in}$. With these parameters, we will get a code over $\mathbb{F}_{q^m}$ of length $n = q^{e/2}(\sqrt{q}-1)$ which is $(\alpha_{out}, L_{in}, L_{out})$-list-recoverable, where

$$\begin{aligned} L_{out} &= \exp_q\left((L_{in}/\varepsilon')^{2^{\log^*(mk)}}\right) \\ &= \exp_q\left(\left(\frac{q^{2c\ell/\varepsilon'}}{\varepsilon'}\right)^{2^{\log^*(mk)}}\right) \end{aligned}$$

and where

$$\alpha_{out} = 1 - \rho_{in} - \varepsilon' = \varepsilon'/2.$$

Let $m_{\min}$ be as in Theorem A.7, so that

$$m_{\min} = O(L_{in} \log_q(L_{in}/\varepsilon')/(\varepsilon')^2) = O\left(\frac{q^{c\ell/\varepsilon'} c\ell}{(\varepsilon')^3}\right).$$

We will choose m so that

$$m_{\min} \leq m \leq q \cdot m_{\min}. \tag{4}$$

Notice that, given the definition of $m_{\min} = O(q^{c\ell/\varepsilon'} c\ell/(\varepsilon')^3)$, choosing m slightly larger than $m_{\min}$—as large as $q \cdot m_{\min}$—amounts to replacing the constant c with $c+1$. Thus, the choices of m and c (subject to (4)) will not affect the list-recoverability of C_{out}, but they will affect the block length of the concatenated code.

Formally, Lemma 7.4 implies that the concatenated code has rate $\rho_{in} \cdot \rho_{out} = (1-\varepsilon')^2 \geq 1 - \varepsilon$, and is $(\alpha_{in}\alpha_{out}, \ell, L_{out})$-list-recoverable. Here, we have

$$\alpha_{in}\alpha_{out} = (\varepsilon')^2/4 = \Omega(\varepsilon^2),$$

which is what is claimed in Theorem A.1. The output list size claimed in Theorem A.1 follows from the choice of m and our guarantee on L_{out}. We note that the concatenated code will have message length $K = mk$, and so we write $\log^*(mk) = \log^*(K)$.

Finally, we choose m and e. At this point, the choice of these parameters (subject to (4)) will not affect that list-recoverability of the concatenated code, but they do control the block length of the code and the running time of the decoding algorithm. The block length is

$$\frac{m}{\rho_{in}} \cdot q^{e/2}(\sqrt{q} - 1).$$

In order to prove that we can come up with such codes for all sufficiently large block lengths N, as required in the statement of Theorem A.1, we must show that for all sufficiently large N, we can choose m satisfying (4) and e so that

$$N = \frac{m}{\rho_{in}} \cdot q^{e/2}(\sqrt{q} - 1).$$

That is, we want to find an integer e so that

$$\frac{N \cdot (1 - \varepsilon/2)}{q^{e/2}(\sqrt{q} - 1)} \in [m_{\min}, q \cdot m_{\min}].$$

However, we have chosen this window for m to be large enough so that such an e exists as long as N is sufficiently large (in terms of q, ℓ, ε). More precisely, for some large enough constant C, we require

$$N \geq q^{C\ell/\varepsilon},$$

which is our choice of N_0 in Theorem A.1.

Now we verify the running time of the list-recovery algorithm. The outer code C_{out} can be list-recovered in time $\text{poly}(n, L_{out})$ by Theorem A.7. The inner code can be list-recovered by brute force in time $q^{O(m)} = \exp_q\left(O\left(q^{2(c+1)\ell/\varepsilon}\ell/\varepsilon^3\right)\right) = \text{poly}(L_{out})$. Lemma 7.4 implies that the final running time is $\text{poly}(N, L)$, where $L = L_{out}$ is the final list size and N is the block length of the concatenated code.

ECCC ISSN 1433-8092

https://eccc.weizmann.ac.il