# Chapter 4

# RESULTS VIA CODE CONCATENATION

## *4.1 Introduction*

In Chapter 3, we presented efficient list-decoding algorithms for folded Reed-Solomon codes that can correct $1 - R - \varepsilon$ fraction of errors with rate $R$ (for any $\varepsilon > 0$). One drawback of folded Reed-Solomon codes is that they are defined over alphabets whose size is polynomial in the blocklength of the code. This is an undesirable feature of the code and we address this issue in this chapter.

First, we show how to convert folded Reed-Solomon codes to a related code that can still be list decoded up to $1-R-\varepsilon$ fraction of errors with rate $R$ (for any $\varepsilon > 0$). However, unlike folded Reed-Solomon codes these codes are defined over alphabets of size $2^{O(\varepsilon^{-4}\log(1/\varepsilon))}$. Recall that codes that can be list decoded up to $1 - R - \varepsilon$ fraction of errors need alphabets of size $2^{\Omega(\varepsilon^{-1})}$ (see section 2.2.1).

Next, we will show how to use folded Reed-Solomon codes to obtain codes over *fixed* alphabets (for example, binary codes). We will present explicit linear codes over fixed alphabets that achieve tradeoffs between rate and fraction of errors that satisfy the so called Zyablov and Blokh-Zyablov bounds (along with efficient list-decoding algorithms that achieve these tradeoffs). The codes list decodable up to the Blokh-Zyablov bound tradeoff are the best known to date for explicit codes over fixed alphabets. However, unlike Chapter 3, these results do not get close to the list-decoding capacity (see Figure 4.1). In particular, for binary codes, if $1/2 - \gamma$ fraction of errors are targeted, our codes have rate $\Omega(\gamma^3)$. By contrast, codes on list-decoding capacity will have rate $\Omega(\gamma^2)$. Unfortunately (as has been mentioned before), the only codes that are known to achieve list-decoding capacity are random codes for which no efficient list-decoding algorithms are known. Previous to our work, the best known explicit codes had rate $\Theta(\gamma^4)$ [51] (these codes also had efficient list-decoding algorithms). We choose to present the codes that are list decodable up to the Zyablov bound (even though the code that are list decodable up to the Blokh Zyablov have better rate vs. list decodability tradeoff) because of the following reasons (i) The construction is much simpler and these codes give the same asymptotic rate for the high error regime and (ii) The worst case list sizes and the code construction time are asymptotically smaller.

All our codes are based on code concatenation (and their generalizations called multi-level code concatenation). We next turn to an informal description of code concatenation.

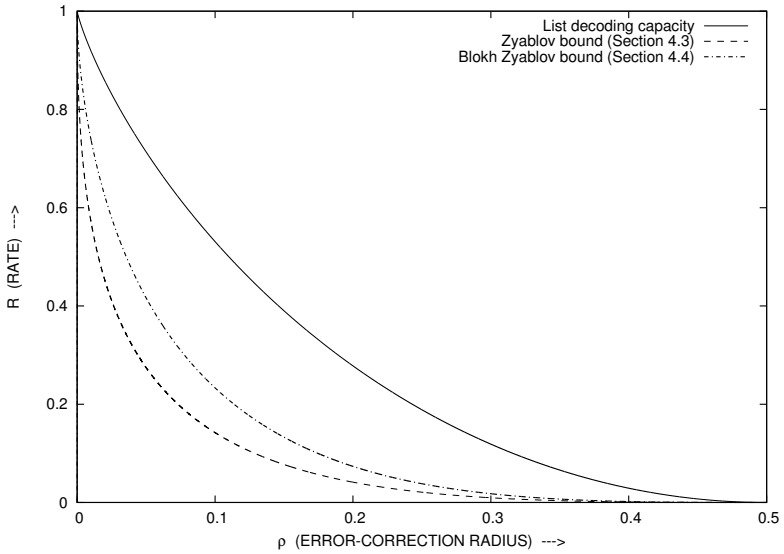


Figure 4.1: Rate $R$ of our binary codes plotted against the list-decoding radius $\rho$ of our algorithms. The best possible trade-off, i.e., list-decoding capacity, $\rho = H^{-1}(1-R)$ is also plotted.

### *4.1.1 Code Concatenation and List Recovery*

Concatenated codes were defined in the seminal thesis of Forney [40]. Concatenated codes are constructed from two different codes that are defined over alphabets of different sizes. Say we are interested in a code over $[q]$ (in this chapter, we will always think of $q \geqslant 2$ as being a fixed constant). Then the *outer code* $C_{out}$ is defined over $[Q]$, where $Q = q^k$ for some positive integer $k$. The second code, called the *inner code* is defined over $[q]$ and is of dimension $k$ (Note that the message space of $C_{in}$ and the alphabet of $C_{out}$ have the same size). The concatenated code, denoted by $C = C_{out} \circ C_{in}$, is defined as follows. Let the rate of $C_{out}$ be $R$ and let the blocklengths of $C_{out}$ and $C_{in}$ be $N$ and $n$ respectively. Define $K = RN$ and $r = k/n$. The input to $C$ is a vector $\mathbf{m} = \langle m_1, \ldots, m_K \rangle \in ([q]^k)^K$. Let $C_{out}(\mathbf{m}) = \langle x_1, \ldots, x_N \rangle$. The codeword in $C$ corresponding to $\mathbf{m}$ is defined as follows

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

It is easy to check that $C$ has rate $rR$, dimension $kK$ and blocklength $nN$.

Notice that to construct a $q$-ary code $C$ we use another $q$-ary code $C_{in}$. However, the nice thing about $C_{in}$ is that it has small blocklength. In particular, since $R$ and $r$ are constants (and typically $Q$ and $N$ are polynomially related), $n = O(\log N)$. This implies that we can use up exponential time (in $n$) to search for a "good" inner code. Further, one can use the brute force algorithm to (list) decode $C_{in}$.

Table 4.1: Values of rate at different decoding radius for List decoding capacity ($R_{Cap}$), Zyablov bound ($R_Z$) and Blokh Zyablov bound ($R_{BZ}$) in the binary case. For rates above 0.4, the Blokh Zyablov bound is 0 up to 3 decimal places, hence we have not shown this.

| $\rho$ | 0.01 | 0.02 | 0.03 | 0.05 | 0.10 | 0.15 | 0.20 | 0.25 | 0.30 | 0.35 |
|---|---|---|---|---|---|---|---|---|---|---|
| $R_{Cap}$ | 0.919 | 0.858 | 0.805 | 0.713 | 0.531 | 0.390 | 0.278 | 0.188 | 0.118 | 0.065 |
| $R_Z$ | 0.572 | 0.452 | 0.375 | 0.273 | 0.141 | 0.076 | 0.041 | 0.020 | 0.009 | 0.002 |
| $R_{BZ}$ | 0.739 | 0.624 | 0.539 | 0.415 | 0.233 | 0.132 | 0.073 | 0.037 | 0.017 | 0.006 |

Finally, we motivate why we are interested in list recovery. Consider the following natural decoding algorithm for the concatenated code $C_{out} \circ C_{in}$. Given a received word in $([q]^n)^N$, we divide it into $N$ blocks from $[q]^n$. Then we use a decoding algorithm for $C_{in}$ to get an intermediate received word to feed into a decoding algorithm for $C_{out}$. Now one can use unique decoding for $C_{in}$ and list decoding for $C_{out}$. However, this loses information in the first step. Instead, one can use the brute force list-decoding algorithm for $C_{in}$ to get a sequence of lists (each of which is a subset of $[Q]$). Now we use a list-recovery algorithm for $C_{out}$ to get the final list of codewords.

The natural algorithm above is used to design codes over fixed alphabets that are list decodable up to the Zyablov bound in Section 4.3 and (along with expanders) to design codes that achieve list-decoding capacity (but have much smaller alphabet size as compared to those for folded Reed-Solomon codes) in Section 4.2.

### *4.2 Capacity-Achieving Codes over Smaller Alphabets*

Theorem 3.5 has two undesirable aspects: both the alphabet size and worst-case list size output by the list-decoding algorithm are a polynomial of large degree in the block length. We now show that the alphabet size can be reduced to a constant that depends only on the distance $\varepsilon$ to capacity.

**Theorem 4.1.** *For every $R$, $0 < R < 1$, every $\varepsilon > 0$, there is a polynomial time constructible family of codes over an alphabet of size $2^{O(\varepsilon^{-4}\log(1/\varepsilon))}$ that have rate at least $R$ and which can be list decoded up to a fraction $(1 - R - \varepsilon)$ of errors in polynomial time.*

*Proof.* The theorem is proved using the code construction scheme used by Guruswami and Indyk in [54] for linear time unique decodable codes with optimal rate, with different components appropriate for list decoding plugged in. We briefly describe the main ideas behind the construction and proof below. The high level approach is to concatenate two codes $C_{\text{out}}$ and $C_{\text{in}}$, and then redistribute the symbols of the resulting codeword using an expander graph. Assume that $\varepsilon < (1 - R)/7$ and let $\delta = \varepsilon^2$.

The outer code $C_{\text{out}}$ will be a code of rate $(1 - 2\varepsilon)$ over an alphabet $\Sigma$ of size $n^{(1/\delta)^{O(1)}}$ that can be $(\varepsilon, O(1/\varepsilon))$-list recovered in polynomial time (to recall definitions pertaining to

list recovery, see Definition 2.4), as guaranteed by Theorem 3.6. That is, the rate of $C_{\text{out}}$ will be close to 1, and it can be $(\zeta, l)$-list recovered for large $l$ and $\zeta \to 0$.

The inner code $C_{\text{in}}$ will be a $((1 - R - 4\varepsilon), O(1/\varepsilon))$-list decodable code with near-optimal rate, say rate at least $(R + 3\varepsilon)$. Such a code is guaranteed to exist over an alphabet of size $O(1/\varepsilon^2)$ using random coding arguments. A naive brute-force for such a code, however, is too expensive, since we need a code with $|\Sigma| = n^{\Omega(1)}$ codewords. Guruswami and Indyk [52], see also [49, Sec. 9.3], prove that there is a small (quasi-polynomial sized) sample space of *pseudolinear codes* in which most codes have the needed property. Furthermore, they also present a deterministic polynomial time construction of such a code (using derandomization techniques), see [49, Sec. 9.3.3].

The concatenation of $C_{\text{out}}$ and $C_{\text{in}}$ gives a code $C_{\text{concat}}$ of rate at least $(1 - 2\varepsilon)(R + 3\varepsilon) \geqslant R$ over an alphabet $\Sigma$ of size $|\Sigma| = O(1/\varepsilon^2)$. Moreover, given a received word of the concatenated code, one can find all codewords that agree with the received word on a fraction $R + 4\varepsilon$ of locations in at least $(1 - \varepsilon)$ fraction of the inner blocks. Indeed, we can do this by running the natural list-decoding algorithm, call it $\mathcal{A}$, for $C_{\text{concat}}$ that decodes each of the inner blocks to a radius of $(1 - R - 4\varepsilon)$ returning up to $l = O(1/\varepsilon)$ possibilities for each block, and then $(\varepsilon, l)$-list recovering $C_{\text{out}}$ in polynomial time.

The last component in this construction is a $D = O(1/\varepsilon^4)$-regular bipartite expander graph which is used to redistribute symbols of the concatenated code in a manner so that an overall agreement on a fraction $R + 7\varepsilon$ of the redistributed symbols implies a fractional agreement of at least $R + 4\varepsilon$ on most (specifically a fraction $(1 - \varepsilon)$) of the inner blocks of the concatenated code. In other words, the expander redistributes symbols in a manner that "smoothens" the distributions of errors evenly among the various inner blocks (except for possibly a $\varepsilon$ fraction of the blocks). This expander based redistribution incurs no loss in rate, but increases the alphabet size to $O(1/\varepsilon^2)^{O(1/\varepsilon^4)} = 2^{O(\varepsilon^{-4} \log(1/\varepsilon))}$.

We now discuss some details of how the expander is used. Suppose that the block length of the folded Reed-Solomon code $C_{\text{out}}$ is $N_1$ and that of $C_{\text{in}}$ is $N_2$. Let us assume that $N_2$ is a multiple of $D$, say $N_2 = n_2 D$ (if this is not the case, we can make it so by padding at most $D - 1$ dummy symbols at a negligible loss in rate). Therefore codewords of $C_{\text{in}}$, and therefore also of $C_{\text{concat}}$, can be thought of as being composed of blocks of $D$ symbols each. Let $N = N_1 n_2$, so that codewords of $C_{\text{concat}}$ can be viewed as elements in $(\Sigma^D)^N$.

Let $G = (L, R, E)$ be a $D$-regular bipartite graph with $N$ vertices on each side (i.e., $|L| = |R| = N$), with the property that for every subset $Y \subseteq R$ of size at least $(R + 7\varepsilon)N$, the number of vertices belonging to $L$ that have at most $(R + 6\varepsilon)D$ of their neighbors in $Y$ is at most $\delta N$ (for $\delta = \varepsilon^2$). It is a well-known fact (used also in [54]) that if $G$ is picked to be the double cover of a Ramanujan expander of degree $D \geqslant 4/(\delta\varepsilon^2)$, then $G$ will have such a property.

We now define our final code $C^* = G(C_{\text{concat}}) \subseteq (\Sigma^D)^N$ formally. The codewords in $C^*$ are in one-one correspondence with those of $C_{\text{concat}}$. Given a codeword $c \in C_{\text{concat}}$, its $ND$ symbols (each belonging to $\Sigma$) are placed on the $ND$ edges of $G$, with the $D$ symbols in its $i$'th block (belonging to $\Sigma^D$, as defined above) being placed on the $D$ edges incident

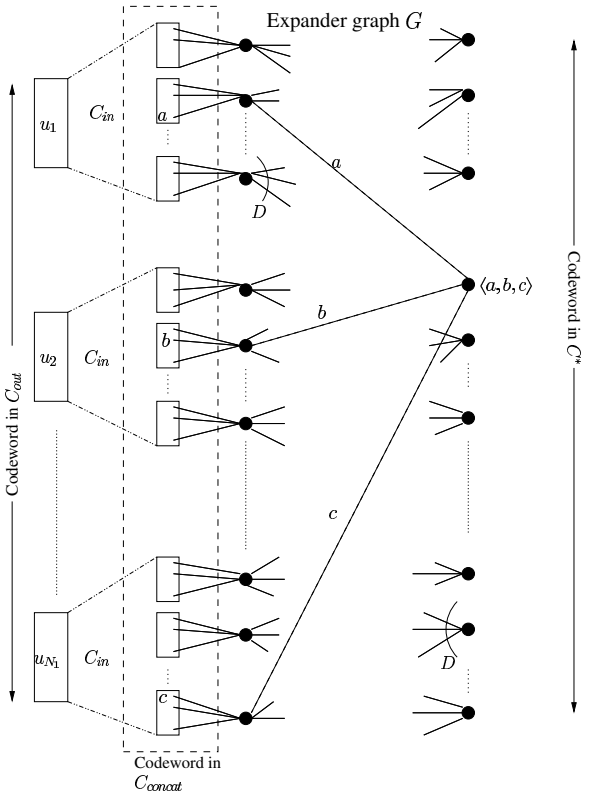


Figure 4.2: The code $C^*$ used in the proof of Theorem 4.1. We start with a codeword $\langle u_1, \ldots, u_{N_1} \rangle$ in $C_{out}$. Then every symbol is encoded by $C_{in}$ to form a codeword in $C_{concat}$ (this intermediate codeword is marked by the dotted box). The symbols in the codeword for $C_{concat}$ are divided into chunks of $D$ symbols and then redistributed along the edges of an expander $G$ of degree $D$. In the figure, we use $D = 3$ for clarity. Also the distribution of three symbols $a$, $b$ and $c$ (that form a symbol in the final codeword in $C^*$) is shown.

on the $i$'th vertex of $L$ (in some fixed order). The codeword in $C^*$ corresponding to $c$ has as its $i$'th symbol the collection of $D$ symbols (in some fixed order) on the $D$ edges incident on the $i$'th vertex of $R$. See Figure 4.2 for a pictorial view of the construction.

Note that the rate of $C^*$ is identical to that $C_{\text{concat}}$, and is thus at least $R$. Its alphabet size is $|\Sigma|^D = O(1/\varepsilon^2)^{O(1/\varepsilon^4)} = 2^{O(\varepsilon^{-4}\log(1/\varepsilon))}$, as claimed. We will now argue how $C^*$ can be list decoded up to a fraction $(1 - R - 7\varepsilon)$ of errors.

Given a received word $\mathbf{r} \in (\Sigma^D)^N$, the following is the natural algorithm to find all codewords of $C^*$ with agreement at least $(R+7\varepsilon)N$ with $\mathbf{r}$. Redistribute symbols according to the expander backwards to compute the received word $\mathbf{r'}$ for $C_{\text{concat}}$ which would result in $\mathbf{r}$. Then run the earlier-mentioned decoding algorithm $\mathcal{A}$ on $\mathbf{r'}$.

We now briefly argue the correctness of this algorithm. Let $\mathbf{c} \in C^*$ be a codeword with

agreement at least $(R + 7\varepsilon)N$ with $\mathbf{r}$. Let $\mathbf{c}'$ denote the codeword of $C_{\text{concat}}$ that leads to $\mathbf{c}$ after symbol redistribution by $G$, and finally suppose $\mathbf{c}''$ is the codeword of $C_{\text{out}}$ that yields $\mathbf{c}'$ upon concatenation by $C_{\text{in}}$. By the expansion properties of $G$, it follows that all but a $\delta$ fraction of $N$ $D$-long blocks of $\mathbf{r}'$ have agreement at least $(R+6\varepsilon)D$ with the corresponding blocks of $\mathbf{c}'$. By an averaging argument, this implies that at least a fraction $(1 - \sqrt{\delta})$ of the $N_1$ blocks of $\mathbf{c}'$ that correspond to codewords of $C_{\text{in}}$ encoding the $N_1$ symbols of $\mathbf{c}''$, agree with at least a fraction $(1 - \sqrt{\delta})(R + 6\varepsilon) = (1 - \varepsilon)(R + 6\varepsilon) \geqslant R + 4\varepsilon$ of the symbols of the corresponding block of $\mathbf{r}'$. As argued earlier, this in turn implies that the decoding algorithm $\mathcal{A}$ for $C_{\text{concat}}$ when run on input $\mathbf{r}'$ will output a polynomial size list that will include $\mathbf{c}'$. □

### *4.3 Binary Codes List Decodable up to the Zyablov Bound*

Concatenating the folded Reed-Solomon codes with suitable inner codes also gives us polytime constructible binary codes that can be efficiently list decoded up to the Zyablov bound, i.e., up to twice the radius achieved by the standard GMD decoding of concatenated codes [41]. The optimal list recoverability of the folded Reed-Solomon codes plays a crucial role in establishing such a result.

**Theorem 4.2.** *For all $0 < R, r < 1$ and all $\varepsilon > 0$, there is a polynomial time constructible family of binary linear codes of rate at least $R \cdot r$ which can be list decoded in polynomial time up to a fraction $(1 - R)H^{-1}(1 - r) - \varepsilon$ of errors.*

*Proof.* Let $\gamma > 0$ be a small constant that will be fixed later. We will construct binary codes with the claimed property by concatenating two codes $C_1$ and $C_2$. For $C_1$, we will use a folded Reed-Solomon code over a field of characteristic 2 with block length $n_1$, rate at least $R$, and which can be $(1-R-\gamma, l)$-list recovered in polynomial time for $l = \lceil 10/\gamma \rceil$. Let the alphabet size of $C_1$ be $2^M$ where $M$ is $O(\gamma^{-2} \log(1/\gamma)(1 - R)^{-1} \log n_1)$ (by Theorem 3.6, such a $C_1$ exists). For $C_2$, we will use a binary linear code of dimension $M$ and rate at least $r$ which is $(\rho, l)$-list decodable for $\rho = H^{-1}(1 - r - \gamma)$. Such a code is known to exist via a random coding argument that employs the semi-random method [51]. Also, a greedy construction of such a code by constructing its $M$ basis elements in turn is presented in [51] and this process takes $2^{O(M)}$ time. We conclude that the necessary inner code can be constructed in $n_1^{O(\gamma^{-2}(1-R)^{-1} \log(1/\gamma))}$ time. The code $C_1$, being a folded Reed-Solomon code over a field of characteristic 2, is $\mathbb{F}_2$-linear, and therefore when concatenated with a binary linear inner code such as $C_2$, results in a binary linear code. The rate of the concatenated code is at least $R \cdot r$.

The decoding algorithm proceeds in a natural way. Given a received word, we break it up into blocks corresponding to the various inner encodings by $C_1$. Each of these blocks is list decoded up to a radius $\rho$, returning a set of at most $l$ possible candidates for each outer codeword symbol. The outer code is then $(1 - R - \gamma, l)$-list recovered using these sets, each of which has size at most $l$, as input. To argue about the fraction of errors this

algorithm corrects, we note that the algorithm fails to recover a codeword only if on more than a fraction $(1-R-\gamma)$ of the inner blocks the codeword differs from the received word on more than a fraction $\rho$ of symbols. It follows that the algorithm correctly list decodes up to a radius $(1-R-\gamma)\rho = (1-R-\gamma)H^{-1}(1-r-\gamma)$. If we pick an appropriate $\gamma$ in $\Theta(\varepsilon^2)$, then by Lemma 2.4, $H^{-1}(1-r-\gamma) \geqslant H^{-1}(1-r)-\varepsilon/3$ (and $(1-R-\gamma) \geqslant 1-R-\varepsilon/3$), which implies $(1-R-\gamma)H^{-1}(1-r-\gamma) \geqslant (1-R)H^{-1}(1-r)-\varepsilon$ as desired. □

Optimizing over the choice of inner and outer codes rates $r, R$ in the above results, we can decode up to the Zyablov bound, see Figure 4.1. For an analytic expression, see (4.2) with $s = 1$.

**Remark 4.1.** *In particular, decoding up to the Zyablov bound implies that we can correct a fraction $(1/2-\varepsilon)$ of errors with rate $\Omega(\varepsilon^3)$ for small $\varepsilon \to 0$, which is better than the rate of $\Omega(\varepsilon^3/\log(1/\varepsilon))$ achieved in [55]. However, our construction and decoding complexity are $n^{O(\varepsilon^{-2}\log(1/\varepsilon))}$ whereas these are at most $f(\varepsilon)n^c$ for an absolute constant $c$ in [55]. Also, we bound the list size needed in the worst-case by $n^{O(\varepsilon^{-1}\log(1/\varepsilon))}$, while the list size needed in the construction in [55] is $(1/\varepsilon)^{O(\log\log(1/\varepsilon))}$.*

### *4.4 Unique Decoding of a Random Ensemble of Binary Codes*

We will digress a bit to talk about a consequence of (the proof of) Theorem 4.2.

One of the biggest open questions in coding theory is to come up with explicit binary codes that are on the Gilbert Varshamov (or GV) bound. In particular, these are codes that achieve relative distance $\delta$ with rate $1 - H(\delta)$. There exist ensembles of binary codes for which if one picks a code at random then with high probability it lies on the GV bound. Coming up with an explicit construction of such a code, however, has turned out to be an elusive task.

Given the bleak state of affairs, some attention has been paid to the following problem. Give a probabilistic construction of binary codes that meet the GV bound (with high probability) together with efficient (encoding and) *decoding up to half the distance* of the code. Zyablov and Pinsker [110] give such a construction for binary codes of rate about 0.02 with *subexponential* time decoding algorithms. Guruswami and Indyk [53] give such a construction for binary linear codes up to rates about $10^{-4}$ with *polynomial* time encoding and decoding algorithms. Next we briefly argue that Theorem 4.2 can be used to extend the result of [53] to work till rates of about 0.02. In other words, we get the rate achieved by the construction of [110] but (like [53]) we get *polynomial* time encoding and decoding (up to half the GV bound).

We start with a brief overview of the construction of [53], which is based on code concatenation. The outer code is chosen to be the Reed-Solomon code (of say length $N$ and rate $R$) while there are $N$ linear binary inner codes of rate $r$ (recall that in the "usual" code concatenation only one inner code is used) that are chosen uniformly (and independently) at random. A result of Thommesen [102] states that with high probability such a code

lies on the GV bound provided the rates of the codes satisfy $R \leqslant \alpha(r)/r$, where $\alpha(r) = 1 - H(1 - 2^{r-1})$. Guruswami and Indyk then give list-decoding algorithms for such codes such that for (overall) rate $rR \leqslant 10^{-4}$, the fraction of errors they can correct is at least $\frac{1}{2} \cdot H^{-1}(1 - rR)$ (that is, more than half the distance on the GV bound) as well as satisfy the constraint in Thommesen's result.

Given Theorem 4.2, here is the natural way to extend the result of [53]. We pick the outer code of rate $R$ to be a folded Reed-Solomon code (with the list recoverable properties as required in Theorem 4.2) and the pick $N$ independent binary linear codes of rate $r$ as the inner codes. It is not hard to check that the proof of Thommesen also works when the outer code is folded Reed-Solomon. That is, the construction just mentioned lies on the GV bound with high probability. It is also easy to check that the proof of Theorem 4.2 also works when all the inner codes are different (basically the list decoding for the inner code in Theorem 4.2 is done by brute-force, which of course one can do even if all the $N$ inner codes are different). Thus, if $rR \leqslant \alpha(r)$, we can list decode up to $(1 - R)H^{-1}(1 - r)$ fraction of errors and at the same time have the property that with high probability, the constructed code lies on the GV bound. Thus, all we now need to do is to check what is the maximum rate $rR$ one can achieve while at the same time satisfying $rR \leqslant \alpha(r)$ and $(1-R)H^{-1}(1-r) \geqslant \frac{1}{2}H^{-1}(1-rR)$. This rate turns out to be around $0.02$ (see Figure 4.3).

Thus, we have argued the following.

**Theorem 4.3.** *There is a probabilistic polynomial time procedure to construct codes whose rate vs. distance tradeoff meets the Gilbert-Varshamov bound with high probability for all rates up to* $0.02$*. Furthermore, these codes can be decoded in polynomial time up to half the relative distance.*

One might hope that this method along with ideas of multilevel concatenated codes (about which we will talk next) can be used to push the overall rate significantly up from $0.02$ that we achieve here. However, the following simple argument shows that one cannot go beyond a rate of $0.04$. If we are targeting list decoding up to $\rho$ fraction of errors (and use code concatenation), then the inner rate $r$ must be at most $1 - H(\rho)$ (see for example (4.2)). Now by the Thommesen condition the overall rate is at most $\alpha(r)$. It is easy to check that $\alpha(\cdot)$ is an increasing function. Thus, the maximum overall rate that we can achieve is $\alpha(1 - H(\rho))$— this is the curve titled "Limit of the method" in Figure 4.3. One can see from Figure 4.3, the maximum rate for which this curve still beats half the GV bound is at most $0.04$.

### *4.5 List Decoding up to the Blokh Zyablov Bound*

We now present linear codes over any fixed alphabet that can be constructed in polynomial time and can be efficiently list decoded up to the so called Blokh-Zyablov bound (Figure 4.1). This achieves a sizable improvement over the tradeoff achieved by codes from Section 4.3 (see Figure 4.1 and Table 4.1).

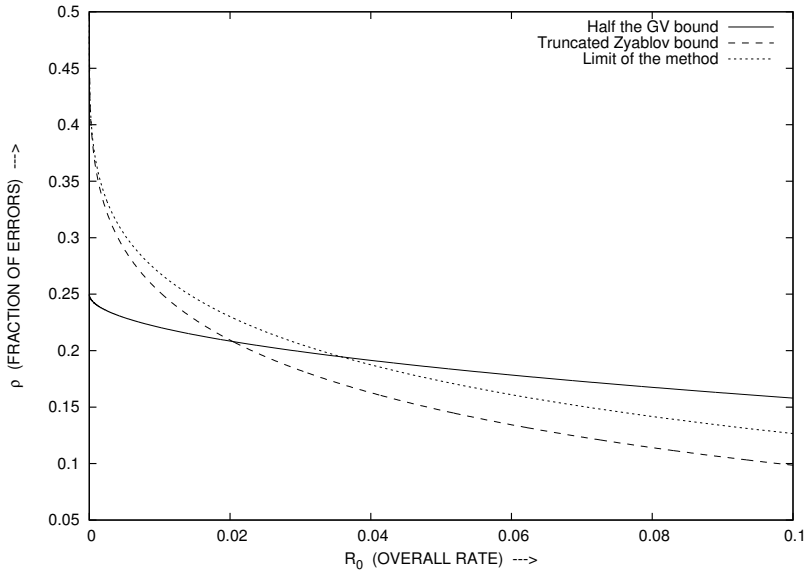


Figure 4.3: Tradeoffs for decoding certain random ensemble of concatenated codes. "Half the GV bound" is the curve $\frac{1}{2} \cdot H^{-1}(1 - R_0)$ while "Truncated Zyablov bound" is the limit till which we can list decode the concatenated codes (and still satisfy the Thommesen condition, that is for inner and outer rates $r$ and $R$, $rR = R_0 \leqslant \alpha(r)$). "Limit of the method" is the best tradeoff one can hope for using list decoding of code concatenation along with the Thommesen result.

Our codes are constructed via multilevel concatenated codes. We will provide a formal definition later on — we just sketch the basic idea here. For an integer $s \geqslant 1$, a multilevel concatenated code $C$ over $\mathbb{F}_q$ is obtained by combining $s$ "outer" codes $C^0_{out}, C^1_{out}, \ldots, C^{s-1}_{out}$ of the same block length , say $N$, over large alphabets of size say $q^{a_0}, q^{a_1}, \ldots, q^{a_{s-1}}$, respectively, with a suitable "inner" code. The inner code is of dimension $a_0 + a_1 \cdots + a_{s-1}$. Given messages $m^0, m^1, \ldots, m^{s-1}$ for the $s$ outer codes, the encoding as per the multilevel generalized concatenation codes proceeds by first encoding each $m^j$ as per $C^j_{out}$. Then for every $1 \leqslant i \leqslant N$, the collection of the $i$th symbols of $C^j_{out}(m^j)$ for $0 \leqslant j \leqslant s-1$, which can be viewed as a string over $\mathbb{F}_q$ of length $a_0 + a_1 + \cdots + a_{s-1}$, is encoded by the inner code. For $s = 1$ this reduces to the usual definition of code concatenation.

We present a list-decoding algorithm for $C$, given list-recovery algorithms for the outer codes and list-decoding algorithms for the inner code and some of its subcodes. What makes this part more interesting than the usual code concatenation (like those in Section 4.3), is that the inner code in addition to having good list-decodable properties, also needs to have good list-decodable properties for certain subcodes. Specifically, the subcodes of dimension $a_j + a_{j+1} + \cdots + a_{s-1}$ of the inner code obtained by arbitrarily fixing the first

$a_0 + \cdots + a_{j-1}$ symbols of the message, must have better list-decodability properties for increasing $j$ (which is intuitively possible since they have lower rate). In turn, this allows the outer codes $C^j_{out}$ to have rates increasing with $j$, leading to an overall improvement in the rate for a certain list-decoding radius.

To make effective use of the above approach, we also prove, via an application of the probabilistic method, that a random linear code over $\mathbb{F}_q$ has the required stronger condition on list decodability. By applying the method of conditional expectation ([2]), we can construct such a code deterministically in time singly exponential in the block length of the code (which is polynomial if the inner code encodes messages of length $O(\log N)$). Note that constructing such an inner code, given the existence of such codes, is easy in quasi-polynomial time by trying all possible generator matrices. The lower time complexity is essential for constructing the final code $C$ in polynomial time.

### *4.5.1 Multilevel Concatenated Codes*

We will be working with multilevel concatenation coding schemes [33]. We start this section with the definition of multilevel concatenated codes. As the name suggests, these are generalizations of concatenated codes. Recall that for a concatenated code, we start with a code $C_{out}$ over a large alphabet (called the outer code). Then we need a code $C_{in}$ that maps all symbols of the larger alphabet to strings over a smaller alphabet (called the inner code). The encoding for the concatenated code (denoted by $C_{out} \circ C_{in}$) is done as follows. We think of the message as being a string over the large alphabet and then encode it using $C_{out}$. Now we use $C_{in}$ to encode each of the symbols in the codeword of $C_{out}$ to get our codeword (in $C_{out} \circ C_{in}$) over the smaller alphabet.

Multilevel code concatenation generalizes the usual code concatenation in the following manner. Instead of there being one outer code, there are multiple outer codes. In particular, we "stack" codewords from these multiple outer codes and construct a matrix. The inner codes then act on the columns of these intermediate matrix. We now formally define multilevel concatenated codes.

There are $s \geqslant 1$ outer codes, denoted by $C^0_{out}, C^1_{out}, \ldots, C^{s-1}_{out}$. For every $0 \leqslant i \leqslant s-1$, $C^i_{out}$ is a code of block length $N$ and rate $R_i$ and defined over a field $\mathbb{F}_{Q_i}$. The inner code $C_{in}$ is code of block length $n$ and rate $r$ that maps tuples from $\mathbb{F}_{Q_0} \times \mathbb{F}_{Q_1} \times \cdots \times \mathbb{F}_{Q_{s-1}}$ to symbols in $\mathbb{F}_q$. In other words,

$$C^i_{out} : (\mathbb{F}_{Q_i})^{R_i N} \to (\mathbb{F}_{Q_i})^N,$$

$$C_{in} : \mathbb{F}_{Q_0} \times \mathbb{F}_{Q_1} \times \cdots \times \mathbb{F}_{Q_{s-1}} \to (\mathbb{F}_q)^n.$$

The multilevel concatenated code, denoted by $(C^0_{out} \times C^1_{out} \times \ldots C^{s-1}_{out}) \circ C_{in}$ is a map of the following form:

$$(C^0_{out} \times C^1_{out} \times \ldots C^{s-1}_{out}) \circ C_{in} : (\mathbb{F}_{Q_0})^{R_0 N} \times (\mathbb{F}_{Q_1})^{R_1 N} \times \cdots \times (\mathbb{F}_{Q_{S-1}})^{R_{s-1} N} \to (\mathbb{F}_q)^{nN}.$$

We now describe the encoding scheme. Given a message $(m^0, m^1, \ldots, m^{s-1}) \in (\mathbb{F}_{Q_0})^{R_0 N} \times (\mathbb{F}_{Q_1})^{R_1 N} \times \cdots \times (\mathbb{F}_{Q_{S-1}})^{R_{s-1} N}$, we first construct an $s \times N$ matrix $M$, whose $i^{\text{th}}$ row is the codeword $C^i_{out}(m^i)$. Note that every column of $M$ is an element from the set $\mathbb{F}_{Q_0} \times \mathbb{F}_{Q_1} \times \cdots \times \mathbb{F}_{Q_{s-1}}$. Let the $j^{\text{th}}$ column (for $1 \leqslant j \leqslant N$) be denoted by $M_j$. The codeword corresponding to the multilevel concatenated code ($C \stackrel{def}{=} (C^0_{out} \times C^1_{out} \times \ldots C^{s-1}_{out}) \circ C_{in}$) is defined as follows

$$C(m^0, m^1, \ldots, m^{s-1}) = (C_{in}(M_1), C_{in}(M_2), \cdots, C_{in}(M_N)) .$$

(The codeword can be naturally be thought of as an $n \times N$ matrix, whose $i$'th column corresponds to the inner codeword encoding the $i$'th symbols of the $s$ outer codewords.)

For the rest of the chapter, we will only consider outer codes over the same alphabet, that is, $Q_0 = Q_1 = \cdots = Q_{s-1} = Q$. Further, $Q = q^a$ for some integer $a \geqslant 1$. Note that if $C^0_{out}, \ldots, C^{s-1}_{out}$ and $C_{in}$ are all $\mathbb{F}_q$ linear, then so is $(C^0_{out} \times C^1_{out} \times \cdots \times C^{s-1}_{out}) \circ C_{in}$.

The gain from using multilevel concatenated codes comes from looking at the inner code $C_{in}$ along with its subcodes. For the rest of the section, we will consider the case when $C_{in}$ is linear (though the ideas can easily be generalized for general codes). Let $G \in \mathbb{F}_q^{as \times n}$ be the generator matrix for $C_{in}$. Let $r_0 = as/n$ denote the rate of $C_{in}$. For $0 \leqslant j \leqslant s-1$, define $r_j = r_0(1 - j/s)$, and let $G_j$ denote $r_j n \times n$ submatrix of $G$ containing the last $r_j n$ rows of $G$. Denote the code generated by $G_j$ by $C^j_{in}$; the rate of $C^j_{in}$ is $r_j$. For our purposes we will actually look at the subcode of $C_{in}$ where one fixes the first $0 \leqslant j \leqslant s-1$ message symbols. Note that for every $j$ these are just cosets of $C^j_{in}$. We will be looking at $C_{in}$, which in addition to having good list decoding properties as a "whole," also has good list-decoding properties for each of its subcode $C^j_{in}$.

The multilevel concatenated code $C$ ($= (C^0_{out} \times \cdots \times C^{s-1}_{out}) \circ C_{in}$) has rate $R(C)$ that satisfies

$$R(C) = \frac{r_0}{s} \sum_{i=0}^{s-1} R_i \, . \tag{4.1}$$

The Blokh-Zyablov bound is the trade-off between rate and relative distance obtained when the outer codes meet the Singleton bound (i.e., $C^j_{out}$ has relative distance $1 - R_j$), and the various subcodes $C^j_{in}$ of the inner code, including the whole inner code $C_{in} = C^0_{in}$, lie on the Gilbert-Varshamov bound (i.e., have relative distance $\delta_j \geqslant H_q^{-1}(1 - r_j)$). The multilevel concatenated code then has relative distance at least $\min_{0 \leqslant j \leqslant s-1}(1 - R_j) H_q^{-1}(1 - r_j)$. Expressing the rate in terms of distance, the Blokh-Zyablov bound says that there exist multilevel concatenated code $C$ with relative distance at least $\delta$ with the following rate:

$$R^s_{BZ}(C) = \max_{0 < r < 1 - H_q(\delta)} r - \frac{r}{s} \sum_{i=0}^{s-1} \frac{\delta}{H_q^{-1}(1 - r + ri/s)} . \tag{4.2}$$

As $s$ increases, the trade-off approaches the integral

$$R_{BZ}(C) = 1 - H_q(\delta) - \delta \int_0^{1 - H_q(\delta)} \frac{dx}{H_q^{-1}(1 - x)} \, . \tag{4.3}$$

The convergence of $R^s_{BZ}(C)$ to $R_{BZ}(C)$ happens quite quickly even for small $s$ such as $s = 10$.

*Nested List Decoding*

We will need to work with the following generalization of list decoding. The definition looks more complicated than it really is.

**Definition 4.1 (Nested linear list decodable code).** *Given a linear code $C$ in terms of some generator matrix $G \in \mathbb{F}_q^{k \times n}$, an integer $s$ that divides $k$, a vector $\mathbf{L} = \langle L_0, L_1, \ldots, L_{s-1} \rangle$ of integers $L_j$ ($0 \leqslant j \leqslant s-1$), a vector $\rho = \langle \rho_0, \rho_1 \ldots, \rho_{s-1} \rangle$ with $0 < \rho_j < 1$, and a vector $\mathbf{r} = \langle r_0, \ldots, r_{s-1} \rangle$ of reals where $r_0 = k/n$ and $0 \leqslant r_{s-1} < \cdots < r_i < r_0$, $C$ is called an $(\mathbf{r}, \rho, \mathbf{L})$-nested list decodable if the following holds:*

*For every $0 \leqslant j \leqslant s-1$, $C^j$ is a rate $r_j$ code that is $(\rho_j, L_j)$-list decodable, where $C^j$ is the subcode of $C$ generated by the the last $r_j n$ rows of the generator matrix $G$.*

### *4.5.2 Linear Codes with Good Nested List Decodability*

In this section, we will prove the following result concerning the existence (and constructibility) of linear codes over any fixed alphabet with good nested list-decodable properties.

**Theorem 4.4.** *For any integer $s \geqslant 1$ and reals $0 < r_{s-1} < r_{s-2} < \cdots < r_1 < r_0 < 1$, $\varepsilon > 0$, let $\rho_j = H_q^{-1}(1 - r_j - 2\varepsilon)$ for every $0 \leqslant j \leqslant s-1$. Let $\mathbf{r} = \langle r_0, \ldots, r_{s-1} \rangle$, $\rho = \langle \rho_0, \rho_1, \ldots, \rho_{s-1} \rangle$ and $\mathbf{L} = \langle L_0, L_1, \ldots, L_{s-1} \rangle$, where $L_j = q^{1/\varepsilon}$. For large enough $n$, there exists a linear code (over fixed alphabet $\mathbb{F}_q$) that is $(\mathbf{r}, \rho, \mathbf{L})$-nested list decodable. Further, such a code can be constructed in time $q^{O(n/\varepsilon)}$.*

*Proof.* We will show the existence of the required codes via a simple use of the probabilistic method (in fact, we will show that a random linear code has the required properties with high probability). We will then use the method of conditional expectation ([2]) to derandomize the construction with the claimed time complexity.

Define $k_j = \lfloor r_i n \rfloor$ for every $0 \leqslant j \leqslant s-1$. We will pick a random $k_0 \times n$ matrix $G$ with entries picked independently from $\mathbb{F}_q$. We will show that the linear code $C$ generated by $G$ has good nested list decodable properties with high probability. Let $C_j$, for $0 \leqslant j \leqslant s-1$ be the code generated by the "bottom" $k_j$ rows of $G$. Recall that we have to show that with high probability $C_j$ is $(\rho_j, q^{1/\varepsilon})$ list decodable for every $0 \leqslant j \leqslant s-1$ ($C_j$ obviously has rate $r_j$). Finally for integers $J, k \geqslant 1$, and a prime power $q$, let $\mathrm{Ind}(q, k, J)$ denote the collection of subsets $\{x^1, x^2, \ldots, x^J\} \subseteq \mathbb{F}_q^k$ such that all vectors $x^1, \ldots, x^J$ are linearly independent over $\mathbb{F}_q$.

We recollect the following two straightforward facts: (i) Given any $L$ distinct vectors from $\mathbb{F}_q^k$, for some $k \geqslant 1$, at least $\lceil \log_q L \rceil$ of them are linearly independent; (ii) Any set of linearly independent vectors in $\mathbb{F}_q^k$ are mapped to independent random vectors in $\mathbb{F}_q^n$ by

a random $k \times n$ matrix over $\mathbb{F}_q$. The first claim is obvious. For the second claim, first note that for any $\mathbf{v} \in \mathbb{F}_q^k$ and a random $k \times n$ matrix $\mathbf{G}$ (where each of the $kn$ values are chosen uniformly and independently at random from $\mathbb{F}_q$) the values at the $n$ different positions in $\mathbf{v} \cdot \mathbf{G}$ are independent. Further, the value at position $1 \leqslant i \leqslant n$, is given by $\mathbf{v} \cdot \mathbf{G}_i$, where $\mathbf{G}_i$ is the $i^{\text{th}}$ column of $\mathbf{G}$. Now for fixed $\mathbf{v}$, $\mathbf{v} \cdot \mathbf{G}_i$ takes values from $\mathbb{F}_q$ uniformly at random (note that $\mathbf{G}_i$ is a random vector from $\mathbb{F}_q^k$). Finally, for linearly independent vectors $\mathbf{v}^1, \ldots, \mathbf{v}^m$ by a suitable linear invertible map can be mapped to the standard basis vectors $\mathbf{e}_1, \ldots, \mathbf{e}_m$. Obviously, the values $\mathbf{e}_1 \cdot \mathbf{G}_i \ \ldots, \mathbf{e}_m \cdot \mathbf{G}_i$ are independent.

We now move on to the proof of existence of linear codes with good nested list decodability. We will actually do the proof in a manner that will facilitate the derandomization of the proof. Define $J = \lceil \log_q(q^{1/\varepsilon} + 1) \rceil$. For any vector $\mathbf{y} \in \mathbb{F}_q^n$, integer $0 \leqslant j \leqslant s-1$, subset $T = \{x^1, \ldots, x^J\} \in \mathrm{Ind}(q, k_j, J)$ and any collection $\mathcal{S}$ of subsets $S_1, S_2, \ldots, S_J \subseteq \{1, \ldots, n\}$ of size at most $\rho_j n$, define an indicator variable $I(j, \mathbf{y}, T, \mathcal{S})$ in the following manner. $I(j, \mathbf{y}, T, \mathcal{S}) = 1$ if and only if for every $1 \leqslant i \leqslant J$, $C(x^i)$ differs from $\mathbf{y}$ in exactly the set $S_i$. Note that if for some $0 \leqslant j \leqslant s-1$, there are $q^{1/\varepsilon} + 1$ codewords in $C_j$ all of which differ from some received word $\mathbf{y}$ in at most $\rho_j n$ places, then this set of codewords is a "counter-example" that shows that $C$ is not $(\mathbf{y}, \rho, \mathbf{L})$-nested list decodable. Since the $q^{1/\varepsilon} + 1$ codewords will have some set $T$ of $J$ linearly independent codewords, the counter example will imply that $I(j, \mathbf{y}, T, \mathcal{S}) = 1$ for some collection of subsets $\mathcal{S}$. In other words, the indicator variable captures the set of bad events we would like to avoid. Finally define the sum of all the indicator variables as follows:

$$S_C = \sum_{j=0}^{s-1} \sum_{\mathbf{y} \in \mathbb{F}_q^n} \sum_{T \in \mathrm{Ind}(q, k_j, J)} \sum_{\substack{\mathcal{S} = \{S_1, \ldots, S_J\}, \\ S_i \subseteq \{1, \ldots, n\}, |S_i| \leqslant \rho_j n}} I(j, \mathbf{y}, T, \mathcal{S}).$$

Note that if $S_C = 0$, then $C$ is $(\mathbf{y}, \rho, \mathbf{L})$-nested list decodable as required. Thus, we can prove the existence of such a $C$ if we can show that $\mathbb{E}_{\mathbf{C}}[S_C] < 1$. By linearity of expectation, we have

$$\mathbb{E}[S_C] = \sum_{j=0}^{s-1} \sum_{\mathbf{y} \in \mathbb{F}_q^n} \sum_{T \in \mathrm{Ind}(q, k_j, J)} \sum_{\substack{\mathcal{S} = \{S_1, \ldots, S_J\}, \\ S_i \subseteq \{1, \ldots, n\}, |S_i| \leqslant \rho_j n}} \mathbb{E}[I(j, \mathbf{y}, T, \mathcal{S})]. \tag{4.4}$$

Fix some arbitrary $j, \mathbf{y}, T = \{x^1, x^2, \ldots, x^J\}, \mathcal{S} = \{S_1, S_2, \ldots, S_J\}$ (in their corresponding domains). Then we have

$$\begin{aligned}
\mathbb{E}[I(j, \mathbf{y}, T, \mathcal{S})] &= \Pr[I(j, \mathbf{y}, T, \mathcal{S}) = 1] \\
&= \prod_{x^i \in T} \Pr[C(x^i) \text{ differ from } \mathbf{y} \text{ in exactly the positions in } S_i] \\
&= \prod_{i=1}^{J} \left(\frac{q-1}{q}\right)^{|S_i|} \left(\frac{1}{q}\right)^{n - |S_i|}
\end{aligned} \tag{4.5}$$

$$= \prod_{i=1}^{J} \frac{(q-1)^{|S_i|}}{q^n} \,, \tag{4.6}$$

where the second and the third equality follow from the definition of the indicator variable, the fact that vectors in $T$ are linearly independent and the fact that a random matrix maps linearly independent vectors to independent uniformly random vectors in $\mathbb{F}_q^n$. Using (4.6) in (4.4), we get

$$\begin{aligned}
\mathbb{E}[S_C] &= \sum_{j=0}^{s-1} \sum_{\mathbf{y}\in\mathbb{F}_q^n} \sum_{T\in \mathrm{Ind}(q,k_j,J)} \sum_{\substack{\mathcal{S}=\{S_1,\ldots,S_J\},\\ S_i\subseteq\{1,\ldots,n\},|S_i|\leqslant\rho_j n}} \prod_{i=1}^{J} \frac{(q-1)^{|S_i|}}{q^n} \\
&= \sum_{j=0}^{s-1} \sum_{\mathbf{y}\in\mathbb{F}_q^n} \sum_{T\in \mathrm{Ind}(q,k_j,J)} \sum_{(\ell_1,\ell_2,\ldots,\ell_J)\in\{0,1,\ldots,\rho_j n\}^J} \prod_{i=1}^{J} \binom{n}{\ell_i} \frac{(q-1)^{\ell_i}}{q^n} \\
&= \sum_{j=0}^{s-1} \sum_{\mathbf{y}\in\mathbb{F}_q^n} \sum_{T\in \mathrm{Ind}(q,k_j,J)} \left( \sum_{\ell=0}^{\rho_j n} \binom{n}{\ell} \frac{(q-1)^{\ell}}{q^n} \right)^J \\
&\leqslant \sum_{j=0}^{s-1} \sum_{\mathbf{y}\in\mathbb{F}_q^n} \sum_{T\in \mathrm{Ind}(q,k_j,J)} q^{nJ(H_q(\rho_j)-1)} \\
&\leqslant \sum_{j=0}^{s-1} q^n \cdot q^{Jk_j} \cdot q^{nJ(H_q(\rho_j)-1)} \\
&\leqslant \sum_{j=0}^{s-1} q^{nJ(1/J+r_j+1-r_j-2\varepsilon-1)} \\
&\leqslant sq^{-\varepsilon nJ} \,.
\end{aligned} \tag{4.7}$$

The first inequality follows from Proposition 2.1. The second inequality follows by upper bounding the number of $J$ linearly independent vectors in $\mathbb{F}_q^{k_j}$ by $q^{Jk_j}$. The third inequality follows from the fact that $k_j = \lfloor r_j n \rfloor$ and $\rho_j = H_q^{-1}(1-r_j-2\varepsilon)$, The final inequality follows from the fact that $J = \lceil \log_q(q^{1/\varepsilon}+1) \rceil$.

Thus, (4.7) shows that there exists a code $C$ (in fact with high probability) that is $(\mathbf{y}, \rho, \mathbf{L})$-nested list decodable. In fact, this could have been proved using a simpler argument. However, the advantage of the argument above is that we can now apply the method of conditional expectations to derandomize the above proof.

The algorithm to deterministically generate a linear code $C$ that is $(\mathbf{y}, \rho, \mathbf{L})$-nested list decodable is as follows. The algorithm consists of $n$ steps. At any step $1 \leqslant i \leqslant n$, we choose the $i^{\text{th}}$ column of the generator matrix to be the value $\mathbf{v}^i \in \mathbb{F}_q^{k_0}$ that minimizes the conditional expectation $\mathbb{E}[S_C | \mathbf{G}_1 = \mathbf{v}^1, \ldots, \mathbf{G}_{i-1} = \mathbf{v}^{i-1}, \mathbf{G}_i = \mathbf{v}^i]$, where $\mathbf{G}_i$ denotes the $i^{\text{th}}$ column of $\mathbf{G}$ and $\mathbf{v}^1, \ldots, \mathbf{v}^{i-1}$ are the column vectors chosen in the previous $i-1$

steps. This algorithm would work if for any $1 \leqslant i \leqslant n$ and vectors $\mathbf{v}^1, \ldots, \mathbf{v}^i$, we can exactly compute $\mathbb{E}[S_C | \mathbf{G}_1 = \mathbf{v}^1, \ldots, \mathbf{G}_i = \mathbf{v}^i]$. Indeed from (4.4), we have $\mathbb{E}[S_C | \mathbf{G}_1 = \mathbf{v}^1, \ldots, \mathbf{G}_i = \mathbf{v}^i]$ is

$$\sum_{j=0}^{s-1} \sum_{\mathbf{y} \in \mathbb{F}_q^n} \sum_{T \in \mathrm{Ind}(q,k_j,J)} \sum_{\substack{\mathcal{S}=\{S_1,\ldots,S_J\}, \\ S_i \subseteq \{1,\ldots,n\}, |S_i| \leqslant \rho_j n}} \mathbb{E}[I(j,\mathbf{y},T,\mathcal{S}) | \mathbf{G}_1 = \mathbf{v}^1, \ldots, \mathbf{G}_i = \mathbf{v}^i].$$

Thus, we would be done if we can compute the following for every value of $j, \mathbf{y}, T = \{x^1, \ldots, x^J\}, \mathcal{S} = \{S_1, \ldots, S_J\}$: $\mathbb{E}[I(j,\mathbf{y},T,\mathcal{S}) = 1 | \mathbf{G}_1 = \mathbf{v}^1, \ldots, \mathbf{G}_i = \mathbf{v}^i]$. Note that fixing the first $i$ columns of $G$ implies fixing the value of the codewords in the first $i$ positions. Thus, the indicator variable is 0 (or in other words, the conditional expectation we need to compute is 0) if for some message, the corresponding codeword does not disagree with $\mathbf{y}$ exactly as dictated by $\mathcal{S}$ in the first $i$ positions. More formally, $I(j,\mathbf{y},T,\mathcal{S}) = 0$ if the following is true for some $1 \leqslant \ell \leqslant i$ and $0 \leqslant i' \leqslant J$: $x^{i'} \cdot \mathbf{G}_\ell \neq \mathbf{y}_\ell$, if $\ell \notin S_{i'}$ and $x^{i'} \cdot \mathbf{G}_\ell = \mathbf{y}_\ell$ otherwise. However, if none of these conditions hold, then using argument similar to the ones used to obtain (4.6), one can show that

$$\mathbb{E}[I(j,\mathbf{y},T,\mathcal{S}) | \mathbf{G}_1 = \mathbf{v}^1, \ldots, \mathbf{G}_i = \mathbf{v}^i] = \prod_{\ell=1}^{J} \left(\frac{q-1}{q}\right)^{|S'_\ell|} \left(\frac{1}{q}\right)^{n-i-|S'_\ell|},$$

where $S'_\ell = S_\ell \setminus \{1, 2, \ldots, i\}$ for every $1 \leqslant \ell \leqslant J$.

To complete the proof, we need to estimate the time complexity of the above algorithm. There are $n$ steps and at every step $i$, the algorithm has to consider $q^{k_0} \leqslant q^n$ different choices of $\mathbf{v}^i$. For every choice of $\mathbf{v}^i$, the algorithm has to compute the conditional expectation of the indicator variables for all possible values of $j, \mathbf{y}, T, \mathcal{S}$. It is easy to check that there are $\sum_{i=1}^{s} q^n \cdot q^{Jk_j} \cdot 2^{nJ} \leqslant sq^{n(1+2J)}$ possibilities. Finally, the computation of the conditional expected value of a fixed indicator variable takes time $O(snJ)$. Thus, in all the total time taken is $O(n \cdot q^n \cdot sq^{n(1+2J)} \cdot snJ) = q^{O(n/\varepsilon)}$, as required. □

### *4.5.3 List Decoding Multilevel Concatenated Codes*

In this section, we will see how one can list decode multilevel concatenated codes, provided the outer codes have good list recoverability and the inner code has good nested list decodability. We have the following result, which generalizes Theorem 4.2 for regular concatenated codes (the case $s = 1$).

**Theorem 4.5.** *Let $s \geqslant 1$ and $\ell \geqslant 1$ be integers. Let $0 < R_0 < R_1 < \cdots < R_{s-1} < 1$, $0 < r_0 < 1$ be rationals and $0 < \xi_0, \cdots, \xi_{s-1} < 1$, $0 < \rho_0, \cdots, \rho_{s-1} < 1$ and $\varepsilon > 0$ be reals. Let $q$ be a prime power and let $Q = q^a$ for some integer $a > 1$. Further, let $C_{out}^j$ ($0 \leqslant j \leqslant s-1$) be an $\mathbb{F}_q$-linear code over $\mathbb{F}_Q$ of rate $R_j$ and block length $N$ that is $(\xi_j, \ell, L)$-list recoverable. Finally, let $C_{in}$ be a linear $(\mathbf{r}, \rho, \mathbf{L})$-nested list decodable code over $\mathbb{F}_q$ of*

*rate $r_0$ and block length $n = as/r_0$, where $\mathbf{r} = \langle r_0, \cdots, r_{s-1} \rangle$ with $r_i = (1 - i/s)r_0$, $\rho = \langle \rho_0, \cdots, \rho_{s-1} \rangle$ and $\mathbf{L} = \langle \ell, \ell, \cdots, \ell \rangle$. Then $C = (C_{out}^0 \times \cdots \times C_{out}^{s-1}) \circ C_{in}$ is a linear $(\min_j \xi_j \cdot \rho_j, L^s)$-list decodable code. Further, if the outer code $C_{out}^j$ can be list recovered in time $T_j(N)$ and the inner code $C_{in}$ can be list decoded in time $t_j(n)$ (for the $j^{\text{th}}$ level), then $C$ can be list decoded in time $O\left(\sum_{j=0}^{s-1} L^j \left(T_j(N) + N \cdot t_j(n)\right)\right)$.*

*Proof.* Given list-recovery algorithms for $C_{out}^j$ and list-decoding algorithms for $C_{in}$ (and its subcodes $C_{in}^j$), we will design a list-decoding algorithm for $C$. Recall that the received word is an $n \times N$ matrix over $\mathbb{F}_q$. Each consecutive "chunk" of $n/s$ rows should be decoded to a codeword in $C_{out}^j$. The details follow.

Before we describe the algorithm, we will need to fix some notation. Define $\delta = \min_j \xi_j \rho_j$. Let $\mathbf{Y} \in \mathbb{F}_q^{nN}$ be the received word, which we will think of as an $n \times N$ matrix over $\mathbb{F}_q$ (note that $s$ divides $n$). For any $n \times N$ matrix $M$ and for any $1 \leqslant i \leqslant N$, let $M_i \in \mathbb{F}_q^n$ denote the $i^{\text{th}}$ column of the matrix $M$. Finally, for every $0 \leqslant j \leqslant s-1$, let $C_{in}^j$ denote the subcode of $C_{in}$ generated by all but the first $ja$ rows of the generator matrix of $C_{in}$. We are now ready to describe our algorithm.

Recall that the algorithm needs to output all codewords in $C$ that differ from $\mathbf{Y}$ in at most $\delta$ fraction of positions. For the ease of exposition, we will consider an algorithm that outputs matrices from $C_{out}^0 \times \cdots \times C_{out}^{s-1}$. The algorithm has $s$ phases. At the end of phase $j$ ($0 \leqslant j \leqslant s-1$), the algorithm will have a list of matrices (called $\mathcal{L}_j$) from $C_{out}^0 \times \cdots \times C_{out}^j$, where each matrix in $\mathcal{L}_j$ is a possible submatrix of some matrix that will be in the final list output by the algorithm. The following steps are performed in phase $j$ (where we are assuming that the list-decoding algorithm for $C_{in}^j$ returns a list of messages while the list-recovery algorithm for $C_{out}^j$ returns a list of codewords).

1. Set $\mathcal{L}_j$ to be the empty set.

2. For every $\mathbf{c} = (c^0, \cdots, c^{j-1}) \in \mathcal{L}_{j-1}$ repeat the following steps (if this is the first phase, that is $j = 0$, then repeat the following steps once):

   (a) Let $G_j$ be the first $aj$ rows of the generator matrix of $C_{in}$. Let $\mathbf{X} = (G_j)^T \cdot \mathbf{c}$, where we think of $\mathbf{c}$ as an $ja \times N$ matrix over $\mathbb{F}_q$. Let $\mathbf{Z} = \mathbf{Y} - \mathbf{X}$ (for $j = 0$ we use the convention that $\mathbf{X}$ is the all 0s matrix). For every $1 \leqslant i \leqslant N$, use the list-decoding algorithm for $C_{in}^j$ on column $\mathbf{Z}_i$ for up to $\rho_j$ fraction of errors to obtain list $S_i^j \subseteq (\mathbb{F}_Q)^{s-j}$. Let $T_i^j \subseteq \mathbb{F}_Q$ be the projection of every vector in $S_i^j$ on to its first component.

   (b) Run the list-recovery algorithm for $C_{out}^j$ on set of lists $\{T_i^j\}_i$ obtained from the previous step for up to $\xi_j$ fraction of errors. Store the set of codewords returned in $I_j$.

   (c) Add $\{(\mathbf{c}, \mathbf{v}) | \mathbf{v} \in I_j\}$ to $\mathcal{L}_j$.

At the end, remove all the matrices $M \in \mathcal{L}_{s-1}$, for which the codeword $(C_{in}(M_1), C_{in}(M_2), \cdots, C_{in}(M_N))$ is at a distance more than $\delta$ from $\mathbf{Y}$. Output the remaining matrices as the final answer.

We will first talk about the running time complexity of the algorithm. It is easy to check that each repetition of steps 2(a)-(c) takes time $O(T_j(N) + N \cdot t_j(n))$. To compute the final running time, we need to get a bound on number of times step 2 is repeated in phase $j$. It is easy to check that the number of repetitions is exactly $|\mathcal{L}_{j-1}|$. Thus, we need to bound $|\mathcal{L}_{j-1}|$. By the list recoverability property of $C^j_{out}$, we can bound $|I_j|$ by $L$. This implies that $|\mathcal{L}_j| \leqslant L|\mathcal{L}_{j-1}|$, and therefore by induction we have

$$|\mathcal{L}_i| \leqslant L^{i+1} \quad \text{for } i = 0, 1, \ldots, s-1 \,. \tag{4.8}$$

Thus, the overall running time and the size of the list output by the algorithm are as claimed in the statement of the theorem.

We now argue the correctness of the algorithm. That is, we have to show that for every $M \in C^0_{out} \times \cdots \times C^{s-1}_{out}$, such that $(C_{in}(M_1), C_{in}(M_2) \cdots, C_{in}(M_N))$ is at a distance at most $\delta$ from $\mathbf{Y}$ (call such an $M$ a *good* matrix), $M \in \mathcal{L}_{s-1}$. In fact, we will prove a stronger claim: for every good matrix $M$ and every $0 \leqslant j \leqslant s-1$, $M^j \in \mathcal{L}_j$, where $M^j$ denotes the submatrix of $M$ that lies in $C^0_{out} \times \cdots \times C^j_{out}$ (that is the first $j$ "rows" of $M$). For the rest of the argument fix an arbitrary good matrix $M$. Now assume that the stronger claim above holds for $j' - 1$ ($< s-1$). In other words, $M^{j'-1} \in \mathcal{L}_{j'-1}$. Now, we need to show that $M^{j'} \in \mathcal{L}_{j'}$.

For concreteness, let $M = (m^0, \cdots, m^{s-1})^T$. As $M$ is a good matrix and $\delta \leqslant \xi_{j'}\rho_{j'}$, $C_{in}(M_i)$ can disagree with $\mathbf{Y}_i$ on at least a fraction $\rho_{j'}$ of positions for at most $\xi_{j'}$ fraction of column indices $i$. The next crucial observation is that for any column index $i$, $C_{in}(M_i) = (G_{j'})^T \cdot (m^0_i, \cdots, m^{j'-1}_i) + (G \setminus G_{j'})^T \cdot (m^{j'}_i, \cdots, m^{s-1}_i)$, where $G_{j'}$ is as defined in step 2(a), $G \setminus G_{j'}$ is the submatrix of $G$ obtained by "removing" $G_{j'}$ and $m^{j'}_i$ is the $i^{\text{th}}$ component of the vector $m^{j'}$. Figure 4.5.3 might help the reader to visualize the different variables. Note that $G \setminus G_{j'}$ is the generator matrix of $C^{j'}_{in}$. Thus, for at most $\xi_{j'}$ fraction of column indices $i$, $(m^{j'}_i, \cdots, m^{s-1}_i) \cdot (G \setminus G_{j'})$ disagrees with $\mathbf{Y}_i - \mathbf{X}_i$ on at least $\rho_{j'}$ fraction of places, where $\mathbf{X}$ is as defined in Step 2(a), and $\mathbf{X}_i$ denotes the $i$'th column of $\mathbf{X}$. As $C^{j'}_{in}$ is $(\rho_{j'}, \ell)$-list decodable, for at least $1 - \xi_{j'}$ fraction of column index $i$, $M^{j'}_i$ will be in $S^{j'}_i$ (where $M^{j'}_i$ is $M_i$ projected on it's last $s - j'$ co-ordinates and $S^{j'}_i$ is as defined in Step 2(a)). In other words, $m^{j'}_i$ is in $T^{j'}_i$ for at least $1 - \xi_{j'}$ fraction of $i$'s. Further, as $|S^{j'}_i| \leqslant \ell$, $|T^{j'}_i| \leqslant \ell$. This implies with the list recoverability property of $C^{j'}_{out}$ that $m^{j'} \in I_{j'}$, where $I_{j'}$ is as defined in step 2(b). Finally, step 2(c) implies that $M^{j'} \in \mathcal{L}_{j'}$ as required.

The proof of correctness of the algorithm along with (4.8) shows that $C$ is $(\delta, L^s)$-list decodable, which completes the proof. $\square$

$$G^T \cdot M = \begin{pmatrix} (G_{j'})^T & (G \setminus G_{j'})^T \end{pmatrix} \cdot \begin{pmatrix} m_1^0 & \cdots & m_i^0 & \cdots & m_N^0 \\ & & \vdots & & \\ m_1^{j'-1} & \cdots & m_i^{j'-1} & \cdots & m_N^{j'-1} \\ m_1^{j'} & \cdots & m_i^{j'} & \cdots & m_N^{j'} \\ & & \vdots & & \\ m_1^{s-1} & \cdots & m_i^{s-1} & \cdots & m_N^{s-1} \end{pmatrix}$$
$$= \begin{pmatrix} \uparrow & & \uparrow & & \uparrow \\ C_{in}(M_1) & \cdots & C_{in}(M_i) & \cdots & C_{in}(M_N) \\ \downarrow & & \downarrow & & \downarrow \end{pmatrix}$$

Figure 4.4: Different variables in the proof of Theorem 4.5.

### *4.5.4 Putting it Together*

We combine the results we have proved in the last couple of subsections to get the main result of this section.

**Theorem 4.6.** *For every fixed field $\mathbb{F}_q$, reals $0 < \delta < 1, 0 < r \leqslant 1 - H_q(\delta), \varepsilon > 0$ and integer $s \geqslant 1$, there exists linear codes $C$ over $\mathbb{F}_q$ of block length $N$ that are $(\delta - \varepsilon, L(N))$-list decodable with rate $R$ such that*

$$R = r - \frac{r}{s} \sum_{i=0}^{s-1} \frac{\delta}{H_q^{-1}(1 - r + ri/s)}, \tag{4.9}$$

*and $L(N) = (N/\varepsilon^2)^{O\left(s\varepsilon^{-3}\delta/(H_q^{-1}(1-r)-\delta)\right)}$. Finally, $C$ can be constructed in time $(N/\varepsilon^2)^{O(s/(\varepsilon^6 r\delta))}$ and list decoded in time polynomial in $N$.*

*Proof.* Let $\gamma > 0$ (we will define its value later). For every $0 \leqslant j \leqslant s-1$ define $r_j = r(1 - j/s)$ and $R_j = 1 - \frac{\delta}{H_q^{-1}(1-r_j)}$. The code $C$ is going to be a multilevel concatenated code $(C_{out}^0 \times \cdots \times C_{out}^{s-1}) \circ C_{in}$, where $C_{out}^j$ is the code from Corollary 3.7 of rate $R_j$ and block length $N'$ (over $\mathbb{F}_{q^a}$) and $C_{in}$ is an $(\langle r_0, \ldots, r_{s-1} \rangle, \rho, \mathbf{L})$-nested list decodable code as guaranteed by Theorem 4.4, where for $0 \leqslant j \leqslant s-1$, $\rho_j = H_q^{-1}(1 - r_j - 2\gamma^2)$ and $L_j = q^{1/\gamma^2}$. Finally, we will use the property of $C_{out}^j$ that it is $(1 - R_j - \gamma, q^{1/\gamma^2}, (N'/\gamma^2)^{O(\gamma^{-3}\log(1/R_j))})$-list recoverable for any $0 \leqslant \gamma \leqslant R_j$. Corollary 3.7 implies that such codes exist with (where we apply Corollary 3.7 with $R' = \max_j R_j = 1 - \delta/H_q^{-1}(1 - r/s)$)

$$q^a = (N'/\gamma^2)^{O(\gamma^{-4} H_q^{-1}(1-r/s)/\delta)}. \tag{4.10}$$

Further, as codes from Corollary 3.7 are $\mathbb{F}_q$-linear, $C$ is a linear code.

The claims on the list decodability of $C$ follows from the choices of $R_j$ and $r_j$, Corollary 3.7 and Theorems 4.4 and 4.5. In particular, note that we invoke Theorem 4.5 with the following parameters: $\xi_j = 1 - R_j - \gamma$ and $\rho_j = H_q^{-1}(1 - r_j - 2\gamma^2)$ (which by Lemma 2.4 implies that that $\xi_j \rho_j \geqslant \delta - \varepsilon$ as long as $\gamma = \Theta(\varepsilon)$), $\ell = q^{1/\gamma^2}$ and $L = (N'/\gamma^2)^{O(\gamma^{-1}\log(\ell/R_j))}$. The choices of $\ell$ and $\gamma$ imply that $L = (N/\varepsilon^2)^{O(\varepsilon^{-3}\log(1/R_j))}$. Now $\log(1/R_j) \leqslant \log(1/R_{min})$, where $R_{min} = \min_j R_j = 1 - \delta/H_q^{-1}(1-r)$. Finally, we use the fact that for any $0 < y < 1$, $\ln(1/y) \leqslant 1/y - 1$ to get that $\log(1/R_j) \leqslant O(1/R_{min} - 1) = O(\delta/(H_q^{-1}(1-r) - \delta))$. The claimed upper bound of $L(N)$ follows as $L(N) \leqslant L^s$ (by Theorem 4.5).

By the choices of $R_j$ and $r_j$ and (4.1), the rate of $C$ is as claimed. The construction time for $C$ is the time required to construct $C_{in}$, which by Theorem 4.4 is $2^{O(n/\gamma^2)}$ where $n$ is the block length of $C_{in}$. Note that $n = as/r$, which by (4.10) implies that the construction time is $(N/\varepsilon^2)^{O(\varepsilon^{-6} s H_q^{-1}(1-r/s)/(r\delta))}$. The claimed running time follows by using the bound $H_q^{-1}(1 - r/s) \leqslant 1$.

We finally consider the running time of the list-decoding algorithm. We list decode the inner code(s) by brute force, which takes $2^{O(n)}$ time, that is, $t_j(n) = 2^{O(n)}$. Thus, Corollary 3.7, Theorem 4.5 and the bound on $L(N)$ implies the claimed running time complexity. □

Choosing the parameter $r$ in the above theorem so as to maximize (4.9) gives us linear codes over any fixed field whose rate vs. list-decoding radius tradeoff meets the Blokh-Zyablov bound (4.2). As $s$ grows, the trade-off approaches the integral form (4.3) of the Blokh-Zyablov bound.

### *4.6 Bibliographic Notes and Open Questions*

We managed to reduce the alphabet size needed to approach capacity to a constant independent of $n$. However, this involved a brute-force search for a rather large code. Obtaining a "direct" algebraic construction over a constant-sized alphabet (such as variants of algebraic-geometric codes, or AG codes) might help in addressing these two issues. To this end, Guruswami and Patthak [55] define *correlated AG codes*, and describe list-decoding algorithms for those codes, based on a generalization of the Parvaresh-Vardy approach to the general class of algebraic-geometric codes (of which Reed-Solomon codes are a special case). However, to relate folded AG codes to correlated AG codes like we did for Reed-Solomon codes requires bijections on the set of rational points of the underlying algebraic curve that have some special, hard to guarantee, property. This step seems like a highly intricate algebraic task, and especially so in the interesting asymptotic setting of a family of asymptotically good AG codes over a fixed alphabet.

Our proof of existence of the requisite inner codes with good nested list decodable properties (and in particular the derandomization of the construction of such codes using

conditional expectation) is similar to the one used to establish list decodability properties of random "pseudolinear" codes in [52] (see also [49, Sec. 9.3]).

Concatenated codes were defined in the seminal thesis of Forney [40]. Its generalizations to linear multilevel concatenated codes were introduced by Blokh and Zyablov [20] and general multilevel concatenated codes were introduced by Zinoviev [108]. Our list-decoding algorithm is inspired by the argument for "unequal error protection" property of multilevel concatenated codes [109].

The results on capacity achieving list decodable codes over small alphabets (Section 4.2) and binary linear codes that are list decodable up to the Zyablov bound (Section 4.3) appeared in [58]. The result on linear codes that are list decodable up to the Blokh Zyablov bound (Section 4.5) appeared in [60].

The biggest and perhaps most challenging question left unresolved by our work is the following.

**Open Question 4.1.** *For every $0 < \rho < 1/2$ and every $\varepsilon > 0$ give explicit construction of binary codes that are $(\rho, O(1/\varepsilon))$-list decodable with rate $1 - H(\rho) - \varepsilon$. Further, design polynomial time list decoding algorithms that can correct up to $\rho$ fraction of errors.*

In fact, just resolving the above question for *any* fixed $\rho$ (even with an exponential time list-decoding algorithm) is widely open at this point.