Random Reed–Solomon Codes Achieve the Half-Singleton Bound for Insertions and Deletions over Linear-Sized Alphabets

Roni Con[*] Zeyu Guo[†] Ray Li[‡] Zihan Zhang[§]

Abstract

In this paper, we prove that with high probability, random Reed–Solomon codes approach the half-Singleton bound — the optimal rate versus error tradeoff for linear insdel codes — with linear-sized alphabets. More precisely, we prove that, for any $\varepsilon > 0$ and positive integers n and k, with high probability, random Reed–Solomon codes of length n and dimension k can correct $(1-\varepsilon)n-2k+1$ adversarial insdel errors over alphabets of size $n+2^{\mathsf{poly}(1/\varepsilon)}k$. This significantly improves upon the alphabet size demonstrated in the work of Con, Shpilka, and Tamo (IEEE TIT, 2023), who showed the existence of Reed–Solomon codes with exponential alphabet size $\widetilde{O}\left(\binom{n}{2k-1}^2\right)$ precisely achieving the half-Singleton bound.

Our methods are inspired by recent works on list-decoding Reed–Solomon codes. Brakensiek–Gopi–Makam (STOC 2023) showed that random Reed–Solomon codes are list-decodable up to capacity with exponential-sized alphabets, and Guo–Zhang (FOCS 2023) and Alrabiah–Guruswami–Li (STOC 2024) improved the alphabet-size to linear. We achieve a similar alphabet-size reduction by similarly establishing strong bounds on the probability that certain random rectangular matrices are full rank. To accomplish this in our insdel context, our proof combines the random matrix techniques from list-decoding with structural properties of Longest Common Subsequences.

[*]Department of Computer Science, Technion–Israel Institute of Technology. roni.con93@gmail.com
[†]Department of Computer Science and Engineering, The Ohio State University. zguotcs@gmail.com
[‡]Mathematics and Computer Science Department, Santa Clara University. rli6@scu.edu
[§]Department of Computer Science and Engineering, The Ohio State University. zhang.13691@osu.edu

1 Introduction

Error-correcting codes (hereafter referred to as codes) are constructs designed to enable the recovery of original information from data that has been corrupted. The primary corruption model explored in the literature involves errors or erasures. In this model, each symbol in the transmitted word is either replaced with a different symbol from the alphabet (an error) or with a '?' (an erasure).

The theory of coding theory originated with the influential works of Shannon and Hamming. Shannon [Sha48] studied random errors and erasures, whereas Hamming [Ham50] studied the adversarial model for errors and erasures. These models are well understood, and today, we have efficiently encodable and decodable codes that are optimal for Shannon's model of random errors. For adversarial errors, optimal and efficient codes exist over large alphabets, and there are good codes (codes with constant relative rate and relative distance) for every constant-sized alphabet.

Another important model that has been considered ever since Shannon's work is that of *synchronization* errors. The most common model for studying synchronization errors is the insertion-deletion model (insdel for short): an insertion error is when a new symbol is inserted between two symbols of the transmitted word and a deletion is when a symbol is removed from the transmitted word. For example, over the binary alphabet, when 100110 is transmitted, we may receive the word 1101100, which is obtained from two insertions (1 at the beginning and 0 at the end) and one deletion (one of the 0's at the beginning of the transmitted word). These errors are related to the more common *substitution* and *erasure* errors: a deletion is like an erasure, but the position of the deletion is unknown, and a substitution can be imitated by a deletion followed by an insertion. Despite their apparent similarity to well studied error models, insdel errors are much more challenging to handle.

Coding schemes that correct insdel errors are not only an intriguing theoretical concept but also have implications in real-world scenarios. Such codes find applications in magnetic and optical recording, semiconductor devices, integrated circuits, and synchronous digital communication networks (for detailed applications, refer to the survey by Mercier [MBT10]). This natural theoretical model together coupled with its relevance across various domains, has led many researches in recent years to study and design codes that can correct from insdel errors [BGH16, HS21b, BGZ17, GW17, SWZGY17, CJLW22, Hae19, CGHL21, GH21, GHL22], just to name a few. Although there has been significant progress in recent years on understanding this model of insdel errors (both on limitation and constructing efficient codes), our comprehension of this model lags far behind our understanding of codes that correct erasures and substitution errors (we refer the reader to the following excellent surveys [Mit09, MBT10, CR20, HS21a]).

Codes that can correct insdel errors also attract a lot of attention due to their possible application in designing DNA-based storage systems [GBC$^+$13]. This recent increased interest was paved by substantial progress in synthesis and sequencing technologies. The main advantages of DNA-based storage over classical storage technologies are very high data densities and long-term reliability without an electrical supply. It is thus natural that designing codes for DNA storage and studying the limitations of this model received a lot of attention recently [HSRD17, LSWZY19, HMG19, SH22].

While linear codes are highly desirable, most of the works constructing codes for the insdel model are not linear codes, in contrast to the predominant use of linear codes for Hamming errors. Notable examples of linear codes for Hamming errors include Hamming codes, Reed–Solomon codes, Reed–Muller codes, algebraic-geometry codes, polar codes, Turbo codes, expander codes, and LDPC codes. The reason for the absence of linear codes in the insdel model can be found in works such as [AGFC07, CGHL21], which show that the maximal rate of a linear code capable of correcting insdel errors is significantly worse than that of a nonlinear code correcting the same number of insdel errors. However, linear codes are desirable for many reasons: they offer compact representation, efficient encoding, easier analysis, and in many cases, efficient decoding algorithms. Moreover, linear codes are a central mathematical object that has found widespread applications across various research areas. As such, studying them in the context of insdel errors is important and was the subject of recent works [CGHL21, Che22, CST22, CST23, JZCW23, CJL$^+$23, Liu24].

This work concerns the performance of Reed–Solomon codes in the insdel model. As Reed–Solomon codes are ubiquitous in theory and practice, it is important to understand whether they can also decode from insdel errors. This question received a lot of attention recently [SNW02, WMSN04, TSN07, DLTX21, LT21, CZ22, CST23, CST24, Liu24]. Our result makes a substantial improvement in this line of research. Specifically, we show that "most" RS codes over linear-sized fields have almost optimal capabilities correcting insdel errors. Our methods are inspired by recent works on list-decoding Reed–Solomon codes [GZ23, AGL24], which showed that 'most' RS codes are list-decodable up to capacity over linear-sized alphabets. Specifically, we achieve our result by adapting the random matrix techniques from [GZ23, AGL24] to the insdel setting which required the development of several chain-based structural results of longest common subsequences.

1.1 Previous works

Insdel codes. To offer a wider context, we now discuss some results related to the broader study of codes correcting adversarial insdel errors.

Codes correcting adversarial insdel errors were first considered in the seminal works of Levenshtein [Lev66] and Varshamov and Tenengolts [VT65], the latter of which constructed binary codes that can correct a single insdel error with optimal redundancy. The first efficient construction of asymptotically good binary insdel codes — codes with asymptotically positive rate and tolerating an asymptotically positive insdel fraction — are, as far as we know, given in [SZ99].

Despite these early successes and much recent interest, there are still significant gaps in our understanding of insdel codes, particularly for binary codes. For binary codes, gaps remain for several important questions, including (i) determining the optimal redundancy-error tradeoff and finding explicit binary codes for correcting t insdels, for constant t [GH21, SB20, SGB20]; (ii) finding explicit binary codes for correcting εn insdels for small constant $\varepsilon > 0$ with optimal rate [CJLW22, Hae19], (iii) determining the *zero-rate threshold*, the maximum fraction of errors correctable by an asymptotically positive rate code (we know the answer to be in $[\sqrt{2}-1, 1/2-10^{-40}]$) [GL16, GHL22], and (iv) determining the optimal rate-versus distance tradeoff for deletion codes [Lev66, Lev02, Yas24], among others.

On the other hand, when the alphabet can be larger, as is the case in this work, the picture is more complete. Haeupler and Shahrasbi [HS21b], using a novel primitive called *synchronization strings*, constructed codes of rate $1 - \delta - \varepsilon$ that can correct δ fraction of insdel errors over an alphabet of size $\mathcal{O}_\varepsilon(1)$. This work was the first one to show explicit and efficient constructions of codes that can achieve the Singleton bound in the edit distance setting over constant alphabets. This work also inspired several follow-up works that improved the secondary code parameters and applied synchronization strings to related problems; we refer the reader to [HS21a] for a discussion.

Linear insdel codes. As far as we know, the first study about the performance of linear codes against insdel errors is due to [AGFC07]. Specifically, they showed that *any* linear code that can correct one deletion must have rate at most 1/2. Note that this result shows that linear codes are provably worse than nonlinear codes in the context of insdel errors. Indeed, nonlinear can achieve rate close to 1 whereas linear codes have rate $\leq 1/2$.

Nevertheless, as described in the introduction, studying the performance of linear codes against insdel errors is an important question that indeed has been studied in recent years. The basic question of whether there exist good linear codes for the insdel model was first addressed in the work of Cheng, Guruswami, Haeupler, and Li [CGHL21]. They showed that there are linear codes of rate $R = (1-\delta)/2 - h(\delta)/\log_2(q)$ that can correct a δ fraction of insdel errors. They also established two upper bounds on the rate of linear codes that can correct a δ fraction of deletions. First, they proved the half-Plotkin bound ([CGHL21, Theorem 5.1]), which states that every linear code over a finite field $\mathbb{F}_q$ capable of correcting a δ-fraction of insdel errors has rate at most $\frac{1}{2}\left(1 - \frac{q}{q-1}\delta\right) + o(1)$. Then, they established the following alphabet-independent bound,

Theorem 1 (Half-Singleton bound [CGHL21, Corollary 5.2])**.** *Every linear insdel code which is capable of correcting a δ fraction of deletions has rate at most $(1-\delta)/2 + o(1)$.*

Remark. The following non-asymptotic version of the half-Singleton bound can be derived from the proof of [CGHL21, Corollary 5.2]: An $[n, k]$ linear code can correct at most $n - 2k + 1$ insdel errors.

The question of constructing explicit linear codes that are efficiently decodable has also been studied and there are explicit and efficient constructions of linear codes over *small alphabets* (such as binary) that are asymptotically good [CST22, CJL$^+$23]. However, we note that over *small alphabets*, determining the optimal rate-error tradeoff and finding optimal explicit constructions remains an open problem.

Reed–Solomon codes against insdel errors. In this work, we focus on *Reed–Solomon* codes, which are among the most well-known codes. These codes are defined as follows.

Definition 2 (Reed–Solomon code)**.** Let $\alpha_1, \alpha_2, \ldots, \alpha_n$ be distinct elements in the finite field $\mathbb{F}_q$ of order q. For $k < n$, the $[n, k]_q$ *Reed–Solomon* (RS) code of dimension k and block length n associated with the evaluation vector $(\alpha_1, \ldots, \alpha_n) \in \mathbb{F}_q^n$ is defined to be the set of codewords

$$\mathrm{RS}_{n,k}(\alpha_1, \ldots, \alpha_n) := \{(f(\alpha_1), \ldots, f(\alpha_n)) \mid f \in \mathbb{F}_q[x],\ \deg f < k\}.$$

The performance of RS codes against insdel errors was first considered in [SNW02] in the context of traitor tracing. In [WMSN04], the authors constructed a $[5, 2]_q$ RS code correcting a single deletion. In [TSN07], an $[n, k]$ generalized RS code that is capable of correcting $\log_{k+1} n - 1$ deletions was constructed. Several constructions of two-dimensional RS codes correcting $n - 3$ insdel errors are given in [DLTX21, LT21]. Note that the half-Singleton bound states that an $[n, 2]$ code can correct at most $n - 3$ deletions. In [CST23], it was shown that for an $[n, 2]_q$ RS code to decode from $n - 3$ insdel errors, it must be that $q = \Omega(n^3)$. On the other hand, [CST24] gave an explicit construction with $q = O(n^3)$. Thus, for the special case of two-dimensional RS codes, there is a complete characterization of RS codes that achieve the half-Singleton bound.

For $k > 2$, much less is known. It was shown in [CST23] that over large enough fields, there exist RS codes that exactly achieve the half-Singleton bound. Specifically,

Theorem 3 ([CST23, Theorem 16])**.** *Let n and k be positive integers such that $2k - 1 \leq n$. For $q = 2^{\Theta(n)}$, there exists an $[n, k]_q$ RS code that can decode from $n - 2k + 1$ insdel errors.*[1]

[1] The alphabet size in [CST23, proof of Theorem 16] is actually $q = \binom{n}{2k-1}^2 \cdot k^2 + n^2$, which is better, especially for sublinear $k = o(n)$. However, given that the primary parameter regime of interest in this paper is $k = \Theta(n)$, we state this simplified version for brevity.

In both theoretical and practical scenarios, codes over smaller alphabets tend to be more valuable, but the field size above is very large. Whether Reed–Solomon codes over significantly smaller fields are capable of correcting insdel errors up to the half-Singleton bound remained an interesting and important open problem, which we address in this work.

List-decodable (Hamming) codes. As we mentioned, we port some crucial ideas from a different context (List-decoding under Hamming distance) to build our main results in the insertion-deletion world. To facilitate a better understanding, we provide a concise summary of recent and significant advancements in the field of list decoding of codes under the Hamming metric.

The notion of *list decoding* was introduced by Elias and Wozencraft [Eli57, Woz58] in 1950s as a natural generalization of unique decoding. Briefly, a code exhibits satisfactory list decodability if its codewords are widely dispersed throughout the entire space, that is, there are not too many codewords located within a single ball under the Hamming metric. After the seminal work of Sudan [Sud97] and Guruswami–Sudan [GS98], which provided efficient algorithms for list decoding RS codes up to the Johnson bound [Joh62], the issue of understanding the list decodability of RS codes beyond the Johnson bound became very important. A long line of works [RW14, ST20, GLS+22, FKS22, GST22, BGM23, GZ23, AGL24, BDGZ24, RZVW24, GXYZ24] have made significant advancements in the understanding of list decodability for RS and related codes. Specifically, recent works [BGM23, GZ23, AGL24] have shown that "most" RS codes over exponential-sized alphabets (in terms of the code length) are optimally list decodable, and "most" RS codes over linear-sized alphabets are in fact almost optimally list decodable.

1.2 Our results

When $(\alpha_1, \dots, \alpha_n)$ is uniformly distributed over the set of all n-tuples of distinct elements in $\mathbb{F}_q$, we say the code $\mathrm{RS}_{n,k}(\alpha_1, \dots, \alpha_n)$ over $\mathbb{F}_q$ is a *random RS code* of dimension k and length n over $\mathbb{F}_q$. In this work, we show that random RS codes over alphabets of size $O_\varepsilon(n)$, with high probability, approach the half-Singleton bound for insdel errors. Specifically,

Theorem 4 (Informal, Details in Theorem 43)**.** *Let $\varepsilon \in (0,1)$, and let n and k be positive integers. For a prime power q satisfying $q \geq n + 2^{\mathrm{poly}(1/\varepsilon)}k$, with high probability, a random RS code of dimension k and length n over $\mathbb{F}_q$ corrects at least $(1-\varepsilon)n - 2k + 1$ adversarial insdel errors.*

For the constant rate regime, $R = \Theta(1)$, our result exponentially improves the alphabet size of Con, Shpilka, and Tamo [CST23], where they have $q = 2^{\Theta(n)}$. As a warmup to this result, we prove a weaker but more straightforward result (Theorem 21), which establishes Theorem 4 for $q = 2^{O(1/\varepsilon)}n^2$.

1.3 Proof overview

We outline the proof of our main theorem in this section. First, we review the proof of Theorem 3 [CST23] that achieves the half-Singleton bound with exponential-sized alphabets. We slightly modify the proof's presentation to parallel our proofs. Second, we show how to prove a weaker version of Theorem 4 (Theorem 21) with quadratic-sized alphabets. Lastly, we describe how to prove Theorem 4 that achieves linear-sized alphabets. Throughout this overview, let C be a random Reed–Solomon code of length n and dimension k, where the tuple of evaluation points $(\alpha_1, \dots, \alpha_n)$ is sampled uniformly from all n-tuples of pairwise-distinct elements from $\mathbb{F}_q$.

Warmup: exponential alphabet size. We start with the argument from [CST23] that proves that Reed–Solomon codes achieve the half-Singleton bound over exponential-sized alphabets $q = 2^{\Theta(n)}$. Let $\ell \overset{\text{def}}{=} 2k-1$. We want (with high probability over $\alpha_1, \dots, \alpha_n$) that our code C corrects $n-2k+1 = n-\ell$ insdel errors, or equivalently, has pairwise small LCS: $\mathrm{LCS}(c, c') < \ell$ for any two distinct codewords $c, c' \in C$.

The key observation is that, if our code C fails to correct $n-2k+1$ insdels, then there exist indices $I_1 < \dots < I_\ell$ and $J_1 < \dots < J_\ell$ such that the following matrix, which we call the V*-matrix*, is *bad*, meaning it does not have full column rank.

$$V_{k,\ell,I,J}(\alpha_1, \alpha_2, \cdots, \alpha_n) := \begin{pmatrix} 1 & \alpha_{I_1} & \cdots & \alpha_{I_1}^{k-1} & \alpha_{J_1} & \cdots & \alpha_{J_1}^{k-1} \\ 1 & \alpha_{I_2} & \cdots & \alpha_{I_2}^{k-1} & \alpha_{J_2} & \cdots & \alpha_{J_2}^{k-1} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 1 & \alpha_{I_\ell} & \cdots & \alpha_{I_\ell}^{k-1} & \alpha_{J_\ell} & \cdots & \alpha_{J_\ell}^{k-1} \end{pmatrix}, \tag{1}$$

Indeed, if C fails to correct $n-2k+1$ insdels, there exist two distinct polynomials $f(X) = f_0 + f_1 X + \dots + f_{k-1}X^{k-1}$ and $f'(X) = f'_0 + f'_1 X + \dots + f'_{k-1}X^{k-1}$, such that the $(I_1, \dots, I_\ell)$-indexed subsequence of the codeword for f equals the $(J_1, \dots, J_\ell)$-indexed subsequence of the codeword for f'. In that case,

$$\begin{pmatrix} 1 & \alpha_{I_1} & \cdots & \alpha_{I_1}^{k-1} & \alpha_{J_1} & \cdots & \alpha_{J_1}^{k-1} \\ 1 & \alpha_{I_2} & \cdots & \alpha_{I_2}^{k-1} & \alpha_{J_2} & \cdots & \alpha_{J_2}^{k-1} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 1 & \alpha_{I_\ell} & \cdots & \alpha_{I_\ell}^{k-1} & \alpha_{J_\ell} & \cdots & \alpha_{J_\ell}^{k-1} \end{pmatrix} \cdot \begin{pmatrix} f_0 - f'_0 \\ f_1 \\ \vdots \\ f_{k-1} \\ -f'_1 \\ \vdots \\ -f'_{k-1} \end{pmatrix} = 0, \tag{2}$$

so the V-matrix is bad.

Now, it suffices to show, with high probability, that all V-matrices are good (have full collumn rank). However, by considering the determinant of the V-matrix (which is square as $\ell = 2k-1$), the probability that one V-matrix is bad is at most $\frac{k(k-1)}{q-n}$ by the Schwartz–Zippel lemma.[2] A V-matrix is defined by the indices of the subsequences $I_1 < \dots < I_\ell$ and $J_1 < \dots < J_\ell$, so there are at most 2^{2n} possible V-matrices. Hence, by the union bound, the probability that some V-matrix is bad is at most $2^{2n} \cdot \frac{k(k-1)}{q-n}$. Hence, for sufficiently large exponential alphabet sizes $q \geq 2^{\Theta(n)}$, our code corrects $n-2k+1$ insdel errors with high probability, as desired.

Quadratic alphabet size. We now discuss how to improve the field size bound, first to quadratic, and then to linear.

Our main idea, inspired by [GZ23, AGL24], is to use "slackness" in the coding parameters to amplify the probability that a V-matrix is bad. The above warmup gives random RS codes that correct $n-(2k-1)$ errors, *exactly* achieving the half-Singleton bound. We relax the guarantee, and now ask for a random RS code to correct $n-(2k-1)-\varepsilon n$ errors, *approaching* the half-Singleton bound. Now, the corresponding V-matrix is a $\ell \times (2k-1)$ matrix, for $\ell \overset{\text{def}}{=} (2k-1)+\varepsilon n$. For this relaxation, we show the probability V-matrix is bad is at most $\left(\frac{kn}{q-n}\right)^{\Theta(\varepsilon n)}$, rather than $\frac{k(k-1)}{q-n}$.

[2]An important detail here, which [CST23] proves, is that (to apply the Schwartz-Zippel lemma) the determinant needs to be symbolically nonzero.

First, we discuss a toy problem that illustrates this probability amplification. Consider the toy problem of independently picking ℓ *uniformly* random row vectors $v_1, \cdots, v_\ell \in \mathbb{F}_q^{2k-1}$ to form an $\ell \times (2k-1)$ matrix M, which we want to have full column rank. If we choose $\ell = 2k-1$, then the probability that M has full column rank is bounded by a function that is $\Theta(1/q)$, and this happens only if each v_i is not in the span of $v_1, \ldots, v_{i-1}$. However, suppose we choose $\ell = (2k-1) + \varepsilon n$ for some small $\varepsilon > 0$. In this case, we could afford εn "faulty" vectors v_i , i.e., v_i may be in the span of previous vectors, in which case we just skip it and consider the next vector. The probability that the matrix M has full column rank is then exponentially small, $1/q^{\Omega(\varepsilon n)}$.

Now we outline the formal proof of this probability amplification, captured in Corollary 20.

Corollary (Corollary 20, informal). *Let $\varepsilon \in [0,1]$, $\ell = (2k-1) + \varepsilon n$, and $r = \varepsilon n/2$. Let $I, J \in [n]^\ell$ be two increasing subsequences that agree on at most $k-1$ coordinates*[3]*. Then,*

$$\Pr\left[\textit{Matrix } V_{k,\ell,I,J}(\alpha_1, \ldots, \alpha_n) \textit{ is bad}\right] \le \left(\frac{2n(k-1)}{q-n+1}\right)^r.$$

At the highest level, our proof of Corollary 20 is a union bound over "certificates." For all evaluation points $(\alpha_1, \ldots, \alpha_n) \in \mathbb{F}_q^n$ where $V_{k,\ell,I,J}$ is bad, we show that there is a *certificate* $(i_1, \ldots, i_r) \in [n]^r$ of distinct indices in $[n]$ (Lemma 18) that intuitively "attests" that the matrix $V_{k,\ell,I,J}$ is bad.

We generate the certificate $(i_1, \ldots, i_r)$ deterministically from the evaluations $\alpha_1, \ldots, \alpha_n$ using Algorithm 1. We compute the certificate one index i_j at a time. Given indices $i_1, \ldots, i_{j-1}$, define index i_j as follows: let A_j be the top $(2k-1) \times (2k-1)$ square submatrix of $V_{k,\ell,I,J}^{\{i_1,\ldots,i_{j-1}\}}$ — the V-matrix $V_{k,\ell,I,J}$ after removing rows containing any of variables $X_{i_1}, X_{i_2}, \ldots, X_{i_{j-1}}$ — and let i_j be the smallest index such that $A_j|_{X_1=\alpha_1,\ldots,X_{i_j}=\alpha_{i_j}}$ is not full column rank (we call i_j a *faulty index*, Definition 17). Since A_j is a full rank submatrix of a bad V-matrix[4], $A_j|_{X_1=\alpha_1,\ldots,X_n=\alpha_n}$ is not full rank, so index i_j always exists. Hence, we can keep generating indices i_j as long as the truncated V-matrix, $V_{k,\ell,I,J}^{\{i_1,\ldots,i_{j-1}\}}$, has at least $2k-1$ rows. By definition, each X_i participates in at most 2 rows of a V-matrix, so we get a certificate of length at least $\left\lfloor \frac{\ell-(2k-1)}{2} \right\rfloor + 1 \ge \varepsilon n/2 = r$.

We then show (Lemma 19), for any certificate $(i_1, \ldots, i_r)$, the probability that the V-matrix has certificate $(i_1, \ldots, i_r)$ is exponentially small. Conditioned on A_j being full rank with $X_1 = \alpha_1, \ldots, X_{i_j-1} = \alpha_{i_j-1}$, the probability that A_j becomes not-full-rank when setting $X_{i_j} = \alpha_{i_j}$ is at most $\frac{2(k-1)}{q-n}$: α_{i_j} is uniformly random over at least $q-n$ field elements, and the degree of X_{i_j} in the determinant of A_j is at most $2(k-1)$. This event needs to happen r times, and it is possible to run the conditional probabilities in the correct order to conclude that the probability of generating a particular certificate $(i_1, \ldots, i_r)$ is at most $\left(\frac{2(k-1)}{q-n}\right)^r$.

Since there are at most n^r certificates, the total probability that that a particular V-matrix is bad is at most $n^r \cdot \left(\frac{2(k-1)}{q-n}\right)^r$. This is at most 2^{-3n} for sufficiently large quadratic alphabet sizes $q = 2^{\Theta(1/\varepsilon)} \cdot n^2$. For such q, by a union bound over the at-most-2^{2n} V-matrices, with probability at least $1 - 2^{-n}$, the code C corrects $n - \ell$ deletions, thus approaching the half-Singleton bound with quadratic alphabet size.

Linear alphabet size. To improve the alphabet size to linear, we modify the certificate argument so that the number of certificates is only $\binom{n}{r}$, rather than n^r. The idea is to force the certificates

[3]More precisely, $I_i = J_i$ for at most $k-1$ values of i. This technical condition ensures that the V-matrix is symbolically full column rank.

[4]The full-rank part needs to be checked, but follows from [CST23].

to have increasing coordinates $i_1 < i_2 < \cdots < i_r$ (this does not hold automatically).[5]

First, we preprocess the V-matrix by removing some of its rows (equivalently, we remove elements from I and J), so that the remaining matrix can be partitioned into "blocks" of length at most $O(1/\varepsilon)$. Crucially, the variables in a block appear *only* in that block, so that the blocks partition the variables $X_1, \dots, X_n$ (in the proof, these blocks are given by *chains*, Definition 23). Proving this step requires establishing structural properties of longest common subsequences.

We then generate our certificates in a more careful way. We remove the largest $\Omega_\varepsilon(n)$ of these blocks from our V-matrix to create a *bank* of blocks, and, we reindex the variables so that the banked blocks have the highest-indexed variables.[6] As in Algorithm 1, we choose A_1 to be the top $(2k-1)\times(2k-1)$ submatrix of the V-matrix — this time, after removing the blocks in the bank —, and for all j, we again choose i_j as the smallest index such that setting $X_{i_j} = \alpha_{i_j}$ makes A_j not full rank. However, we choose the matrices $A_2, A_3, \dots$ more carefully. After choosing i_j, we let A_{j+1} be a submatrix of $V_{k,\ell,I,J}$ that "re-indeterminates" the matrix A_j: we remove from A_j the block containing variable X_{i_j}, and replace it with an "equivalent" new block from our bank — possibly truncating the new block, if the new block is longer than the old block — to get A_{j+1}. This results in a matrix A_{j+1} "equivalent" to A_j; it is the same polynomial matrix up to permuting the rows and relabeling the indeterminates. Since this matrix A_{j+1} is an equivalent, "re-indeterminated" version of A_j, we must have $A_{j+1}|_{X_1=\alpha_1,\dots,X_{i_j}=\alpha_{i_j}}$ is full column rank, so we have $i_j < i_{j+1}$, which is our desired property for our certificates. Further, since our bank has at least $\Omega_\varepsilon(n)$ blocks, we can "re-indeterminate" at least $\Omega_\varepsilon(n)$ times, giving a certificate of length $r = \Omega_\varepsilon(n)$.

Since our certificates now satisfy $i_1 < \cdots < i_r$, the number of certificates is at most $\binom{n}{r}$, the probability C fails to correct $n-\ell$ deletions is only $\binom{n}{r}\left(\frac{2(k-1)}{q-n}\right)^r$, which is exponentially small 2^{-3n} for sufficiently large linear alphabet sizes $q = n + \Theta_\varepsilon(k)$. Again, a union bound over (at most 2^{2n}) V-matrices gives that, with high probability, our code C corrects $n-\ell$ deletions, thus approaching the half-Singleton bound with linear-sized alphabets.

1.4 Future research directions

We conclude the introduction by outlining several open questions for future research:

1. *Lower bounds on the field size.* In [CST23], it was demonstrated that there exist RS codes over fields of size $n^{O(k)}$ that exactly attain the half-Singleton bound. This paper shows that the field size can be significantly reduced if we only need to get ε-close to the half-Singleton bound. A remaining open question is to prove a lower bound on the field size of linear codes, not just RS codes, that achieve (or get close to) the half-Singleton bound.

2. *Explicit constructions.* The results presented in this paper are existential; specifically, we demonstrate the existence of RS codes capable of correcting insdel errors with an almost optimal rate-distance tradeoff. An important question remains: How to provide explicit constructions of RS codes that achieve these parameters?

3. *Decoding algorithms.* One of the primary advantages of RS codes in the Hamming metric is their efficient decoding algorithms. Currently, as far as we know, there are no efficient algorithms for RS codes that correct insdel errors. Therefore, a crucial open question is how to design efficient decoding algorithms for RS codes handling insdel errors.

[5]For convenience, in the actual proof, our certificate is slightly different. Instead of $1 \le i_1 < i_2 < \cdots < i_r \le n$, we take certificates $0 \le i_1 \le i_2 \le \cdots \le i_r \le 2k-2$. The i_j's have slightly different meaning, but the idea is the same.

[6]This is acceptable, since we don't use the original ordering on the variables after this point.

4. *Affine codes.* In [CGHL21, Theorem 1.5], it was demonstrated that affine codes outperform linear codes in correcting insdel errors. Specifically, efficient affine binary codes of rate $1-\varepsilon$ were constructed to correct $\Omega(\varepsilon^3)$ insdel errors. By efficient codes, we mean explicit construction of codes with efficient encoding and decoding algorithms. An immediate open question is to construct efficient affine code with improved rate-distance trade-offs. Moreover, considering that RS codes achieve the half-Singleton bound, a natural question arises: can affine RS codes perform even better? In particular, can they approach the Singleton bound?

Acknowledgments

This work was conducted while all four authors were visiting the Simons Institute for the Theory of Computing at UC Berkeley. The authors would like to thank the institute for its support and hospitality. Additionally, Z. Zhang wishes to thank Prof. Xin Li for the helpful discussions and suggestions. R. Li is supported by NSF grant CCF-2347371.

2 Notation and Preliminaries

Define $\mathbb{N} = \{0,1,2,\dots,\}$, $\mathbb{N}^+ = \{1,2,\dots\}$, and $[n] = \{1,2,\dots,n\}$ for $n \in \mathbb{N}$. Throughout the paper, $\mathbb{F}_q$ denotes the finite field of order q.

Denote by $\mathbb{F}_q[X_1,\dots,X_n]$ the ring of polynomials in $X_1,\dots,X_n$ over $\mathbb{F}_q$ and by $\mathbb{F}_q(X_1,\dots,X_n)$ the field of fractions of that ring. More generally, for a collection of variables X_i indexed by a set I, denote by $\mathbb{F}_q[X_i : i \in I]$ the ring of polynomials in these variables over $\mathbb{F}_q$. The value of $f \in \mathbb{F}_q[X_i : i \in I]$ at $(\alpha_i)_{i\in I}$ is denoted by $f(\alpha_i : i \in I)$, or $f(\alpha_1,\dots,\alpha_n)$ when $I = [n]$.

The degree $\deg(f)$ of a nonzero multivariate polynomial $f \in \mathbb{F}_q[X_i : i \in I]$ refers to its total degree. And for $i \in [n]$, $\deg_{X_i}(f)$ denotes the degree of f in X_i, where the other variables are viewed as constants.

We also need the following notation, adapted from [GZ23], regarding the partial assignment of a symbolic matrix.

Definition 5 (Partial assignment)**.** Let A be a matrix over $\mathbb{K} := \mathbb{F}_q(X_1,\dots,X_n)$ such that the entries of A are in $\mathbb{F}_q[X_1,\dots,X_n]$. For $i \in \{0,1,\dots,n\}$ and $\alpha_1,\dots,\alpha_i \in \mathbb{K}$, denote by $A|_{X_1=\alpha_1,\dots,X_i=\alpha_i}$ the matrix obtained from A by substituting α_j for X_j for $j = 1,\dots,i$. More generally, for $I \subseteq [n]$ and a tuple $(\alpha_i)_{i\in I} \in \mathbb{K}^I$, denote by $A|_{X_i=\alpha_i \text{ for } i\in I}$ the matrix obtained from A by substituting α_i for X_i for $i \in I$.

We need the following variation of the Schwarz-Zippel Lemma.

Lemma 6. *Let $Q(X_1,\dots,X_n) \in \mathbb{F}_q[X_1,\dots,X_n]$ be a nonzero polynomial such that $\deg_{X_i}(Q) \le d$ for $i \in [n]$. Let $T \subseteq \mathbb{F}_q$ be a set of size at least n, and let $\alpha = (\alpha_1,\dots,\alpha_n)$ be uniformly distributed over the set of n-tuples with distinct coordinates in T. Then $\Pr[Q(\alpha_1,\dots,\alpha_n) = 0] \le \frac{nd}{|T|-n+1}$.*

Proof. For $i \in [n]$, let E_i be the event that $Q(\alpha_1,\dots,\alpha_{i-1},X_i,\dots,X_n) \neq 0$ but $Q(\alpha_1,\dots,\alpha_i,X_{i+1},\dots,X_n) = 0$. As $Q \neq 0$, the event $Q(\alpha_1,\dots,\alpha_n) = 0$ occurs precisely when one of $E_1,\dots,E_n$ occurs. By the union bound, it suffices to prove that $\Pr[E_i] \le \frac{d}{|T|-n+1}$.

Fix $i \in [n]$. Further fix $\alpha_1,\dots,\alpha_{i-1} \in \mathbb{F}_q$ such that $Q(\alpha_1,\dots,\alpha_{i-1},X_i,\dots,X_n) \neq 0$. It suffices to prove that E_i occurs with probability at most $\frac{d}{|T|-n+1}$ after fixing such $\alpha_1,\dots,\alpha_{i-1}$. (If such $\alpha_1,\dots,\alpha_{i-1}$ do not exist, then $\Pr[E_i] = 0$ and we are done.)

Now α_i is uniformly distributed over the set $T' := T \setminus \{\alpha_1,\dots,\alpha_{i-1}\}$. Let $Q^* := Q(\alpha_1,\dots,\alpha_{i-1},X_i,\dots,X_n) \neq 0$. View Q^* as a polynomial in $X_{i+1},\dots,X_n$ over the ring $\mathbb{F}_q[X_i]$. Let

$Q_0^* \in \mathbb{F}_q[X_i]$ be the coefficient of a nonzero term of Q^*. Then $Q_0^* \neq 0$, and $\deg(Q^*) \leq \deg_{X_i}(Q) \leq d$. Therefore,

$$\Pr[Q_0^*(\alpha_i) = 0] \leq \frac{d}{|T'|} \leq \frac{d}{|T| - n + 1}. \tag{3}$$

Note that $Q_0^*(\alpha_i)$ is the coefficient of a term of $Q^*(\alpha_i, X_{i+1}, \ldots, X_n) = Q(\alpha_1, \ldots, \alpha_i, X_{i+1}, \ldots, X_n)$. Therefore, the latter is zero only if $Q_0^*(\alpha_i) = 0$. So by (3), E_i occurs with probability at most $\frac{d}{|T|-n+1}$, as desired. □

We recall the notion of a subsequence and a longest common subsequence.

Definition 7. A *subsequence* of a string s is a string obtained by removing some (possibly none) of the symbols in s.

Definition 8. Let s, s' be strings over an alphabet Σ. A *longest common subsequence* between s and s', is a subsequence of both s and s', of maximal length. We denote by $\mathrm{LCS}(s, s')$ the length of a longest common subsequence.

The *edit distance* between s and s', denoted by $\mathrm{ED}(s, s')$, is the minimal number of insertions and deletions needed in order to turn s into s'.

It is well known that the insdel correction capability of a code is determined by the LCS of its codewords. Specifically,

Lemma 9. *A code C can correct δn insdel errors if and only* $\mathrm{LCS}(c, c') \leq n - \delta n - 1$ *for any distinct $c, c' \in C$.*

We give the proof for completeness.

Proof. A code can correct δn insdel errors if and only if for any two distinct codewords c, c', it holds $\mathrm{ED}(c, c') \geq 2\delta n + 1$. The claim follow by the well-known relation between edit distance and LCS which states that $\mathrm{ED}(s, s') = |s| + |s'| - 2\mathrm{LCS}(s, s')$ [CR03, Lemma 12.1]. □

We now adopt two definitions and a lemma from [CST23]. These will establish the algebraic conditions ensuring that an RS code can correct insdel errors.

Definition 10 (Increasing subsequence)**.** We call $I = (I_1, \ldots, I_\ell) \in [n]^\ell$ an *increasing subsequence* if it holds that $I_1 < I_2 < \cdots < I_\ell$, where ℓ is called the *length* of I.

Definition 11 (V-matrix)**.** For positive integers ℓ, k and increasing subsequences $I = (I_1, \ldots, I_\ell), J = (J_1, \ldots, J_\ell) \in [n]^\ell$ of length ℓ, define the $\ell \times (2k-1)$ matrix

$$V_{k,\ell,I,J}(X_1, X_2, \cdots, X_n) := \begin{pmatrix} 1 & X_{I_1} & \cdots & X_{I_1}^{k-1} & X_{J_1} & \cdots & X_{J_1}^{k-1} \\ 1 & X_{I_2} & \cdots & X_{I_2}^{k-1} & X_{J_2} & \cdots & X_{J_2}^{k-1} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 1 & X_{I_\ell} & \cdots & X_{I_\ell}^{k-1} & X_{J_\ell} & \cdots & X_{J_\ell}^{k-1} \end{pmatrix},$$

over the field $\mathbb{F}_q(X_1, \ldots, X_n)$. We call $V_{k,\ell,I,J}(X_1, X_2, \cdots, X_n)$ a *V-matrix*.

The following lemma states that if a Reed–Solomon code $\mathsf{RS}_{n,k}(\alpha_1, \ldots, \alpha_n) \subseteq \mathbb{F}_q^n$ cannot correct ℓ insdel errors, then we can identify a specific V-matrix that does not have full column rank. The proof of this lemma is identical to the proof of Proposition 2.1 in [CST23]. However, we include it here for completeness.

Lemma 12. *Let $\ell \geq 2k-1$. Consider the $[n,k]$ Reed–Solomon code $\mathsf{RS}_{n,k}(\alpha_1,\ldots,\alpha_n) \subseteq \mathbb{F}_q^n$ associated with an evaluation vector $\vec{\alpha} := (\alpha_1,\ldots,\alpha_n) \in \mathbb{F}_q^n$. If the code cannot correct arbitrary $n-\ell$ insdel errors, then there exist two increasing subsequences $I = (I_1,\ldots,I_\ell), J = (J_1,\ldots,I_\ell) \in [n]^\ell$ that agree on at most $k-1$ coordinates such that matrix $V_{k,\ell,I,J}|_{X_1=\alpha_1,\ldots,X_n=\alpha_n}$ does not have full column rank.*

Proof. Suppose the code $\mathsf{RS}_{n,k}(\alpha_1,\ldots,\alpha_n) \subseteq \mathbb{F}_q^n$ can not recover from $n-\ell$ insdel errors, then by Lemma 9, there exist two distinct codewords $c_i := (f_i(\alpha_1),\ldots,f_i(\alpha_n))$ $(i=1,2)$ such that $\mathsf{LCS}(c_1,c_2) \geq \ell$. Therefore, by definition, we have two increasing subsequences $I,J \in [n]^\ell$ such that

$$f_1(\alpha_{I_s}) = f_2(\alpha_{J_s}) \quad \text{for } s = 1,2,\ldots,\ell, \tag{4}$$

where $f_i := \sum_{j=0}^{k-1} f_i^{(j)} X^j \in \mathbb{F}_q[X]$ $(i=1,2)$ are two distinct polynomials of degree at most $k-1$. To verify that I and J agree on at most $k-1$ coordinates, assume that this is not true, i.e., there exist distinct $i_1,\ldots,i_k \in [\ell]$ such that $I_{i_j} = J_{i_j}$ for $j \in [k]$. Then by (4), $f_1 - f_2$ vanishes at the k distinct points $\alpha_{I_{i_1}},\ldots,\alpha_{I_{i_k}}$. But this contradicts the fact that f_1 and f_2 are distinct polynomials of degree at most $k-1$.

Consider the vector $\vec{v} := \left(f_1^{(0)} - f_2^{(0)}, f_1^{(1)}, f_1^{(2)}, \ldots, f_1^{(k-1)}, -f_2^{(1)}, -f_2^{(2)}, \ldots, -f_2^{(k-1)}\right) \in \mathbb{F}_q^{2k-1}$, which is non-zero. By the algebraic relations stated in (4), we have $V_{k,\ell,I,J}|_{X_1=\alpha_1,\ldots,X_n=\alpha_n} \cdot \vec{v} = 0$. Therefore, the matrix $V_{k,\ell,I,J}|_{X_1=\alpha_1,\ldots,X_n=\alpha_n}$ does not have full column rank. □

A key technical lemma regarding the V-matrices that was proved in [CST23] is the following.

Lemma 13 (Proposition 2.4, [CST23])**.** *Let $I,J \in [n]^{2k-1}$ be two increasing subsequences that agree on at most $k-1$ coordinates. Then, we have* $\det(V_{k,2k-1,I,J}(X_1,X_2,\ldots,X_n)) \neq 0$ *as a multivariate polynomial in $\mathbb{F}_q[X_1,X_2,\ldots,X_n]$.*

Informally speaking, by using this lemma, and by considering all the V-matrices, the authors of [CST23] showed by using the Schwarz-Zippel lemma that there exists an assignment $(\alpha_1,\ldots,\alpha_n)$ over a large enough field for which $\det(V_{k,\ell,I,J}|_{X_1=\alpha_1,\ldots,X_n=\alpha_n})$ is nonzero for all pairs I,J of length $2k-1$ that agree on at most $k-1$ coordinates.

For convenience, we introduce the following definition.

Definition 14 (Selecting a subset of coordinates)**.** For an increasing subsequence $I \in [n]^\ell$ and $P = \{i_1,\ldots,i_{\ell'}\} \subseteq [\ell]$ with $i_1 < \cdots < i_{\ell'}$, denote by I^P the increasing subsequence $(I_{i_1},\ldots,I_{i_{\ell'}})$.

The following is a corollary of Lemma 13.

Corollary 15. *Let $I,J \in [n]^\ell$ with $\ell \geq 2k-1$ be two increasing subsequences that agree on at most $k-1$ coordinates. Let $P \subseteq [\ell]$ be a set of size $2k-1$. Then* $\det(V_{k,2k-1,I^P,J^P}) \neq 0$. *Additionally, for $i \in [n]$, we have* $\deg_{X_i}(\det(V_{k,2k-1,I^P,J^P})) \leq 2(k-1)$.

Proof. Note that $I^P, J^P \in [n]^{2k-1}$ are two increasing subsequences that agree on at most $k-1$ coordinates. So the first claim follows immediately from Lemma 13. The second claim follows from the facts that every entry of $V_{k,2k-1,I^P,J^P}$ has the form X_i^j with $j \leq k-1$ and that each X_i appears in at most two rows. □

3 Achieving Quadratic Alphabet Size

In this section, we present a warm-up result, Theorem 21, which achieves the alphabet size $O_\varepsilon(n^2)$. Our proof resembles the proof by Guo and Zhang [GZ23] showing that random RS codes achieve list decodability over fields of quadratic size.

3.1 Full rankness of $V_{k,\ell,I,J}$ under a random assignment

Firstly, we introduce the following definition, $V^B_{k,\ell,I,J}$, which is a submatrix of $V_{k,\ell,I,J}$ obtained by deleting certain rows.

Definition 16. Under the notation in Definition 11, for a subset $B \subseteq [n]$, define the matrix $V^B_{k,\ell,I,J}$ to be the submatrix of $V_{k,\ell,I,J}$ obtained by deleting the i-th row for all $i \in [\ell]$ such that $I_i \in B$ or $J_i \in B$. In other words, $V^B_{k,\ell,I,J}$ is obtained from $V_{k,\ell,I,J}$ by deleting all the rows containing powers of X_j for every index $j \in B$.

We also need to define the notion of *faulty indices*. Our definition is a simplification of a similar definition in [GZ23].

Definition 17 (Faulty index)**.** Let $A \in \mathbb{F}_q(X_1, \dots, X_n)^{m\times s}$ be a matrix, where $m \geq s$, and let A' be the $s \times s$ submatrix consisting of the first s rows of A. Suppose the entries of A are in $\mathbb{F}_q[X_1, \dots, X_n]$. For $\alpha_1, \dots, \alpha_n \in \mathbb{F}_q$, we say $i \in [n]$ is the *faulty index* of A (with respect to $\alpha_1, \dots, \alpha_n$) if $\det(A'|_{X_1=\alpha_1,\dots,X_{i-1}=\alpha_{i-1}}) \neq 0$ but $\det(A'|_{X_1=\alpha_1,\dots,X_i=\alpha_i}) = 0$. Note that the faulty index of A is unique if it exists.

Next, we will present an algorithm that, when provided with two increasing subsequences $I, J \in [n]^\ell$ of length $\ell = 2k-1+\lfloor \varepsilon n\rfloor$ that agree on at most $k-1$ coordinates, elements $\alpha_1, \dots, \alpha_n \in \mathbb{F}_q$, and a parameter $r \in \mathbb{N}^+$, attempts to verify whether the matrix $V_{k,\ell,I,J}|_{X_1=\alpha_1,\dots,X_n=\alpha_n}$ has full column rank. If it is unable to confirm this, the algorithm produces either a "FAIL" message or identifies a sequence of faulty indicies $(i_1, \dots, i_r) \in [n]^r$. The algorithm is given as Algorithm 1.

Algorithm 1: `CertifyFullColumnRankness`

Input: $n, k, r \in \mathbb{N}^+$, increasing subsequences $I, J \in [n]^\ell$ with $\ell = 2k-1+\lfloor \varepsilon n\rfloor$ that agree on at most $k-1$ coordinates, and $\alpha_1, \dots, \alpha_n \in \mathbb{F}_q$.

Output: "SUCCESS", "FAIL", or a sequence $(i_1, \dots, i_r) \in [n]^r$.

```
1  $B \leftarrow \emptyset$.
2  for $j = 1$ to $r$ do
3      if rank($V^B_{k,\ell,I,J}$) $< 2k-1$ then
4          Output "FAIL" and halt.
5      else if the faculty index $i \in [n]$ of $V^B_{k,\ell,I,J}$ exists then   // $i$ is unique if it exists
6          $i_j \leftarrow i$ and $B \leftarrow B \cup \{i\}$.
7      else
8          Output "SUCCESS" and halt.
9      end
10 end
11 Output $(i_1, \dots, i_r)$.
```

Lemma 18 (Behavior of Algorithm 1)**.** *Let $\varepsilon \geq 0$. Let $I, J \in [n]^\ell$ be two increasing subsequences that agree on at most $k-1$ coordinates, where $\ell = 2k-1+\lfloor \varepsilon n\rfloor$. Let r be a positive integer such that $r \leq \lceil \frac{\varepsilon n}{2} \rceil$. Then for all $\alpha_1, \dots, \alpha_n \in \mathbb{F}_q$, running Algorithm 1 on the input $I, J, \alpha_1, \dots, \alpha_n$, and r yields one of the following two possible scenarios:*

1. *Algorithm 1 outputs "SUCCESS". In this case, $V_{k,\ell,I,J}|_{X_1=\alpha_1,\dots,X_n=\alpha_n}$ has full column rank.*
2. *Algorithm 1 outputs a sequence of distinct indices $(i_1, \dots, i_r) \in [n]^r$. For each $j \in [r]$, i_j is the faulty index of $V^{B_j}_{k,\ell,I,J}$, where $B_j := \{i_1, \dots, i_{j-1}\}$.*

Proof. We first show that the algorithm never outputs "FAIL". If the algorithm reaches the j-th round of the loop, where $j \in [r]$, then at the beginning of this round, we have $|B| = j - 1 \leq r - 1$. For any $j \in B$, the number of indices $i \in [\ell]$ for which $I_i = j$ or $J_i = j$ is at most two. Therefore, $V^B_{k,\ell,I,J}$ is obtained from $V_{k,\ell,I,J}$ by removing at most $2|B|$ rows. It follows that the number of rows that $V^B_{k,\ell,I,J}$ contains is at least $\ell - 2|B| \geq (2k-1) + \lfloor \varepsilon n \rfloor - 2(r-1) \geq 2k - 1$. Note that the top $(2k-1) \times (2k-1)$ submatrix of $V^B_{k,\ell,I,J}$ equals $V_{k,2k-1,I^P,J^P}$ for some $P \subseteq [\ell]$ of size $2k-1$. So by Corollary 15, the matrix $V^B_{k,\ell,I,J}$ has full column rank, which implies that the algorithm never outputs "FAIL".

Assume the algorithm outputs "SUCCESS" and halts in the j-th round for some $j \in [r]$, which means the faulty index of $V^B_{k,\ell,I,J}$ does not exist in that round. Let M be the top $(2k-1) \times (2k-1)$ submatrix of $V^B_{k,\ell,I,J}$. Then $\det(M|_{X_1=\alpha_1,\ldots,X_n=\alpha_n}) \neq 0$, which implies that $V_{k,\ell,I,J}|_{X_1=\alpha_1,\ldots,X_n=\alpha_n}$ has full column rank.

On the other hand, suppose that the algorithm does not output "SUCCESS". Then it outputs some $(i_1, \ldots, i_r) \in [n]^r$ where i_j is the faulty index of $V^{B_j}_{k,\ell,I,J}$ and $B_j = \{i_1, \ldots, i_{j-1}\}$ for $j \in [r]$. Note that $i_1, \ldots, i_r \in [n]$ must be distinct. Indeed, if an index i is in B_j, then X_i does not appear in $V^{B_j}_{k,\ell,I,J}$, and hence i cannot be the faulty index of $V^{B_j}_{k,\ell,I,J}$. □

The next lemma bounds the probability that Algorithm 1 outputs a particular sequence of faulty indices over random $(\alpha_1, \ldots, \alpha_n)$. The proof follows the same approach as [GZ23, Lemma 4.5].

Lemma 19. *Under the notation and conditions in Lemma 18, suppose $q \geq n$ and $(\alpha_1, \ldots, \alpha_n)$ is chosen uniformly at random from the set of all n-tuples of distinct elements in $\mathbb{F}_q$. Then for any sequence $(i_1, \ldots, i_r) \in [n]^r$, the probability that Algorithm 1 outputs $(i_1, \ldots, i_r)$ on the input I, J, $\alpha_1, \ldots, \alpha_n$, and r is at most $\left(\frac{2(k-1)}{q-n+1}\right)^r$.*

Proof. For $j \in [r]$, define the following:

1. $B_j := \{i_1, \ldots, i_{j-1}\}$.

2. Let A_j be the top $(2k-1) \times (2k-1)$ submatrix of $V^{B_j}_{k,\ell,I,J}$. (The fact that $\det A_j \neq 0$ symbolically is guaranteed by Corollary 15.)

3. Let E_j be the event that $\det(A_j|_{X_1=\alpha_1,\ldots,X_{i_j-1}=\alpha_{i_j-1}}) \neq 0$ but $\det(A_j|_{X_1=\alpha_1,\ldots,X_{i_j}=\alpha_{i_j}}) = 0$.

If Algorithm 1 outputs $(i_1, \ldots, i_r)$, then i_j is the faulty index of $V^{B_j}_{k,\ell,I,J}$ for $j \in [r]$ by Lemma 18 and therefore the events $E_1, \ldots, E_r$ all occur. So it suffies to verify that $\Pr[E_1 \wedge \cdots \wedge E_r] \leq \left(\frac{2(k-1)}{q-n+1}\right)^r$.

Let $(j_1, j_2, \ldots, j_r)$ be a permutation of $(1, 2, \ldots, r)$ such that $i_{j_1} < \cdots < i_{j_r}$, i.e., i_{j_ℓ} is the ℓ-th smallest index among $i_1, \ldots, i_r$ for $\ell \in [r]$. For $\ell \in \{0, 1, \ldots, r\}$, we define $F_\ell := E_{j_1} \wedge \cdots \wedge E_{j_\ell}$, where F_0 is the event that always occurs. Then $F_r = E_{j_1} \wedge \cdots \wedge E_{j_r} = E_1 \wedge \cdots \wedge E_r$. We may assume $\Pr[F_r] > 0$ since otherwise we are done. By definition, if F_ℓ occurs and $\ell' < \ell$, then $F_{\ell'}$ also occurs. So $\Pr[F_\ell] > 0$ for all $\ell \in \{0, 1, \ldots, r\}$. We have that

$$\Pr[E_1 \wedge \cdots \wedge E_r] = \Pr[F_r] = \prod_{\ell=1}^{r} \frac{\Pr[F_\ell]}{\Pr[F_{\ell-1}]} .$$

Thus, to prove $\Pr[E_1 \wedge \cdots \wedge E_r] \leq \left(\frac{2(k-1)}{q-n+1}\right)^r$, it suffices to show that $\frac{\Pr[F_\ell]}{\Pr[F_{\ell-1}]} \leq \frac{2(k-1)}{q-n+1}$ for every $\ell \in [r]$.

Fix $\ell \in [r]$ and let $j = j_\ell$. Let S be the set of all $\beta = (\beta_1, \dots, \beta_{i_j-1}) \in \mathbb{F}_q^{i_j-1}$ such that $\Pr\left[\left(\alpha_{<i_j} = \beta\right) \wedge F_{\ell-1}\right] > 0$, where $\alpha_{<i_j} = \beta$ is a shorthand for $(\alpha_1 = \beta_1) \wedge \cdots \wedge (\alpha_{i_j-1} = \beta_{i_j-1})$. Note that for $\beta \in S$, the event $\left(\alpha_{<i_j} = \beta\right) \wedge F_{\ell-1}$ is simply $\alpha_{<i_j} = \beta$ since $F_{\ell-1} = E_{j_1} \wedge \cdots \wedge E_{j_{\ell-1}}$ depends only on $\alpha_1, \dots, \alpha_{i_{j_{\ell-1}}}$ and as $\alpha_{<i_j} = \beta \in S$, $F_{\ell-1}$ occurs (by the definition of S). We then have

$$\frac{\Pr[F_\ell]}{\Pr[F_{\ell-1}]} = \frac{\sum_{\beta\in S}\Pr\left[\left(\alpha_{<i_j} = \beta\right) \wedge F_\ell\right]}{\sum_{\beta\in S}\Pr\left[\left(\alpha_{<i_j} = \beta\right) \wedge F_{\ell-1}\right]} = \frac{\sum_{\beta\in S}\Pr\left[\left(\alpha_{<i_j} = \beta\right) \wedge E_j\right]}{\sum_{\beta\in S}\Pr\left[\alpha_{<i_j} = \beta\right]}$$
$$\le \max_{\beta\in S} \frac{\Pr\left[\left(\alpha_{<i_j} = \beta\right) \wedge E_j\right]}{\Pr\left[\alpha_{<i_j} = \beta\right]} = \max_{\beta\in S} \Pr\left[E_j \mid \alpha_{<i_j} = \beta\right].$$

Fix $\beta = (\beta_1, \dots, \beta_{i_j-1}) \in S$. We just need to prove that $\Pr\left[E_j \mid \alpha_{<i_j} = \beta\right] \le \frac{2(k-1)}{q-n+1}$. Let

$$Q := \det(A_j|_{X_1=\beta_1,\dots,X_{i_j-1}=\beta_{i_j-1}}) \in \mathbb{F}_q[X_{i_j}, \dots, X_n].$$

If $Q = 0$, then E_j never occurs conditioned on $\alpha_{<i_j} = \beta$ and we are done. So assume $Q \neq 0$. View Q as a polynomial in $X_{i_j+1}, \dots, X_n$ over the ring $\mathbb{F}_q[X_{i_j}]$, and let $Q_0 \in \mathbb{F}_q[X_{i_j}]$ be the coefficient of a nonzero term of Q. Then conditioned on $\alpha_{<i_j} = \beta$, the event E_j occurs only if α_{i_j} is a root of $Q_0 \neq 0$. Note that $\deg Q_0 \le \deg_{X_{i_j}} Q \le \deg_{X_{i_j}}(\det(A_j)) \le 2(k-1)$ by Lemma 15. Also note that conditioned on $\alpha_{<i_j} = \beta$, the random variable α_{i_j} is uniformly distributed over the set $\mathbb{F}_q \setminus \{\beta_1, \dots, \beta_{i_j-1}\}$, whose size is at least $q-n+1$ since $i_j \le n$. It follows that $\Pr\left[E_j \mid \alpha_{<i_j} = \beta\right] \le \frac{2(k-1)}{q-n+1}$, as desired. □

By combining Lemma 18 and Lemma 19 and taking a union bound over the set of possible outputs $(i_1, \dots, i_r)$ of Algorithm 1, we obtain the following corollary.

Corollary 20. *Under the notation and conditions in Lemma 18, suppose $q \ge n$ and $(\alpha_1, \dots, \alpha_n)$ is chosen uniformly at random from the set of all n-tuples of distinct elements in $\mathbb{F}_q$. Then for any positive integer $r \le \lceil \frac{\varepsilon n}{2} \rceil$, we have*

$$\Pr\left[\begin{matrix}\text{The matrix } V_{k,\ell,I,J}|_{X_1=\alpha_1,\dots,X_n=\alpha_n} \\ \text{does not have full column rank}\end{matrix}\right] \le \left(\frac{2n(k-1)}{q-n+1}\right)^r.$$

Proof. By Lemma 19 and the union bound, the probability that Algorithm 1 outputs some sequence $(i_1, \dots, i_r) \in [n]^r$ on the input $I, J, \alpha_1, \dots, \alpha_n$, and r is at most $n^r \cdot \left(\frac{2(k-1)}{q-n+1}\right)^r = \left(\frac{2n(k-1)}{q-n+1}\right)^r$. By Lemma 18, whenever this does not occur, the matrix $V_{k,\ell,I,J}|_{X_1=\alpha_1,\dots,X_n=\alpha_n}$ has full column rank, as desired. □

3.2 Putting it together

We are now ready to prove a weaker version of our main result, which achieves quadratic-sized alphabets.

Theorem 21. *Let $\varepsilon \in (0,1)$ and $n, k \in \mathbb{N}^+$, where $k \le n$. Let q be a prime power such that $q \ge \left(1 + 2 \cdot 2^{6/\varepsilon} k\right) n$. Suppose $(\alpha_1, \dots, \alpha_n)$ is chosen uniformly at random from the set of all n-tuples of distinct elements in $\mathbb{F}_q$. Then with probability at least $1 - 2^{-n} > 0$, the code $RS_{n,k}(\alpha_1, \dots, \alpha_n)$ over $\mathbb{F}_q$ corrects at least $(1-\varepsilon)n - 2k + 1$ adversarial insdel errors.*

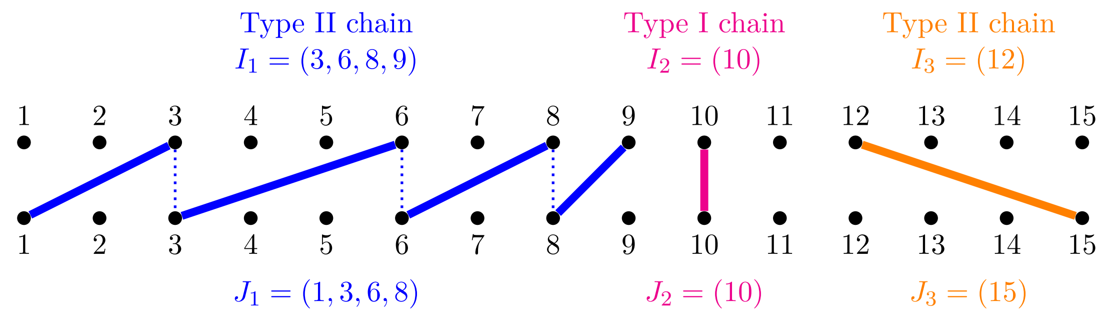


Figure 1: Example of chains

Proof. Let $\ell = (2k-1) + \lfloor \varepsilon n \rfloor$. By Lemma 12, if $\mathsf{RS}_{n,k}(\alpha_1, \ldots, \alpha_n) \subseteq \mathbb{F}_q^n$ fails to correct $n - \ell$ adversarial insdel errors, then we will have two increasing subsequences $I, J \in [n]^\ell$ which agree on at most $k-1$ coordinates, such that the matrix $V_{k,\ell,I,J}|_{X_1=\alpha_1,\ldots,X_n=\alpha_n}$ does not have full column rank. On the other hand, applying Corollary 20 with $r = \lceil \frac{\varepsilon n}{2} \rceil$, for fixed I, J, we have that

$$\Pr\left[\begin{matrix}\text{The matrix } V_{k,\ell,I,J}|_{X_1=\alpha_1,\ldots,X_n=\alpha_n} \\ \text{does not have full column rank}\end{matrix}\right] \leq \left(\frac{2n(k-1)}{q-n+1}\right)^{\frac{\varepsilon n}{2}}.$$

By the union bound over all (I, J) and the fact that $q \geq \left(1 + 2 \cdot 2^{6/\varepsilon} k\right) n$, the probability that $\mathsf{RS}_{n,k}(\alpha_1, \ldots, \alpha_n) \subseteq \mathbb{F}_q^n$ cannot recover from $(1-\varepsilon)n - 2k + 1$ adversarial insdel errors is at most $\binom{n}{\ell}^2 \cdot \left(\frac{2n(k-1)}{q-n+1}\right)^{\frac{\varepsilon n}{2}} \leq 2^{2n} \cdot \left(\frac{2n(k-1)}{q-n+1}\right)^{\frac{\varepsilon n}{2}} \leq 2^{-n}$, as desired. □

4 Achieving Linear Alphabet Size

In this section, we present an improved analysis that achieves a linear alphabet size. We begin by defining the notion of "chains" and establishing related structural results about common subsequences in Section 4.1. Building on these foundations, we then prove Theorem 4 in Section 4.2.

4.1 Chain decomposition

For convenience, we first introduce some definitions.

Definition 22. For an increasing subsequence $I = (I_1, \ldots, I_\ell) \in [n]^\ell$, define

$$\mathrm{Set}(I) := \{I_1, \ldots, I_\ell\}.$$

Definition 23 (Chain)**.** For two increasing subsequences $I, J \in [n]^\ell$, where $n, \ell \in \mathbb{N}^+$, we call the pair (I, J) a *chain* if either (i) $I_i = J_{i+1}$ for all $i = 1, \ldots, \ell-1$, or (ii) $I_{i+1} = J_i$ for all $i = 1, \ldots, \ell-1$. If $\ell = 1$ and $I = J$, we call (I, J) a Type I chain. Otherwise, we call (I, J) a Type II chain. See Figure 1 for some examples.

Recall Definition 14 that for an increasing subsequence $I \in [n]^\ell$ and $P = \{i_1, \ldots, i_{\ell'}\} \subseteq [\ell]$ with $i_1 < \cdots < i_{\ell'}$, we let $I^P = (I_{i_1}, \ldots, I_{i_{\ell'}})$.

Definition 24 (Maximal chain)**.** Let $I, J \in [n]^\ell$ be two increasing subsequences. Let P be a subset of $[\ell]$ of size $\ell' \geq 1$. Let $I' = I^P$ and $J' = J^P$. Suppose (I', J') is a chain. We say the chain (I', J') is *maximal* with respect to (I, J) if

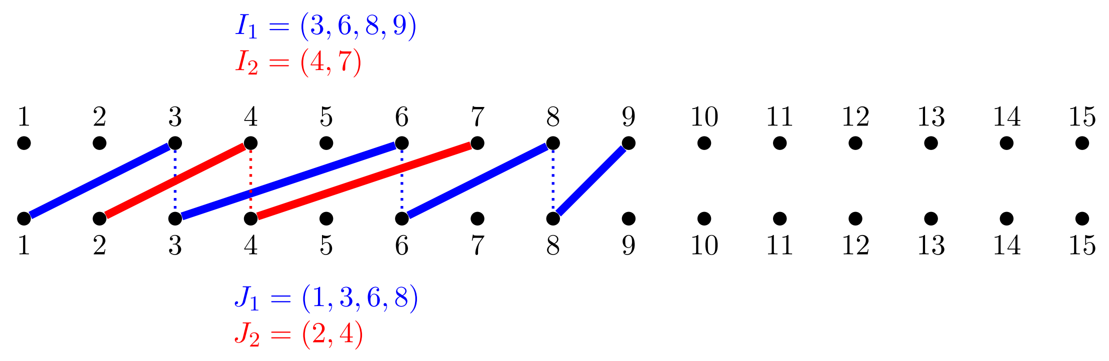


Figure 2: A decomposition of (I, J), where $I = (3, 4, 6, 7, 8, 9)$ and $J = (1, 2, 3, 4, 6, 8)$, into maximal chains (I_1, J_1) and (I_2, J_2). Note that the maximal chains are (I, J)-disjoint but can interleave.

1. (I', J') is a Type I chain, or
2. (I', J') is a Type II chain, and both $\min(I'_1, J'_1)$ and $\max(I'_{\ell'}, J'_{\ell'})$ is in exactly one of $\text{Set}(I)$ and $\text{Set}(J)$.

Intuitively, a chain (I', J') is maximal with respect to (I, J) if it cannot be extended to a longer chain in (I, J) by adding matches from either side.

Definition 25 (Indices of variables)**.** For increasing subsequences $I, J \in [n]^\ell$ and $P \subseteq [\ell]$, define

$$\text{Var}(I, J, P) := \text{Set}(I^P) \cup \text{Set}(J^P) \subseteq [n].$$

Note that $\text{Var}(I, J, P)$ is the set of indices of the variables involved in a V-matrix $V_{k,|P|,I^P,J^P}$.[7]

Definition 26 ((I, J)-disjointness)**.** Let $I, J \in [n]^\ell$ be increasing subsequences. We say two sets $P, P' \subseteq [\ell]$ are (I, J)-*disjoint* if $\text{Var}(I, J, P)$ and $\text{Var}(I, J, P')$ are disjoint. Note that (I, J)-disjointness implies disjointness.

Lemma 27. *Let $I, J \in [n]^\ell$ be two increasing subsequences. Let $P' \subseteq P \subseteq [\ell]$. Suppose $(I^{P'}, J^{P'})$ is a maximal chain with respect to (I^P, J^P). Then P' and $P \setminus P'$ are (I, J)-disjoint.*

Proof. By definition, we have

$$\text{Set}(I^{P\setminus P'}) = \text{Set}(I^P) \setminus \text{Set}(I^{P'}) \quad \text{and} \quad \text{Set}(J^{P\setminus P'}) = \text{Set}(J^P) \setminus \text{Set}(J^{P'}). \tag{5}$$

If $(I^{P'}, J^{P'})$ is a Type I chain, the claim holds by (5) and the fact the that $I^{P'} = J^{P'}$.

Now suppose $(I^{P'}, J^{P'})$ is a Type II chain. Let $\ell' = |P'|$. By definition, either $(I^{P'})_i = (J^{P'})_{i+1}$ for $i = 1, \ldots, \ell' - 1$, or $(I^{P'})_{i+1} = (J^{P'})_i$ for $i = 1, \ldots, \ell' - 1$. This implies that all elements in $\text{Var}(I, J, P')$ except $a := \min((I^{P'})_1, (J^{P'})_1)$ and $b := \max((I^{P'})_{\ell'}, (J^{P'})_{\ell'})$ are in both $\text{Set}(I^{P'})$ and $\text{Set}(J^{P'})$. So by (5), these elements are excluded from $\text{Var}(I, J, P \setminus P')$. As $(I^{P'}, J^{P'})$ is maximal with respect to (I^P, J^P), both a and b are in exactly one of $\text{Set}(I^P)$ and $\text{Set}(J^P)$. And they are in $\text{Set}(I^P)$ (resp. $\text{Set}(J^P)$) iff they are in $\text{Set}(I^{P'})$ (resp. $\text{Set}(J^{P'})$). By (5), a and b are excluded from $\text{Var}(I, J, P \setminus P')$ as well. $\square$

[7]If $k = 1$, these variables do not really appear as $X_i^{k-1} = X_i^0 = 1$. But we still consider $\text{Var}(I, J, P)$ as the set of indices of variables involved in $V_{k,|P|,I^P,J^P}$ in this case.

Theorem 28. *Let $I, J \in [n]^\ell$ be increasing subsequences. Let $P \subseteq [\ell]$. Then there exists a partition $P_1 \sqcup P_2 \sqcup \cdots \sqcup P_s$ of P into nonempty sets P_i such that (I^{P_i}, J^{P_i}) is a maximal chain with respect to (I^P, J^P) for all $i \in [s]$.*

Proof. Induct on $t := |P|$. When $t = 0$, i.e., $P = \emptyset$, the theorem holds trivially. Now assume $t > 0$ and the theorem holds for $t' < t$.

We first construct a nonempty set $P_1 \subseteq P$ such that (I^{P_1}, J^{P_1}) is a maximal chain with respect to (I^P, J^P). Initially, let i_1 be the smallest integer in P, and let $P_1 = \{i_1\}$. If $I_{i_1} = J_{i_1}$, then (I^{P_1}, J^{P_1}) is a Type I chain and hence a maximal chain with respect to (I^P, J^P) by Definition 24.

Now assume $I_{i_1} < J_{i_1}$. In this case, run the following process on $P_1 = \{i_1\}$:

```
1 $s \leftarrow 1$.
2 while there exists $i \in P$ such that $I_i = J_{i_s}$ do        // $i$ is unique if it exists
3     $s \leftarrow s + 1$.
4     $i_s \leftarrow i$.
5     $P_1 \leftarrow P_1 \cup \{i_s\}$.
6 end
```

The above process iteratively finds indices $i_1 < \cdots < i_s$ in P such that $I_{i_1} < J_{i_1} = I_{i_2} < J_{i_2} = \cdots = I_{i_s} < J_{i_s}$, and $J_{i_s} \notin \mathrm{Set}(I^P)$. The resulting set P_1 is $\{i_1, \ldots, i_s\}$. Note that both $I_{i_1} = \min(I_{i_1}, J_{i_1})$ and $J_{i_s} = \max(I_{i_s}, J_{i_s})$ are in exactly one of $\mathrm{Set}(I^P)$ and $\mathrm{Set}(J^P)$. So by Definition 24, (I^{P_1}, J^{P_1}) is a maximal chain with respect to (I^P, J^P).

Finally, in the case where $I_{i_1} > J_{i_1}$, we construct P_1 via a symmetric process, where I and J and swapped.

In all cases, we obtain a nonempty set $P_1 \subseteq P$ such that (I^{P_1}, J^{P_1}) is a maximal chain with respect to (I^P, J^P). By Lemma 27, P_1 and $P \setminus P_1$ are (I, J)-disjoint.

By the induction hypothesis, there exists a partition $P_2 \sqcup \cdots \sqcup P_s$ of $P \setminus P_1$ such that (I^{P_i}, J^{P_i}) is a maximal chain with respect to $(I^{P\setminus P_1}, J^{P\setminus P_1})$ for $i = 2, \ldots, s$. As P_1 and $P \setminus P_1$ are (I, J)-disjoint, it follows from Definition 24 that a maximal chain with respect to $(I^{P\setminus P_1}, J^{P\setminus P_1})$ is also maximal with respect to (I^P, J^P). So (I^{P_i}, J^{P_i}) is a maximal chain with respect to (I^P, J^P) for $i = 1, 2, \ldots, s$, as desired. □

By choosing $P = [\ell]$, we obtain the following corollary, which states that for increasing subsequences $I, J \in [n]^\ell$, (I, J) can always be decomposed into maximal chains.

Corollary 29. *For any two increasing subsequences $I, J \in [n]^\ell$, there exists a partition $P_1 \sqcup P_2 \sqcup \cdots \sqcup P_s$ of $[\ell]$ such that (I^{P_i}, J^{P_i}) is a maximal chain with respect to (I, J) for all $i \in [s]$.*

See Figure 2 for an example. In fact, one can prove that the decomposition into maximal chains is unique, but we do not need this result.

Splitting long chains into short ones. The maximal chains in Corollary 29 can be very long. The next lemma states that, by removing a small fraction of matchings from (I, J), we can split the long chains into very short ones while maintaining their (I, J)-disjointness. (See also Figure 3.)

Lemma 30. *Let $I, J \in [n]^\ell$ be increasing subsequences. Let $\varepsilon \in (0, 1)$. Then there exist a subset $P \subseteq [\ell]$ of size at least $(1 - \varepsilon)\ell$ and a partition $P_1 \sqcup P_2 \sqcup \cdots \sqcup P_s$ of P such that,*

1. *$P_1, \ldots, P_s$ are mutually (I, J)-disjoint.*

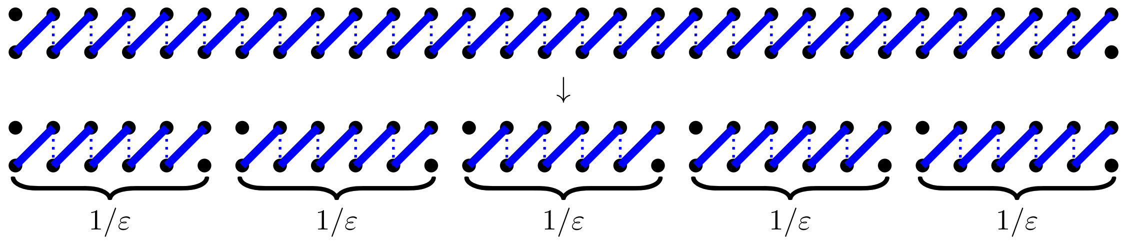


Figure 3: Lemma 30: splitting long chains into short ones. Ensure that all chains have a length of at most $1/\varepsilon$ by removing at most ε fraction of pairs from each chain.

2. $|P_i| \le 1/\varepsilon$ *for* $i \in [s]$*.*

Proof. By Corollary 29, there exists a partition $P_1 \sqcup P_2 \sqcup \cdots \sqcup P_s$ of $[\ell]$ such that (I^{P_i}, J^{P_i}) is a maximal chain with respect to (I, J) for all $i \in [s]$. Fix such a partition. Then Item 1 holds by Lemma 27. Consider any $i \in [s]$ such that $|P_i| > 1/\varepsilon$. Note that (I^{P_i}, J^{P_i}) must be a Type II chain (since otherwise we would have $|P_i| = 1 \le 1/\varepsilon$). Let $t = \lfloor 1/\varepsilon \rfloor + 1 \ge 1/\varepsilon$. We remove the j-th smallest element of P_i for $j = t, 2t, \ldots, \lfloor P_i/t \rfloor t$. The remaining elements split into consecutive blocks $P_{i,j}$, each of size at most $t - 1 \le 1/\varepsilon$. Splitting P_i into $P_{i,j}$ preserves the (I, J)-disjointness of the sets. See Figure 3 for an illustration.

Perform the above operations for each i with $|P_i| > 1/\varepsilon$, and we obtain the desired partition. The set P consists of the remaining elements of $[\ell]$. Its size is at least $(1-\varepsilon)\ell$ since we have removed at most $(1/t)$-fraction of elements from each P_i, where $1/t \le \varepsilon$. □

4.2 Proof of Theorem 4

Fix $I, J \in [n]^\ell$ to be increasing subsequences that agree on at most $k - 1$ coordinates. For convenience, we introduce the following notation.

Definition 31. Let $n, k, \ell \in \mathbb{N}^+$. Let $P_1, \ldots, P_s \subseteq [\ell]$ be nonempty subsets of $[\ell]$. Define the block matrix

$$\widetilde{V}_{k,I,J,(P_i)_{i=1}^s}(X_1, \ldots, X_n) = \begin{pmatrix} V_{k,|P_1|,I^{P_1},J^{P_1}} \\ V_{k,|P_2|,I^{P_2},J^{P_2}} \\ \vdots \\ V_{k,|P_s|,I^{P_s},J^{P_s}} \end{pmatrix}$$

which is a $(\sum_{i=1}^s |P_i|) \times (2k-1)$ matrix over $\mathbb{F}_q(X_1, \ldots, X_n)$.

Lemma 32. *Let* $P_1, \ldots, P_s \subseteq [\ell]$ *and* $\alpha_1, \ldots, \alpha_n \in \mathbb{F}_q$*. If* $\widetilde{V}_{k,I,J,(P_i)_{i=1}^s}$ *has full column rank under the assignment* $X_1 = \alpha_1, \ldots, X_n = \alpha_n$*, then* $V_{k,\ell,I,J}$ *also has full column rank under the same assignment.*

Proof. By definition, each row of $\widetilde{V}_{k,I,J,(P_i)_{i=1}^s}|_{X_1=\alpha_1,\ldots,X_n=\alpha_n}$ is a row of $V_{k,\ell,I,J}|_{X_1=\alpha_1,\ldots,X_n=\alpha_n}$. So if the former matrix has full column rank, i.e., its row space has dimension $2k-1$, then so does the latter matrix. □

Choosing sets $P_1, \ldots, P_s$. Fix a parameter $\varepsilon_0 \in (0, 1)$, whose exact value will be determined later. We note that by Lemma 30, there exist mutually (I, J)-disjoint nonempty sets $P_1, \ldots, P_s \subseteq [\ell]$, each of size at most $1/\varepsilon_0$, such that $\sum_{i=1}^s |P_i| \ge (1 - \varepsilon_0)\ell$ and each (I^{P_i}, J^{P_i}) is a chain. Fix such $P_1, \ldots, P_s$. We further make the following assumption:

Assumption 1. $|P_1| \le |P_2| \le \cdots \le |P_s|$. *Moreover, for $i, j \in [s]$ with $i < j$, if (I^{P_j}, J^{P_j}) is a Type I chain, so is (I^{P_i}, J^{P_i}). In other words, the Type I chains (I^{P_i}, J^{P_i}) have the smallest indices i.*

Note that Assumption 1 can be guaranteed by permuting the sets P_i.

We will show that $\widetilde{V}_{k,I,J,(P_i)_{i=1}^s}|_{X_1=\alpha_1,\ldots,X_n=\alpha_n}$ has full column rank with high probability over random $(\alpha_1, \ldots, \alpha_n)$. By Lemma 32, this would imply that $V_{k,\ell,I,J}|_{X_1=\alpha_1,\ldots,X_n=\alpha_n}$ also has full column rank with high probability.

Algorithm 2: `CertifyFullColumnRankness2`

```
Input: $n, k, \ell, r \in \mathbb{N}^+$, $\varepsilon_0 \in (0,1)$, increasing subsequences $I, J \in [n]^\ell$, $P_1, \ldots, P_s \subseteq [\ell]$, and
       $\alpha_1, \ldots, \alpha_n \in \mathbb{F}_q$.
Output: "SUCCESS" or a sequence $(i_1, \ldots, i_r) \in \{0, \ldots, 2k-2\}^r$.
1  $S \leftarrow \emptyset$.                                  // set of indices of assigned variables
2  $c \leftarrow 0$.                                          // number of assigned rows
3  Pick the smallest $m \in [s]$ such that $\sum_{i=s-m+1}^{s} |P_i| \in [r/\varepsilon_0, (r+1)/\varepsilon_0]$. // last $m$ chains
    form the bank
4  $a \leftarrow 1$.                                          // current chain to assign
5  $b \leftarrow s - m + 1$.                                  // current chain in the bank
6  $j \leftarrow 0$.                                          // number of failed attempts
7  for $i = 1$ to $s$ do $Q_i \leftarrow P_i$.
8  while $c < 2k - 1$ do
9      $M \leftarrow$ the top $(2k-1) \times (2k-1)$ submatrix of $\widetilde{V}_{k,I,J,(Q_i)_{i=1}^{s-m}}$.
10     $S' \leftarrow S \cup \mathrm{Var}(I, J, Q_a)$.
11     $\overline{M} \leftarrow M|_{X_i=\alpha_i \text{ for } i \in S'}$.
12     if $\overline{M}$ is nonsingular then               // assign variables in the $a$-th chain
13         $S \leftarrow S'$.
14         $c \leftarrow c + |Q_a|$.
15         $a \leftarrow a + 1$.
16     else     // replace the $a$-th chain by (part of) the $b$-th chain in the bank
17         $Q_a \leftarrow$ set of the smallest $|Q_a|$ elements in $Q_b$.
18         $Q_b \leftarrow \emptyset$.
19         $b \leftarrow b + 1$.
20         $j \leftarrow j + 1$.
21         $i_j \leftarrow c$.
22         if $j = r$ then output $(i_1, \ldots, i_r)$ and halt.
23     end
24 end
25 Output "SUCCESS".
```

To bound the probability that $\widetilde{V}_{k,I,J,(P_i)_{i=1}^s}|_{X_1=\alpha_1,\ldots,X_n=\alpha_n}$ does not have full column rank, we use Algorithm 2, whose pseudocode is given above, to assign the variables chain by chain. We sketch how the algorithm works.

- First, choose the smallest $m \in [s]$ such that $\sum_{i=s-m+1}^{s} |P_i| \in [r/\varepsilon_0, (r+1)/\varepsilon_0]$ (Line 3). We will soon prove that such m exists for ℓ slightly larger than $2k-1$. The last m chains (I^{P_i}, J^{P_i}), $i = s-m+1, \ldots, s$, form a "bank." At a high level, Algorithm 2 attempts to assign the variables in the first $s - m$ chains, i.e., those not in the bank, while keeping the top $(2k-1) \times (2k-1)$ submatrix of $\widetilde{V}_{k,I,J,(P_i)_{i=1}^{s-m}}$ nonsingular even after the assignment. If assigning the variables in some chain violates this property, the algorithm will use part of a chain in the bank to replace that chain, and continue. Such a replacement is done by updating the sets P_i. To avoid confusion, we create a copy $Q_i = P_i$ for $i \in [s]$ (Line 7) and update the sets Q_i instead.

- The variables are assigned chain by chain. That is, for $a = 1, 2, \ldots,$ Algorithm 2 assigns α_i to X_i for all $i \in \mathrm{Var}(I, J, Q_a)$. The algorithm terminates when the top $2k-1$ rows of $\widetilde{V}_{k,I,J,(Q_i)_{i=1}^{s-m}}$ have been fully assigned. This condition is checked at Line 8. The number of assigned rows is maintained as a variable c. That is, if the first i chains $(I^{Q_1}, J^{Q_1}), \ldots, (I^{Q_i}, J^{Q_i})$ have been assigned, then $c = \sum_{j=1}^{i} |Q_j|$.
- When attempting to assign the variables in the a-th chain (I^{Q_a}, J^{Q_a}), we check (Line 12) whether the top $(2k-1) \times (2k-1)$ submatrix of $\widetilde{V}_{k,I,J,(Q_i)_{i=1}^{s-m}}$ remains nonsingular after the assignment. If so, we perform the assignment, and then move to the next chain (Lines 13–15). If not, we use a prefix of a chain (I^{Q_b}, J^{Q_b}) in the bank to replace the chain (I^{Q_a}, J^{Q_a}), and then throw away (I^{Q_b}, J^{Q_b}).[8] In the algorithm, the updates to the chains (I^{Q_a}, J^{Q_a}) and (I^{Q_b}, J^{Q_b}) are done by simply updating the sets Q_a and Q_b (Lines 17–18).
- Finally, the algorithm maintains a variable j, which is the number of times the nonsingularity condition at Line 12 fails to hold. It also maintains a sequence $(i_1, \ldots, i_j)$. When the nonsingularity condition fails for the j-th time, the number of successfully assigned rows is stored into i_j (Line 21). We have $i_j \le 2k-2$ due to the condition checked at Line 8. If j reaches a given integer r, the algorithm outputs the sequence $(i_1, \ldots, i_r)$ and halts (Line 22). We will eventually choose $r = \Theta(\varepsilon^2 n)$.

From now on, assume ℓ is chosen such that

$$(1-\varepsilon_0)\ell \ge 2k - 1 + (r+1)/\varepsilon_0. \tag{6}$$

As promised, we now show that Line 3 of Algorithm 2 is valid by proving the following lemma.

Lemma 33. *There exists $m \in [s]$ such that $\sum_{i=s-m+1}^{s} |P_i| \in [r/\varepsilon_0, (r+1)/\varepsilon_0]$. Moreover, for such m, it holds that $\sum_{i=1}^{s-m} |P_i| \ge 2k-1$ and $m \ge r$.*

Proof. Recall that $P_1, \ldots, P_s$ that we choose satisfy $\sum_{i=1}^{s} |P_i| \ge (1-\varepsilon_0)\ell$. Combining this with (6), we have that $\sum_{i=1}^{s} |P_i| \ge (1-\varepsilon_0)\ell \ge r/\varepsilon_0$. Thus, there exists $m \in [s]$ such that $\sum_{i=s-m+1}^{s} |P_i| \ge r/\varepsilon_0$. Pick the smallest such m. Then $\sum_{i=s-m+1}^{s} |P_i| \le (r+1)/\varepsilon_0$ by the minimality of m and the fact that the size of each P_i is at most $1/\varepsilon_0$. Therefore,

$$\sum_{i=1}^{s-m} |P_i| = \left(\sum_{i=1}^{s} |P_i|\right) - \left(\sum_{i=s-m+1}^{s} |P_i|\right) \ge (1-\varepsilon_0)\ell - (r+1)/\varepsilon_0 \overset{(6)}{\ge} 2k-1.$$

Finally, as $\sum_{i=s-m+1}^{s} |P_i| \ge r/\varepsilon_0$ and each P_i has size at most $1/\varepsilon_0$, we have $m \ge r$. ☐

The next lemma ensures the validity of the other parts of Algorithm 2. Particularly, Item (b) guarantees that Line 9 is valid, while Items (d) and (e) guarantees that Lines 17–18 are valid.

Lemma 34. *During Algorithm 2, after Line 7:*

[8]Alternatively, one can remove the used portion of (I^{Q_b}, J^{Q_b}) and continue using the remaining part. However, this approach does not qualitatively improve our parameters.

(a) $|Q_i| = |P_i|$ *for* $i \in [s-m]$*. In particular,* $|Q_i| \le 1/\varepsilon_0$ *for* $i \in [s-m]$*.*

(b) $\sum_{i=1}^{s-m} |Q_i| \ge 2k-1$*.*

(c) $1 \le a \le s-m$ *always holds at Line 9.*

(d) $s-m+1 \le b \le s$ *always holds at Line 9.*

(e) $|Q_b| \ge |Q_a|$ *always holds at Line 9, and consequently, also at Line 17.*

Proof. First, note that the algorithm runs Lines 17–22 at most r times due to Line 22 and the fact that each time it runs, b increases by one at Line 19. Initially, $b = s-m+1$. As $r \le m$ by Lemma 33, Item (d) holds.

Note that $\sum_{j=1}^{s-m} |P_i| \ge 2k-1$ holds by Lemma 33. Thus, if (a) is true, so is (b). And if (b) is true, (c) must also be true. This is because if $a > s-m$ ever holds at Line 9, then by (b), we have $c = \sum_{i=1}^{a-1} |Q_i| \ge 2k-1$ and the algorithm would have exited the while loop before at Line 8.

We know (a) holds initially after Line 7. It continues to hold in each iteration of the while loop since Line 17 does not change the size of Q_a and Line 18 only changes Q_b for $b > s-m$.

Finally, to see that (e) holds, note that at Line 9, we have $Q_b = P_b$. This is because Q_b is only changed at Line 18, followed by Line 19 which increases b by one. So $Q_b = P_b$ still holds after Line 19. Then by (a), (c), (d), and the fact that $|P_1| \le |P_2| \le \cdots \le |P_s|$, we have $|Q_b| = |P_b| \ge |P_a| = |Q_a|$ at Line 9, proving (e). □

The output of Algorithm 2 can be used to certify the full-column-rankness of the matrix $V_{k,\ell,I,J}|_{X_1=\alpha_1,\ldots,X_n=\alpha_n}$, as stated by the following lemma.

Lemma 35. *Algorithm 2 terminates, outputting either "SUCCESS" or* $(i_1,\ldots,i_r) \in \{0,\ldots,2k-2\}^r$*. In the former case,* $V_{k,\ell,I,J}|_{X_1=\alpha_1,\ldots,X_n=\alpha_n}$ *has full column rank.*

Proof. In each iteration of the while loop, either a or j increases by one. If j reaches r, the algorithm outputs $(i_1,\ldots,i_r)$ and halts, where $i_1,\ldots,i_r \le 2k-2$ due to the condition checked at Line 8. On the other hand, if a gets large enough, the number of assigned rows $c = \sum_{i=1}^{a-1} |Q_i|$ becomes at least $2k-1$ by Lemma 34, in which case the algorithm outputs "SUCCESS" and terminates. Suppose this is the case. Then the condition at Line 12 is satisfied in the last round of the while loop, i.e., the top $(2k-1)\times(2k-1)$ submatrix of $\widetilde{V}_{k,I,J,(Q_i)_{i=1}^{s-m}}|_{X_i=\alpha_i \text{ for } i\in S'}$, denoted by $\overline{M}$, is nonsingular. As the top $2k-1$ rows have been fully assigned, $\overline{M}$ is also the top $(2k-1)\times(2k-1)$ submatrix of $\widetilde{V}_{k,I,J,(Q_i)_{i=1}^{s-m}}|_{X_1=\alpha_1,\ldots,X_n=\alpha_n}$. By Lemma 32, $V_{k,\ell,I,J}|_{X_1=\alpha_1,\ldots,X_n=\alpha_n}$ has full column rank. □

As the variable c in Algorithm 2 never decreases, we have the following lemma.

Lemma 36. *Any sequence* $(i_1,\ldots,i_r)$ *output by Algorithm 2 satisfies* $i_1 \le \cdots \le i_r$*.*

Recall that for $i \in [s]$, the set of indices of the variables involved in $V_{k,|Q_i|,I^{Q_i},J^{Q_i}}$ is $\mathrm{Var}(I,J,Q_i)$. The next lemma states that these sets are mutually (I,J)-disjoint (see Definition 26).

Lemma 37. $Q_1,\ldots,Q_s$ *are mutually* (I,J)*-disjoint before and after Lines 17–18.*

Proof. Initially, the (I,J)-disjointness holds since it holds for the sets $P_1,\ldots,P_s$ and $Q_i = P_i$ for $i \in [s]$. We just need to verify that Lines 17–18 preserve the (I,J)-disjointness. At Line 17, Q_a is replaced by a subset of Q_b, and at Line 18, we let $Q_b = \emptyset$. So the (I,J)-disjointness still holds. □

The next lemma states that the values α_i with $i \in \mathrm{Var}(I,J,Q_a)$ used in one iteration of the while loop have not been used in the previous iterations.

Lemma 38. *At Line 9 of Algorithm 2, none of the elements in* $\mathrm{Var}(I, J, Q_a)$ *has been added to* S' *in the previous iterations of the while loop.*

Proof. In each iteration of the while loop, either a increases by one, or we replace Q_a by a subset of Q_b and let $Q_b = \emptyset$. Either case, the set $\bigcup_{i=a}^{s} \mathrm{Var}(I, J, Q_i)$ shrinks to a proper subset and excludes the elements in $\mathrm{Var}(I, J, Q_a)$ (for the old a).

Suppose an element $x \in [n]$ is added to S' for the first time at Line 10. Then $x \in \mathrm{Var}(I, J, Q_a)$. (This is because, if $x \in S$ at Line 10, then x must have been added to S' before.) By the first paragraph, x will not appear in $\mathrm{Var}(I, J, Q_a)$ in the later iterations. The contrapositive of this statement is the lemma. □

Next, we make the crucial observation that the behavior of the algorithm is mostly determined by the values of j, $(i_1, \ldots, i_j)$, and c, regardless of $\alpha_1, \ldots, \alpha_n$.

Lemma 39. *At Line 12 of Algorithm 2, the values of* $Q_1, \ldots, Q_s$ *and* a *are determined by the values of* j*,* $i_1, \ldots, i_j$*, and* c*, together with the input excluding* $\alpha_1, \ldots, \alpha_n$*.*

Proof. Fix the input excluding $\alpha_1, \ldots, \alpha_n$ and the values of j, $i_1, \ldots, i_j$, and c. We use j^*, $i_1^*, \ldots, i_{j^*}^*$, and c^* to denote these values and distinguish them from the variables.

Simulate the algorithm until it reaches Line 12 with $j = j^*$, $(i_1, \ldots, i_j) = (i_1^*, \ldots, i_j^*)$, and $c = c^*$. Then read off $Q_1, \ldots, Q_s$ and a. Doing the simulation naively requires knowing $\alpha_1, \ldots, \alpha_n$ to check the condition at Line 12. However, we will show that the outcome of any condition check at Line 12 performed during the simulation can be determined by our fixed values.

Suppose the simulation has reached Line 12, where $i_1, \ldots, i_j$ have been chosen and $j \le j^*$. First consider the case $j < j^*$. In this case, as j has not yet reached j^*, the values $i_{j+1}^*, \ldots, i_{j^*}^*$ have not yet been assigned to $i_{j+1}, \ldots, i_{j^*}$ by the algorithm. Then there are two subcases:

1. $c \ne i_{j+1}^*$. In this case, the condition at Line 12 must be true, since otherwise the algorithm would store c, rather than i_{j+1}^*, into i_{j+1}.

2. $c = i_{j+1}^*$. In this case, the condition at Line 12 must be false, since otherwise the algorithm would increase c at Line 14 and then either choose i_{j+1} to be greater than i_{j+1}^* or not choose it at all.

Now consider the case $j = j^*$. In this situation, j has reached j^*. However, recall that the simulation ends when $c = c^*$. Again, we have two subcases to consider.

1. $c \ne c^*$. In this case, the condition at Line 12 must be true, so that the algorithm can continue to increase c to c^* before choosing i_{j^*+1}.

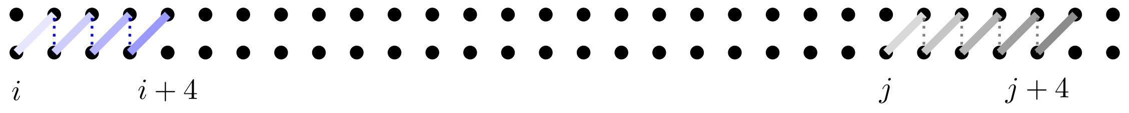
2. $c = c^*$. This is the end of the simulation where we read off $Q_1, \ldots, Q_s$ and a. Note that the next time the algorithm reaches Line 12, either j will be greater than j^* or c will be greater than c^*.

As we can determine $Q_1, \ldots, Q_s$ and a without knowing $\alpha_1, \ldots, \alpha_n$, the claim follows. □

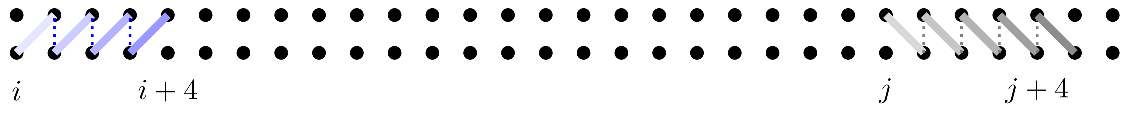
We also need the following crucial lemma.

Lemma 40. *At Line 12 of Algorithm 2,* $M|_{X_i = \alpha_i \text{ for } i \in S}$ *is nonsingular.*

$$\begin{bmatrix} \vdots & & & \vdots & \vdots & & \vdots \\ 1 & X_{i+1} & \cdots & X_{i+1}^{k-1} & X_i & \cdots & X_i^{k-1} \\ 1 & X_{i+2} & \cdots & X_{i+2}^{k-1} & X_{i+1} & \cdots & X_{i+1}^{k-1} \\ 1 & X_{i+3} & \cdots & X_{i+3}^{k-1} & X_{i+2} & \cdots & X_{i+2}^{k-1} \\ 1 & X_{i+4} & \cdots & X_{i+4}^{k-1} & X_{i+3} & \cdots & X_{i+3}^{k-1} \\ \vdots & & & \vdots & \vdots & & \vdots \end{bmatrix} \rightarrow \begin{bmatrix} \vdots & & & \vdots & \vdots & & \vdots \\ 1 & X_{j+1} & \cdots & X_{j+1}^{k-1} & X_j & \cdots & X_j^{k-1} \\ 1 & X_{j+2} & \cdots & X_{j+2}^{k-1} & X_{j+1} & \cdots & X_{j+1}^{k-1} \\ 1 & X_{j+3} & \cdots & X_{j+3}^{k-1} & X_{j+2} & \cdots & X_{j+2}^{k-1} \\ 1 & X_{j+4} & \cdots & X_{j+4}^{k-1} & X_{j+3} & \cdots & X_{j+3}^{k-1} \\ \vdots & & & \vdots & \vdots & & \vdots \end{bmatrix}$$



$$\begin{bmatrix} \vdots & & & \vdots & \vdots & & \vdots \\ 1 & X_{i+1} & \cdots & X_{i+1}^{k-1} & X_i & \cdots & X_i^{k-1} \\ 1 & X_{i+2} & \cdots & X_{i+2}^{k-1} & X_{i+1} & \cdots & X_{i+1}^{k-1} \\ 1 & X_{i+3} & \cdots & X_{i+3}^{k-1} & X_{i+2} & \cdots & X_{i+2}^{k-1} \\ 1 & X_{i+4} & \cdots & X_{i+4}^{k-1} & X_{i+3} & \cdots & X_{i+3}^{k-1} \\ \vdots & & & \vdots & \vdots & & \vdots \end{bmatrix} \rightarrow \begin{bmatrix} \vdots & & & \vdots & \vdots & & \vdots \\ 1 & X_{j+3} & \cdots & X_{j+3}^{k-1} & X_{j+4} & \cdots & X_{j+4}^{k-1} \\ 1 & X_{j+2} & \cdots & X_{j+2}^{k-1} & X_{j+3} & \cdots & X_{j+3}^{k-1} \\ 1 & X_{j+1} & \cdots & X_{j+1}^{k-1} & X_{j+2} & \cdots & X_{j+2}^{k-1} \\ 1 & X_{j} & \cdots & X_{j}^{k-1} & X_{j+1} & \cdots & X_{j+1}^{k-1} \\ \vdots & & & \vdots & \vdots & & \vdots \end{bmatrix}$$



$$\begin{bmatrix} \vdots & & & \vdots & \vdots & & \vdots \\ \vdots & & & \vdots & \vdots & & \vdots \\ 1 & X_i & \cdots & X_i^{k-1} & X_i & \cdots & X_i^{k-1} \\ \vdots & & & \vdots & \vdots & & \vdots \\ \vdots & & & \vdots & \vdots & & \vdots \end{bmatrix} \rightarrow \begin{bmatrix} \vdots & & & \vdots & \vdots & & \vdots \\ \vdots & & & \vdots & \vdots & & \vdots \\ 1 & X_j & \cdots & X_j^{k-1} & X_{j+1} & \cdots & X_{j+1}^{k-1} \\ \vdots & & & \vdots & \vdots & & \vdots \\ \vdots & & & \vdots & \vdots & & \vdots \end{bmatrix}$$

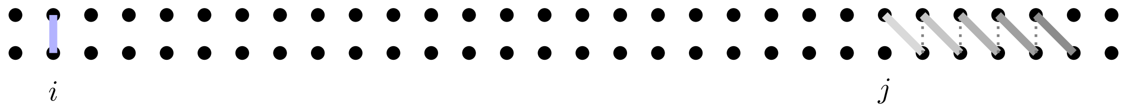


Figure 4: Lemma 40. Re-indeterminating a faulty (blue) chain with (part of) a (gray) chain from the bank. Three cases: (1) The faulty chain and the bank's chain are both type II with the same orientation. The new V-matrix is equivalent to the original one. (2) The faulty chain and the bank's (gray) chain are both type II with different orientations. The new V-matrix is equivalent to the original one when we view the bank's gray-chain in *reverse*. (3) The faulty chain is type I and the bank's chain is type II. The new V-matrix is not exactly equivalent to the original one, but is similar enough: since the old matrix is full rank (before setting X_i), the new matrix certainly is full rank as well, which is what we need.

Proof. We prove by induction on the iteration of the while loop. When the algorithm first reaches Line 12, we have $S = \emptyset$, and hence $M|_{X_i=\alpha_i \text{ for } i\in S}$ is simply M, which is the top $(2k-1)\times(2k-1)$ submatrix of $\widetilde{V}_{k,I,J,(Q_i)_{i=1}^{s-m}} = \widetilde{V}_{k,I,J,(P_i)_{i=1}^{s-m}}$. Note that by Definition 31, this matrix equals, up to reordering of rows, the V-matrix $V_{k,2k-1,I^P,J^P}$ for some $P \subseteq \bigcup_{i=1}^{s-m} P_i$ of size $2k-1$. So by Corollary 15, $M|_{X_i=\alpha_i \text{ for } i\in S}$ is nonsingular.

Next, assume that for some $t \in \mathbb{N}^+$, $M|_{X_i=\alpha_i \text{ for } i\in S}$ is nonsingular at Line 12 in the t-th iteration of the while loop. It suffices to show that $M|_{X_i=\alpha_i \text{ for } i\in S}$ is still nonsingular at Line 12 in the $(t+1)$-th iteration. There are two cases:

Case 1: the matrix $\overline{M} = M|_{X_i=\alpha_i \text{ for } i\in S'}$ is nonsingular at Line 12 in the t-th iteration. The algorithm then runs Lines 13–15 and replaces S by S'. So in the $(t+1)$-th iteration, $M|_{X_i=\alpha_i \text{ for } i\in S}$ is nonsingular at Line 12.

Case 2: the matrix $\overline{M} = M|_{X_i=\alpha_i \text{ for } i\in S'}$ is singular at Line 12 in the t-th iteration. In this case, the algorithm runs Lines 17–22, which do not change the set S. However, Q_a is replaced by a subset of Q_b at Line 17, which we denote by Q'_b. This replacement, together with Line 9 in the $(t+1)$-th iteration, changes the value of M. We need to verify that it does not affect the nonsingularity of $M|_{X_i=\alpha_i \text{ for } i\in S}$.

At Line 12 in the t-th iteration, we have $\mathrm{Var}(I,J,Q_a)\cap S = \emptyset$. This is because if some $x \in \mathrm{Var}(I,J,Q_a)$ is in S, then it must have been added to S' before the t-th iteration, contradicting Lemma 38. Similarly, we have $\mathrm{Var}(I,J,Q'_b)\cap S = \emptyset$ by the same argument and the fact that Q'_b becomes the new Q_a at Line 9 in the $(t+1)$-th iteration. The conclusion is that even under the assignment $X_i = \alpha_i$ for $i \in S$, the variables $X_{i'}$ with $i' \in \mathrm{Var}(I,J,Q_a)\cup\mathrm{Var}(I,J,Q'_b)$ remain free variables.

Let $e = |Q_a|$. Suppose $Q_a = \{j_1,\dots,j_e\} \subseteq [\ell]$ with $j_1 < \cdots < j_e$, which is replaced by $Q'_b = \{j'_1,\dots,j'_e\} \subseteq Q_b$ with $j'_1 < \cdots < j'_e$ at Line 17 in the t-th iteration. As M only contains the first $2k-1$ rows of $\widetilde{V}_{k,I,J,(Q_i)_{i=1}^{s-m}}$, it only contains the first e' rows of $V_{k,e,I^{Q_a},J^{Q_a}}$ for some $e' \in \{0,\dots,e\}$. If $e'=0$, then replacing Q_a by Q'_b does not change M and we are done. So assume $e' > 0$.

Define M' as the matrix $M|_{X_i=\alpha_i \text{ for } i\in S}$ at Line 12 in the t-th iteration, and define M'' as $M|_{X_i=\alpha_i \text{ for } i\in S}$ at Line 12 in the $(t+1)$-th iteration. By assumption, M' is nonsingular, and we want to show that M'' is also nonsingular. Define the increasing subsequences $I', J', I'', J'' \in [n]^{e'}$ by $I'_i = I_{j_i}$, $J'_i = J_{j_i}$, $I''_i = I_{j'_i}$, and $J''_i = J_{j'_i}$ for $i \in [e']$, i.e., $I' = I^{Q_a}$, $J' = J^{Q_a}$, $I'' = I^{Q'_b}$, and $J'' = J^{Q'_b}$. Then replacing Q_a by Q'_b has the effect that the submatrix $V_{k,e',I',J'}$ of M' is replaced by the submatrix $V_{k,e',I'',J''}$ of M'', turning M' into M''.

Note that (I',J') is a chain of the same type as (I^{Q_a}, I^{Q_a}), and (I'',J'') is a chain of the same type as (I^{Q_b}, J^{Q_b}). There are several cases (see Figure 4):

1. (I',J') and (I'',J'') are both Type-II chains, and $I'_1 < J'_1$ iff $I''_1 < J''_1$. In this case, if we substitute $X_{I'_i}$ for $X_{I''_i}$ and $X_{J'_i}$ for $X_{J''_i}$ in M'' for $i \in [e']$, then we recover M'. That is, $M' = M''|_{X_{I''_i}=X_{J'_i}, X_{J''_i}=X_{J'_i} \text{ for } i\in[e']}$. Note that either $I''_i = J''_{i+1}$ for $i = 1,\dots,e'-1$ or $I''_{i+1} = J''_i$ for $i = 1,\dots,e'-1$, but the same relations hold for the coordinates of I' and J' as well since (I',J') and (I'',J'') have the same type and the same orientation. So the substitutions of variables are valid.

2. (I',J') and (I'',J'') are both Type-II chains, and $I'_1 < J'_1$ iff $I''_1 > J''_1$. In this case, if we substitute $X_{I'_{e-i+1}}$ for $X_{I''_i}$ and $X_{J'_{e-i+1}}$ for $X_{J''_i}$ in M'' for $i \in [e']$, then we recover M' up to reordering of rows. That is, M' is, up to reordering of rows, equal to the matrix $M''|_{X_{I''_i}=X_{J'_{e'-i+1}}, X_{J''_i}=X_{J'_{e'-i+1}} \text{ for } i\in[e']}$. Note that if $I''_i = J''_{i+1}$, then $I'_{e-i+1} = J'_{e-i}$. Similarly,

if $I''_{i+1} = J''_i$, then $I'_{e-i} = J'_{e-i+1}$. That is, the relations satisfied by the coordinates of I'' and J'' are also satisfied by the coordinates of the *reversals* of I' and J'. So the substitutions of variables are valid.

3. (I', J') is a Type-I chain. In this case, $e' = 1$ and $I'_1 = J'_1$. Then $M' = M''|_{X_{I''_1}=X_{I'_1}, X_{J''_1}=X_{J'_1}}$. Note that (I'', J'') may or may not be a Type-I chain, but the substitutions of variables are valid regardless.

Finally, if (I'', J'') is a Type-I chain, then by Assumption 1, the chain (I', J'), which comes from $(I^{P_{b'}}, J^{P_{b'}})$ for some $b' < b$, is also a Type-I chain. So the above three cases cover all the possibilities. In each case, M' is, up to reordering rows, equal to $\sigma(M'')$, where σ is a ring endomorphism of $\mathbb{F}_q[X_i : i \in [n] \setminus S]$ induced by the substitutions of variables described above and applied to M'' entrywise. So $\det(M') = \pm\sigma(\det(M''))$. As $\det(M') \neq 0$, we have $\det(M'') \neq 0$, i.e., M'' is nonsingular. □

Now we are ready to bound the probability that Algorithm 2 outputs a particular sequence.

Lemma 41. *Suppose $q \geq n$ and $(\alpha_1, \dots, \alpha_n)$ is chosen uniformly at random from the set of all n-tuples of distinct elements in $\mathbb{F}_q$. The probability that Algorithm 2 outputs a fixed sequence $(i_1^*, \dots, i_r^*) \in \{0, \dots, 2k-2\}^r$ is at most p^r, where*

$$p = \frac{((1/\varepsilon_0) + 1) \cdot 2(k-1)}{q - n + 1}.$$

Proof. For $0 \leq t \leq r$, let F_t be the event that the algorithm chooses $(i_1, \dots, i_t) = (i_1^*, \dots, i_t^*)$. We will prove by induction that $\Pr[F_t] \leq p^t$. The lemma follows by choosing $t = r$.

For $t = 0$, the claim hold trivially. Assume the claim holds for some $t < r$. We now prove that it holds for $t + 1$ as well.

Let F'_t be the sub-event of F_t that the algorithm reaches Line 12 with $j = t$, $(i_1, \dots, i_t) = (i_1^*, \dots, i_t^*)$, and $c = i_{t+1}^*$. Note that F'_t must happen if F_{t+1} happens, i.e., F_{t+1} is a sub-event of F'_t. If $\Pr[F'_t] = 0$, then $\Pr[F_{t+1}] = 0$ and we are done. So assume this is not the case. By the induction hypothesis, we have $\Pr[F'_t] \leq \Pr[F_t] \leq p^t$. So it suffices to bound the conditional probability $\Pr[F_{t+1} | F'_t]$ by p.

Condition on the event F'_t. Consider the moment when algorithm reaches Line 12 with $j = t$, $(i_1, \dots, i_t) = (i_1^*, \dots, i_t^*)$, and $c = i_{t+1}^*$. By Lemma 39, $Q_1, \dots, Q_s$ and a are determined. The set S of assigned variables is also determined via $S = \bigcup_{i=1}^{a-1} \mathrm{Var}(I, J, Q_i)$.

We further fix arbitrary values of α_i for $i \in [n] \setminus \mathrm{Var}(I, J, Q_a)$ consistent with F'_t. It suffices to show that the probability that F_{t+1} occurs (conditioned on F'_t and the fixed α_i for $i \in [n] \setminus \mathrm{Var}(I, J, Q_a)$) is at most p.

By Lemma 38, the elements α_i for $i \in \mathrm{Var}(I, J, Q_a)$ have not been used in the previous iterations of the while loop. Thus, the (conditional) distribution of $(\alpha_i)_{i \in \mathrm{Var}(I,J,Q_a)}$ is the uniform distribution over the set of $|\mathrm{Var}(I, J, Q_a)|$-tuples with distinct coordinates in the set $T := \mathbb{F}_q \setminus \{\alpha_i : i \in [n] \setminus \mathrm{Var}(I, J, Q_a)\}$.

Let $M^* := M|_{X_i = \alpha_i \text{ for } i \in S}$. Then $\overline{M} = M^*|_{X_i = \alpha_i \text{ for } i \in \mathrm{Var}(I,J,Q_a)}$. By Lemma 40, M^* is nonsingular, i.e., $\det(M^*) \neq 0$. View $\det(M^*)$ as a polynomial in the variables X_i, $i \in [n] \setminus (S \cup \mathrm{Var}(I, J, Q_a))$, over the polynomial ring $\mathbb{F}_q[X_i : i \in \mathrm{Var}(I, J, Q_a)]$. Let $Q \in \mathbb{F}_q[X_i : i \in \mathrm{Var}(I, J, Q_a)]$ be the coefficient of a nonzero term of $\det(M^*)$. Then $Q \neq 0$. We also have

$$\deg_{X_i}(Q) \leq \deg_{X_i}(\det(M^*)) \leq 2(k-1)$$

for $i \in \mathrm{Var}(I, J, Q_a)$, where the last inequality holds since X_i appears in at most two rows of M^* and every entry of M^* containing X_i has the form X_i^d with $d \le k-1$. By Lemma 6, we have

$$\Pr[Q(\alpha_i : i \in \mathrm{Var}(I, J, Q_a)) = 0] \le \frac{|\mathrm{Var}(I, J, Q_a)| \cdot 2(k-1)}{|T| - |\mathrm{Var}(I, J, Q_a)| + 1} \le \frac{((1/\varepsilon_0) + 1) \cdot 2(k-1)}{q - n + 1} = p,$$

where the second inequality uses the facts that $|\mathrm{Var}(I, J, Q_a)| \le |Q_a| + 1 \le (1/\varepsilon_0) + 1$ (as (I^{Q_a}, J^{Q_a}) is a chain) and that the size of $T = \mathbb{F}_q \setminus \{\alpha_i : i \in [n] \setminus \mathrm{Var}(I, J, Q_a)\}$ is $q - n + |\mathrm{Var}(I, J, Q_a)|$.

Note that $Q(\alpha_i : i \in \mathrm{Var}(I, J, Q_a))$ is the coefficient of a term of $\det(M^*)|_{X_i = \alpha_i \text{ for } i \in \mathrm{Var}(I,J,Q_a)} = \det(\overline{M})$. So if $\det(\overline{M})$ is zero, so is $Q(\alpha_i : i \in \mathrm{Var}(I, J, Q_a))$. This implies that the probability that $\overline{M}$ is singular is at most p. Note that F_{t+1} occurs only if $\overline{M}$ is singular. (This is because, assuming $\overline{M}$ is not singular, the algorithm will not run Lines 17–22 and store $c = i^*_{t+1}$ into i_{t+1}. Instead, it will run Lines 13–15 and increase c.) Therefore, the (conditional) probability that F_{t+1} occurs is at most p, as desired. □

Taking the union bound over all possible output sequences, we obtain the following corollary.

Corollary 42. *Suppose $q \ge n$ and $(\alpha_1, \dots, \alpha_n)$ is chosen uniformly at random from the set of all n-tuples of distinct elements in $\mathbb{F}_q$. Also suppose $\ell, r \in \mathbb{N}^+$ and $\varepsilon_0 \in (0,1)$ satisfy (6). Then the probability that $V_{k,\ell,I,J}|_{X_1=\alpha_1,\dots,X_n=\alpha_n}$ does not have full column rank is at most $2^{2k+r-2}p^r$, where $p = \frac{((1/\varepsilon_0)+1)\cdot 2(k-1)}{q-n+1}$.*

Proof. By Lemma 36, any sequence $(i_1, \dots, i_r) \in \{0, \dots, 2k-2\}^r$ output by Algorithm 2 satisfies $i_1 \le \dots \le i_r$. The number of such sequences is $\binom{(2k-1)+r-1}{r} \le 2^{2k+r-2}$. By Lemma 41, any fixed sequence is output with probability at most p^r. So by the union bound, the probability that Algorithm 2 outputs a sequence $(i_1, \dots, i_r)$ is at most $2^{2k+r-2}p^r$.

By Lemma 35, Algorithm 2 outputs a sequence $(i_1, \dots, i_r)$ whenever $V_{k,\ell,I,J}|_{X_1=\alpha_1,\dots,X_n=\alpha_n}$ does not have full column rank. So the probability of the latter event is at most $2^{2k+r-2} \cdot p^r$. □

Finally, we prove our main theorem below.

Theorem 43 (Detailed version of Theorem 4)**.** *Let $\varepsilon \in (0,1)$ and $n, k \in \mathbb{N}^+$, where $k \le n$. Let q be a prime power such that $q \ge n + 2^{c/\varepsilon^2}k$, where $c > 0$ is a large enough absolute constant. Suppose $(\alpha_1, \dots, \alpha_n)$ is chosen uniformly at random from the set of all n-tuples of distinct elements in $\mathbb{F}_q$. Then with probability at least $1 - 2^{-n} > 0$, the code $RS_{n,k}(\alpha_1, \dots, \alpha_n)$ over $\mathbb{F}_q$ corrects at least $(1-\varepsilon)n - 2k + 1$ adversarial insdel errors.*

Proof. Let $c_0 > 0$ be a large enough absolute constant, and let $c = c_0^2$. First assume $\varepsilon^2 n \le c_0$. Since $c/\varepsilon^2 \ge c_0 n$, it suffices to prove the claim for $q \ge n + 2^{c_0 n}k = 2^{\Theta(n)}$. In this case, the code can, in fact, correct at least $n - 2k + 1$ adversarial insdel errors. A proof was implicitly given in [CST23, proof of Theorem 16] and also sketched in our Section 1.3.

Next, assume $\varepsilon^2 n \ge c_0$. Choose $\ell = (2k-1) + \lfloor \varepsilon n \rfloor$, $\varepsilon_0 = \varepsilon/4$, and $r = \lceil \frac{\varepsilon^2 n}{16} \rceil$. It is easy to verify that that (6) holds given that the constant c_0 is large enough.

By Lemma 12, if $\mathsf{RS}_{n,k}(\alpha_1, \dots, \alpha_n) \subseteq \mathbb{F}_q^n$ fails to correct $n - \ell$ adversarial insdel errors, then we will have two increasing subsequences $I, J \in [n]^\ell$ which agree on at most $k-1$ coordinates, such that the matrix $V_{k,\ell,I,J}|_{X_1=\alpha_1,\dots,X_n=\alpha_n}$ does not have full column rank. On the other hand, applying Corollary 42 with the chosen parameters ℓ, ε_0, and r, for fixed I, J, we have that

$$\begin{aligned}\Pr\begin{bmatrix}\text{The matrix } V_{k,\ell,I,J}|_{X_1=\alpha_1,\ldots,X_n=\alpha_n} \\ \text{does not have full column rank}\end{bmatrix} &\leq 2^{2k+\lceil\frac{\varepsilon^2 n}{16}\rceil-2}\cdot\left(\frac{((4/\varepsilon)+1)\cdot 2(k-1)}{q-n+1}\right)^{\lceil\frac{\varepsilon^2 n}{16}\rceil} \\ &\leq 2^{3n}\cdot\left(\frac{((4/\varepsilon)+1)\cdot 2(k-1)}{q-n+1}\right)^{\lceil\frac{\varepsilon^2 n}{16}\rceil} \\ &\leq 2^{-3n},\end{aligned}$$

where the last inequality uses the facts that $q \geq n + 2^{c/\varepsilon^2}k$ and that $c > 0$ is a large enough constant. By the union bound over all (I, J), the probability that $\mathsf{RS}_{n,k}(\alpha_1, \ldots, \alpha_n)$ cannot correct $(1-\varepsilon)n - 2k + 1$ adversarial insdel errors is at most $\binom{n}{\ell}^2 \cdot 2^{-3n} \leq 2^{-n}$, as desired. $\square$

References

[AGFC07] Khaled AS Abdel-Ghaffar, Hendrik C Ferreira, and Ling Cheng. On linear and cyclic codes for correcting deletions. In *2007 IEEE International Symposium on Information Theory (ISIT)*, pages 851–855. IEEE, 2007.

[AGL24] Omar Alrabiah, Venkatesan Guruswami, and Ray Li. Randomly punctured Reed–Solomon codes achieve list-decoding capacity over linear-sized fields. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing (STOC)*, pages 1458–1469, 2024.

[BDGZ24] Joshua Brakensiek, Manik Dhar, Sivakanth Gopi, and Zihan Zhang. AG codes achieve list decoding capacity over constant-sized fields. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing (STOC)*, pages 740–751, 2024.

[BGH16] Boris Bukh, Venkatesan Guruswami, and Johan Håstad. An improved bound on the fraction of correctable deletions. *IEEE Transactions on Information Theory*, 63(1):93–103, 2016.

[BGM23] Joshua Brakensiek, Sivakanth Gopi, and Visu Makam. Generic Reed-Solomon codes achieve list-decoding capacity. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing (STOC)*, pages 1488–1501, 2023.

[BGZ17] Joshua Brakensiek, Venkatesan Guruswami, and Samuel Zbarsky. Efficient low-redundancy codes for correcting multiple deletions. *IEEE Transactions on Information Theory*, 64(5):3403–3410, 2017.

[CGHL21] Kuan Cheng, Venkatesan Guruswami, Bernhard Haeupler, and Xin Li. Efficient linear and affine codes for correcting insertions/deletions. In Dániel Marx, editor, *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1–20. SIAM, 2021.

[Che22] Hao Chen. Coordinate-ordering-free upper bounds for linear insertion-deletion codes. *IEEE Transactions on Information Theory*, 68(8):5126–5132, 2022.

[CJL+23] Kuan Cheng, Zhengzhong Jin, Xin Li, Zhide Wei, and Yu Zheng. Linear insertion deletion codes in the high-noise and high-rate regimes. In *50th International*

Colloquium on Automata, Languages, and Programming (ICALP). Schloss-Dagstuhl-Leibniz Zentrum für Informatik, 2023.

[CJLW22] Kuan Cheng, Zhengzhong Jin, Xin Li, and Ke Wu. Deterministic document exchange protocols and almost optimal binary codes for edit errors. *Journal of the ACM*, 69(6):1–39, 2022.

[CR03] Maxime Crochemore and Wojciech Rytter. *Jewels of stringology: text algorithms.* World Scientific, 2003.

[CR20] Mahdi Cheraghchi and João Ribeiro. An overview of capacity results for synchronization channels. *IEEE Transactions on Information Theory*, 67(6):3207–3232, 2020.

[CST22] Roni Con, Amir Shpilka, and Itzhak Tamo. Explicit and efficient constructions of linear codes against adversarial insertions and deletions. *IEEE Transactions on Information Theory*, 68(10):6516–6526, 2022.

[CST23] Roni Con, Amir Shpilka, and Itzhak Tamo. Reed–Solomon codes against adversarial insertions and deletions. *IEEE Transactions on Information Theory*, 2023.

[CST24] Roni Con, Amir Shpilka, and Itzhak Tamo. Optimal two-dimensional Reed–Solomon codes correcting insertions and deletions. *IEEE Transactions on Information Theory*, 2024.

[CZ22] Bocong Chen and Guanghui Zhang. Improved singleton bound on insertion-deletion codes and optimal constructions. *IEEE Transactions on Information Theory*, 68(5):3028–3033, 2022.

[DLTX21] Tai Do Duc, Shu Liu, Ivan Tjuawinata, and Chaoping Xing. Explicit constructions of two-dimensional Reed-Solomon codes in high insertion and deletion noise regime. *IEEE Transactions on Information Theory*, 67(5):2808–2820, 2021.

[Eli57] Peter Elias. List decoding for noisy channels. *Wescon Convention Record, Part 2, Institute of Radio Engineers*, pages 99–104, 1957.

[FKS22] Asaf Ferber, Matthew Kwan, and Lisa Sauermann. List-decodability with large radius for Reed-Solomon codes. In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 720–726. IEEE, 2022.

[GBC$^+$13] Nick Goldman, Paul Bertone, Siyuan Chen, Christophe Dessimoz, Emily M LeProust, Botond Sipos, and Ewan Birney. Towards practical, high-capacity, low-maintenance information storage in synthesized DNA. *Nature*, 494(7435):77–80, 2013.

[GH21] Venkatesan Guruswami and Johan Håstad. Explicit two-deletion codes with redundancy matching the existential bound. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 21–32. SIAM, 2021.

[GHL22] Venkatesan Guruswami, Xiaoyu He, and Ray Li. The zero-rate threshold for adversarial bit-deletions is less than 1/2. *IEEE Transactions on Information Theory*, 69(4):2218–2239, 2022.

[GL16] Venkatesan Guruswami and Ray Li. Efficiently decodable insertion/deletion codes for high-noise and high-rate regimes. In *2016 IEEE International Symposium on Information Theory (ISIT)*, pages 620–624. IEEE, 2016.

[GLS$^+$22] Zeyu Guo, Ray Li, Chong Shangguan, Itzhak Tamo, and Mary Wootters. Improved list-decodability and list-recoverability of Reed-Solomon codes via tree packings. In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 708–719. IEEE, 2022.

[GS98] Venkatesan Guruswami and Madhu Sudan. Improved decoding of Reed-Solomon and algebraic-geometric codes. In *Proceedings 39th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 28–37. IEEE, 1998.

[GST22] Eitan Goldberg, Chong Shangguan, and Itzhak Tamo. List-decoding and list-recovery of Reed–Solomon codes beyond the Johnson radius for every rate. *IEEE Transactions on Information Theory*, 69(4):2261–2268, 2022.

[GW17] Venkatesan Guruswami and Carol Wang. Deletion codes in the high-noise and high-rate regimes. *IEEE Transactions on Information Theory*, 63(4):1961–1970, 2017.

[GXYZ24] Zeyu Guo, Chaoping Xing, Chen Yuan, and Zihan Zhang. Random Gabidulin codes achieve list decoding capacity in the rank metric. *arXiv preprint arXiv:2404.13230*, 2024. To appear in FOCS 2024.

[GZ23] Zeyu Guo and Zihan Zhang. Randomly punctured Reed-Solomon codes achieve the list decoding capacity over polynomial-size alphabets. In *2023 IEEE 64th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 164–176. IEEE, 2023.

[Hae19] Bernhard Haeupler. Optimal document exchange and new codes for insertions and deletions. In *2019 IEEE 60th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 334–347. IEEE, 2019.

[Ham50] Richard W. Hamming. Error detecting and error correcting codes. *Bell System technical journal*, 29(2):147–160, 1950.

[HMG19] Reinhard Heckel, Gediminas Mikutis, and Robert N Grass. A characterization of the DNA data storage channel. *Scientific reports*, 9(1):1–12, 2019.

[HS21a] Bernhard Haeupler and Amirbehshad Shahrasbi. Synchronization strings and codes for insertions and deletions–a survey. *IEEE Transactions on Information Theory*, 67(6):3190–3206, 2021.

[HS21b] Bernhard Haeupler and Amirbehshad Shahrasbi. Synchronization strings: Codes for insertions and deletions approaching the Singleton bound. *Journal of the ACM*, 68(5):1–39, 2021.

[HSRD17] Reinhard Heckel, Ilan Shomorony, Kannan Ramchandran, and NC David. Fundamental limits of DNA storage systems. In *2017 IEEE International Symposium on Information Theory (ISIT)*, pages 3130–3134. IEEE, 2017.

[Joh62] Selmer Johnson. A new upper bound for error-correcting codes. *IRE Transactions on Information Theory*, 8(3):203–207, 1962.

[JZCW23] Qinqin Ji, Dabin Zheng, Hao Chen, and Xiaoqiang Wang. Strict half-Singleton bound, strict direct upper bound for linear insertion-deletion codes and optimal codes. *IEEE Transactions on Information Theory*, 2023.

[Lev66] Vladimir I Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710, 1966.

[Lev02] Vladimir I Levenshtein. Bounds for deletion/insertion correcting codes. In *Proceedings IEEE International Symposium on Information Theory (ISIT)*, page 370. IEEE, 2002.

[Liu24] Jingge Liu. Optimal RS codes and GRS codes against adversarial insertions and deletions and optimal constructions. *IEEE Transactions on Information Theory*, 2024.

[LSWZY19] Andreas Lenz, Paul H Siegel, Antonia Wachter-Zeh, and Eitan Yaakobi. Coding over sets for DNA storage. *IEEE Transactions on Information Theory*, 66(4):2331–2351, 2019.

[LT21] Shu Liu and Ivan Tjuawinata. On 2-dimensional insertion-deletion Reed-Solomon codes with optimal asymptotic error-correcting capability. *Finite Fields and Their Applications*, 73:101841, 2021.

[MBT10] Hugues Mercier, Vijay K Bhargava, and Vahid Tarokh. A survey of error-correcting codes for channels with symbol synchronization errors. *IEEE Communications Surveys & Tutorials*, 12(1):87–96, 2010.

[Mit09] Michael Mitzenmacher. A survey of results for deletion channels and related synchronization channels. *Probability Surveys*, 6:1–33, 2009.

[RW14] Atri Rudra and Mary Wootters. Every list-decodable code for high noise has abundant near-optimal rate puncturings. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing (STOC)*, pages 764–773, 2014.

[RZVW24] Noga Ron-Zewi, S Venkitesh, and Mary Wootters. Efficient list-decoding of polynomial ideal codes with optimal list size. *arXiv preprint arXiv:2401.14517*, 2024.

[SB20] Jin Sima and Jehoshua Bruck. On optimal k-deletion correcting codes. *IEEE Transactions on Information Theory*, 67(6):3360–3375, 2020.

[SGB20] Jin Sima, Ryan Gabrys, and Jehoshua Bruck. Optimal systematic t-deletion correcting codes. In *2020 IEEE International Symposium on Information Theory (ISIT)*, pages 769–774. IEEE, 2020.

[SH22] Ilan Shomorony and Reinhard Heckel. Information-theoretic foundations of DNA data storage. *Foundations and Trends® in Communications and Information Theory*, 19(1):1–106, 2022.

[Sha48] Claude Elwood Shannon. A mathematical theory of communication. *Bell system technical journal*, 27(3):379–423, 1948.

[SNW02] Reihaneh Safavi-Naini and Yejing Wang. Traitor tracing for shortened and corrupted fingerprints. In *ACM workshop on Digital Rights Management*, pages 81–100. Springer, 2002.

[ST20] Chong Shangguan and Itzhak Tamo. Combinatorial list-decoding of Reed-Solomon codes beyond the Johnson radius. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pages 538–551, 2020.

[Sud97] Madhu Sudan. Decoding of Reed Solomon codes beyond the error-correction bound. *Journal of Complexity*, 13(1):180–193, 1997.

[SWZGY17] Clayton Schoeny, Antonia Wachter-Zeh, Ryan Gabrys, and Eitan Yaakobi. Codes correcting a burst of deletions or insertions. *IEEE Transactions on Information Theory*, 63(4):1971–1985, 2017.

[SZ99] Leonard J Schulman and David Zuckerman. Asymptotically good codes correcting insertions, deletions, and transpositions. *IEEE Transactions on Information Theory*, 45(7):2552–2557, 1999.

[TSN07] Dongvu Tonien and Reihaneh Safavi-Naini. Construction of deletion correcting codes using generalized Reed–Solomon codes and their subcodes. *Designs, Codes and Cryptography*, 42(2):227–237, 2007.

[VT65] RR Varshamov and GM Tenengolts. Codes which correct single asymmetric errors (in Russian). *Automatika i Telemkhanika*, 161(3):288–292, 1965.

[WMSN04] Yejing Wang, Luke McAven, and Reihaneh Safavi-Naini. Deletion correcting using generalized Reed-Solomon codes. In *Coding, Cryptography and Combinatorics*, pages 345–358. Springer, 2004.

[Woz58] John M Wozencraft. List decoding. *Quarterly Progress Report*, 48:90–95, 1958.

[Yas24] Kenji Yasunaga. Improved bounds for codes correcting insertions and deletions. *Designs, Codes and Cryptography*, pages 1–12, 2024.