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

Roni Con
Department of Computer Science, Technion–Israel Institute of Technology, Haifa, Israel

Zeyu Guo
Department of Computer Science and Engineering, The Ohio State University, Columbus, OH, USA

Ray Li
Mathematics and Computer Science Department, Santa Clara University, CA, USA

Zihan Zhang
Department of Computer Science and Engineering, The Ohio State University, Columbus, OH, USA

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.

2012 ACM Subject Classification Mathematics of computing → Coding theory

Keywords and phrases coding theory, error-correcting codes, Reed-Solomon codes, insdel, insertion-deletion errors, half-Singleton bound

Digital Object Identifier 10.4230/LIPIcs.ICALP.2025.60

Category Track A: Algorithms, Complexity and Games

Related Version *Full Version*: `https://arxiv.org/abs/2407.07299`

Funding *Roni Con*: Supported by the European Union (DiDAX, 101115134). Views and opinions expressed are however those of the authors only and do not necessarily reflect those of the European Union or the European Research Council Executive Agency. Neither the European Union nor the granting authority can be held responsible for them.
Zeyu Guo: Supported by NSF Grant CCF-2440926.
Ray Li: Supported by NSF Grant CCF-2347371
Zihan Zhang: Supported in part by NSF Grant CCF-2440926.

Acknowledgements 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, the last author wishes to thank Prof. Xin Li for the helpful discussions and suggestions.

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 [49] studied random errors and erasures, whereas Hamming [32] 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 [42]). 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 [6, 31, 5, 28, 46, 11, 29, 9, 24, 25], 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 [43, 42, 12, 30]).

Codes that can correct insdel errors also attract a lot of attention due to their possible application in designing DNA-based storage systems [20]. 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 [34, 37, 33, 50].

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 [1, 9], 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 [9, 8, 13, 14, 35, 10, 40].

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 [45, 56, 54, 16, 41, 7, 14, 15, 40]. 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 [23, 2], 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 [23, 2] 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 [38] and Varshamov and Tenengolts [55], 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 [47].

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 [24, 51, 52]; (ii) finding explicit binary codes for correcting εn insdels for small constant $\varepsilon > 0$ with optimal rate [11, 29], (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 $[\sqrt{2}-1, 1/2-10^{-40}]$) [26, 25], and (iv) determining the optimal rate-versus distance tradeoff for deletion codes [38, 39, 58], 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 [31], 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 [30] 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 [1]. 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 [9]. 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 ([9, 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 [9, 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 [9, 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 [13, 10]. 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*

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

The performance of RS codes against insdel errors was first considered in [45] in the context of traitor tracing. In [56], the authors constructed a $[5,2]_q$ RS code correcting a single deletion. In [54], 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 [16, 41]. Note that the half-Singleton bound states that an $[n,2]$ code can correct at most $n-3$ deletions. In [14], 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, [15] 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 [14] that over large enough fields, there exist RS codes that exactly achieve the half-Singleton bound. Specifically,

▶ **Theorem 3** ([14, 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]

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 [17, 57] 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 [53] and Guruswami–Sudan [27], which provided efficient algorithms for list decoding RS codes up to the Johnson bound [36], the issue of understanding the list decodability of RS codes beyond the Johnson bound became very important. A long line of works [44, 48, 21, 18, 19, 4, 23, 2, 3, 22] have made significant advancements in the understanding of list decodability for RS and related codes. Specifically, recent works [4, 23, 2] 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 $(\alpha_1, \ldots, \alpha_n)$ is uniformly distributed over the set of all n-tuples of distinct elements in $\mathbb{F}_q$, we say the code $\mathrm{RS}_{n,k}(\alpha_1, \ldots, \alpha_n)$ 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 40). *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 [14], where they have $q = 2^{\Theta(n)}$. As a warmup to this result, we prove a weaker but more straightforward result (Theorem 22), which establishes Theorem 4 for $q = 2^{O(1/\varepsilon)} n^2$.

[1] The alphabet size in [14, 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.

1.3 Proof overview

We outline the proof of our main theorem in this section. First, we review the proof of Theorem 3 [14] 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 22) 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 [14] that proves that Reed–Solomon codes achieve the half-Singleton bound over exponential-sized alphabets $q = 2^{\Theta(n)}$. Let $\ell \stackrel{\text{def}}{=} 2k-1$. 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 $I_1 < \cdots < I_\ell$ and $J_1 < \cdots < J_\ell$ 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 $f(X) = f_0 + f_1 X + \cdots + f_{k-1} X^{k-1}$ and $f'(X) = f'_0 + f'_1 X + \cdots + f'_{k-1} X^{k-1}$, 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 $I_1 < \cdots < I_\ell$ and $J_1 < \cdots < J_\ell$, so there are at most 2^{2n} possible V-matrices. Hence, by the union bound, the

[2] An important detail here, which [14] proves, is that (to apply the Schwartz-Zippel lemma) the determinant needs to be symbolically nonzero.

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 [23, 2], 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 $\ell \stackrel{\text{def}}{=} (2k-1)+\varepsilon n$. 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 $\frac{k(k-1)}{q-n}$.

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 $v_1, \dots, v_{i-1}$. 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 21.

▶ **Corollary 5** (Corollary 21, 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,*

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

At the highest level, our proof of Corollary 21 is a union bound over "certificates." For all evaluation points $(\alpha_1,\dots,\alpha_n) \in \mathbb{F}_q^n$ where $V_{k,\ell,I,J}$ is bad, we show that there is a *certificate* $(i_1,\dots,i_r) \in [n]^r$ of distinct indices in $[n]$ (Lemma 19) that intuitively "attests" that the matrix $V_{k,\ell,I,J}$ is bad.

We generate the certificate $(i_1,\dots,i_r)$ deterministically from the evaluations $\alpha_1,\dots,\alpha_n$ using Algorithm 1. We compute the certificate one index i_j at a time. Given indices $i_1,\dots,i_{j-1}$, define index i_j as follows: let A_j be the top $(2k-1)\times(2k-1)$ square submatrix of $V_{k,\ell,I,J}^{\{i_1,\dots,i_{j-1}\}}$ – the V-matrix $V_{k,\ell,I,J}$ after removing rows containing any of variables $X_{i_1}, X_{i_2},\dots,X_{i_{j-1}}$ – and let i_j be the smallest index such that $A_j|_{X_1=\alpha_1,\dots,X_{i_j}=\alpha_{i_j}}$ is not full column rank (we call i_j a *faulty index*, Definition 18). Since A_j is a full rank submatrix of a bad V-matrix[4], $A_j|_{X_1=\alpha_1,\dots,X_n=\alpha_n}$ 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, $V_{k,\ell,I,J}^{\{i_1,\dots,i_{j-1}\}}$, 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 $\left\lfloor\frac{\ell-(2k-1)}{2}\right\rfloor + 1 \geq \varepsilon n/2 = r$.

[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 [14].

We then show (Lemma 20), 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 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 $X_1, \ldots, X_n$ (in the proof, these blocks are given by *chains*, Definition 24). 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 $A_2, A_3, \ldots$ 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 $A_{j+1}|_{X_1=\alpha_1,\ldots,X_{i_j}=\alpha_{i_j}}$ 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.

[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.

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 [14], 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.
4. *Affine codes.* In [9, 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?

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 denote 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 [23], regarding the partial assignment of a symbolic matrix.

▶ **Definition 6** (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 7.** *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 $\Pr[Q(\alpha_1, \dots, \alpha_n) = 0] \le \frac{nd}{|T|-n+1}$.*

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

▶ **Definition 8.** *A* subsequence *of a string s is a string obtained by removing some (possibly none) of the symbols in s.*

▶ **Definition 9.** *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 10.** *A code C can correct δn insdel errors if and only if* $\mathrm{LCS}(c, c') \leq n - \delta n - 1$ *for any distinct $c, c' \in C$.*

We now adopt two definitions and a lemma from [14]. These will establish the algebraic conditions ensuring that an RS code can correct insdel errors.

▶ **Definition 11** (Increasing subsequence)**.** *We call $I = (I_1, \dots, I_\ell) \in [n]^\ell$ an* increasing subsequence *if it holds that $I_1 < I_2 < \dots < I_\ell$, where ℓ is called the* length *of I.*

▶ **Definition 12** (V-matrix)**.** *For positive integers ℓ, k and increasing subsequences $I = (I_1, \dots, I_\ell), J = (J_1, \dots, J_\ell) \in [n]^\ell$ 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 $\mathbb{F}_q(X_1, \dots, X_n)$. 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 $\mathsf{RS}_{n,k}(\alpha_1, \dots, \alpha_n) \subseteq \mathbb{F}_q^n$ 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 [14].

▶ **Lemma 13.** *Let $\ell \geq 2k - 1$. Consider the $[n, k]$ Reed–Solomon code $\mathsf{RS}_{n,k}(\alpha_1, \dots, \alpha_n) \subseteq \mathbb{F}_q^n$ associated with an evaluation vector $\vec{\alpha} := (\alpha_1, \dots, \alpha_n) \in \mathbb{F}_q^n$. If the code cannot correct arbitrary $n - \ell$ insdel errors, then there exist two increasing subsequences $I = (I_1, \dots, I_\ell), J = (J_1, \dots, J_\ell) \in [n]^\ell$ that agree on at most $k - 1$ coordinates such that matrix $V_{k,\ell,I,J}|_{X_1=\alpha_1,\dots,X_n=\alpha_n}$ does not have full column rank.*

A key technical lemma regarding the V-matrices that was proved in [14] is the following.

▶ **Lemma 14** (Proposition 2.4, [14])**.** *Let $I, J \in [n]^{2k-1}$ be two increasing subsequences that agree on at most $k - 1$ coordinates. Then, we have $\det\left(V_{k,2k-1,I,J}(X_1, X_2, \dots, X_n)\right) \neq 0$ as a multivariate polynomial in $\mathbb{F}_q[X_1, X_2, \dots, X_n]$.*

Informally speaking, by using this lemma, and by considering all the V-matrices, the authors of [14] showed by using the Schwarz-Zippel lemma that there exists an assignment $(\alpha_1, \dots, \alpha_n)$ over a large enough field for which $\det(V_{k,\ell,I,J}|_{X_1=\alpha_1,\dots,X_n=\alpha_n})$ 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 15** (Selecting a subset of coordinates). *For an increasing subsequence $I \in [n]^\ell$ and $P = \{i_1, \dots, i_{\ell'}\} \subseteq [\ell]$ with $i_1 < \cdots < i_{\ell'}$, denote by I^P the increasing subsequence $(I_{i_1}, \dots, I_{i_{\ell'}})$.*

The following is a corollary of Lemma 14.

▶ **Corollary 16.** *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 $\deg_{X_i}(\det(V_{k,2k-1,I^P,J^P})) \leq 2(k-1)$.*

3 Achieving Quadratic Alphabet Size

In this section, we present a warm-up result, Theorem 22, which achieves the alphabet size $O_\varepsilon(n^2)$. Our proof resembles the proof by Guo and Zhang [23] 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 17.** *Under the notation in Definition 12, 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 [23].

▶ **Definition 18** (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.

▶ **Lemma 19** (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 $B_j := \{i_1, \dots, i_{j-1}\}$.*

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 $\alpha_1, \ldots, \alpha_n \in \mathbb{F}_q$.

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

$B \leftarrow \emptyset$.
for $j = 1$ **to** r **do**
  if $\operatorname{rank}(V^B_{k,\ell,I,J}) < 2k - 1$ **then**
    Output "FAIL" and halt.
  else if *a faulty index* $i \in [n]$ *of* $V^B_{k,\ell,I,J}$ *exists* **then** `//` i `is unique if it exists`
    $i_j \leftarrow i$ and $B \leftarrow B \cup \{i\}$.
  else
    Output "SUCCESS" and halt.
  end
end
Output $(i_1, \ldots, i_r)$.

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 [23, Lemma 4.5].

▶ **Lemma 20.** *Under the notation and conditions in Lemma 19, 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 $\left(\frac{2(k-1)}{q-n+1}\right)^r$.*

By combining Lemma 19 and Lemma 20 and taking a union bound over the set of possible outputs $(i_1, \ldots, i_r)$ of Algorithm 1, we obtain the following corollary.

▶ **Corollary 21.** *Under the notation and conditions in Lemma 19, 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 positive integer $r \leq \lceil \frac{\varepsilon n}{2} \rceil$, we have*

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

3.2 Putting it together

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

▶ **Theorem 22.** *Let $\varepsilon \in (0,1)$ and $n, k \in \mathbb{N}^+$, where $k \leq n$. Let q be a prime power such that $q \geq \left(1 + 2 \cdot 2^{6/\varepsilon} k\right) n$. Suppose $(\alpha_1, \ldots, \alpha_n)$ is chosen uniformly at random from the set of all n-tuples of distinct elements in $\mathbb{F}_q$. Then with probability at least $1 - 2^{-n} > 0$, the code $RS_{n,k}(\alpha_1, \ldots, \alpha_n)$ over $\mathbb{F}_q$ corrects at least $(1-\varepsilon)n - 2k + 1$ adversarial insdel errors.*

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. The proof of Theorem 4 – which is omitted due to space limitations – is built on these foundations.

4.1 Chain decomposition

For convenience, we first introduce some definitions.

▶ **Definition 23.** *For an increasing subsequence* $I = (I_1, \ldots, I_\ell) \in [n]^\ell$*, define* $\mathrm{Set}(I) := \{I_1, \ldots, I_\ell\}$.

▶ **Definition 24** (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.*

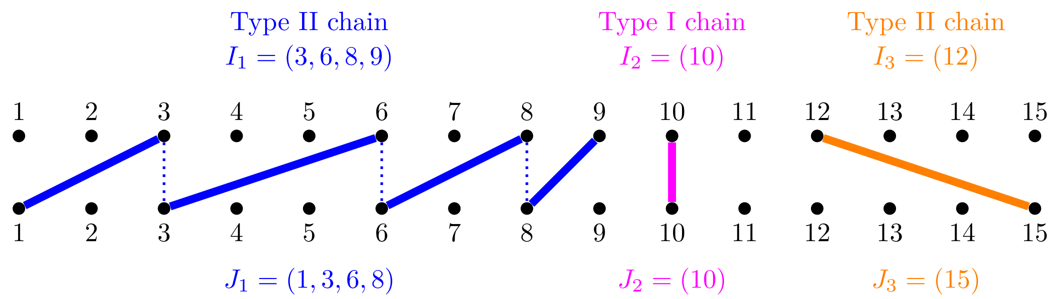

Figure 1 Example of chains.

Recall Definition 15 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 25** (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*

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* $\mathrm{Set}(I)$ *and* $\mathrm{Set}(J)$.

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

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

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

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

▶ **Definition 27** ((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* $\mathrm{Var}(I,J,P)$ *and* $\mathrm{Var}(I,J,P')$ *are disjoint. Note that (I,J)-disjointness implies disjointness.*

The next theorem states that for increasing subsequences $I, J \in [n]^\ell$, (I,J) can always be decomposed into maximal chains. Its proof is omitted.

▶ **Theorem 28.** *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 $i \in [s]$.*

Splitting long chains into short ones

The maximal chains in Theorem 28 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 2.)

▶ **Lemma 29.** *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.*
2. *$|P_i| \le 1/\varepsilon$ for $i \in [s]$.*

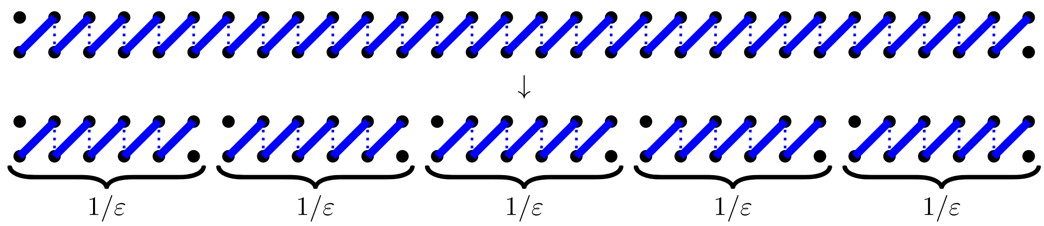

Figure 2 Lemma 29: splitting long chains into short ones. Ensure that all chains have a length of at most $1/\varepsilon$ by removing at most ε fraction of pairs from each chain.

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 30.** *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 31.** *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.*

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 29, 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 $\sum_{i=1}^{s} |P_i| \geq (1-\varepsilon_0)\ell$ 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 32.** $|P_1| \leq |P_2| \leq \cdots \leq |P_s|$. 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 32 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 31, 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. 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 below, 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 13–15).

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 $i_j \leq 2k-2$ 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

$$(1-\varepsilon_0)\ell \geq 2k - 1 + (r+1)/\varepsilon_0. \tag{3}$$

[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.

Algorithm 2 `CertifyFullColumnRankness2`

Input: $n, k, \ell, r \in \mathbb{N}^+$, $\varepsilon_0 \in (0,1)$, increasing subsequences $I, J \in [n]^\ell$, $P_1, \dots, P_s \subseteq [\ell]$, and $\alpha_1, \dots, \alpha_n \in \mathbb{F}_q$.
Output: "SUCCESS" or a sequence $(i_1, \dots, i_r) \in \{0, \dots, 2k-2\}^r$.

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 $\quad M \leftarrow$ the top $(2k-1) \times (2k-1)$ submatrix of $\widetilde{V}_{k,I,J,(Q_i)_{i=1}^{s-m}}$.
10 $\quad S' \leftarrow S \cup \mathrm{Var}(I, J, Q_a)$.
11 $\quad \overline{M} \leftarrow M|_{X_i = \alpha_i \text{ for } i \in S'}$.
12 $\quad$**if** $\overline{M}$ *is nonsingular* **then** `// assign variables in the` a`-th chain`
13 $\quad\quad S \leftarrow S'$.
14 $\quad\quad c \leftarrow c + |Q_a|$.
15 $\quad\quad a \leftarrow a + 1$.
16 $\quad$**else** `// replace the` a`-th chain by (part of) the` b`-th chain in the bank`
17 $\quad\quad Q_a \leftarrow$ set of the smallest $|Q_a|$ elements in Q_b.
18 $\quad\quad Q_b \leftarrow \emptyset$.
19 $\quad\quad b \leftarrow b + 1$.
20 $\quad\quad j \leftarrow j + 1$.
21 $\quad\quad i_j \leftarrow c$.
22 $\quad\quad$**if** $j = r$ **then** output $(i_1, \dots, i_r)$ and halt.
23 $\quad$**end**
24 **end**
25 Output "SUCCESS".

Line 3 of Algorithm 2 is valid by 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 $\sum_{i=1}^{s-m} |P_i| \geq 2k-1$ and $m \geq r$.*

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

▶ **Lemma 34.** *Algorithm 2 terminates, outputting either "SUCCESS" or $(i_1, \dots, i_r) \in \{0, \dots, 2k-2\}^r$. In the former case, $V_{k,\ell,I,J}|_{X_1=\alpha_1,\dots,X_n=\alpha_n}$ has full column rank.*

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

▶ **Lemma 35.** *Any sequence $(i_1, \dots, i_r)$ output by Algorithm 2 satisfies $i_1 \leq \dots \leq i_r$.*

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

▶ **Lemma 36.** *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 $\alpha_1, \ldots, \alpha_n$.*

We also need the following crucial lemma.

▶ **Lemma 37.** *At Line 12 of Algorithm 2, $M|_{X_i=\alpha_i \text{ for } i\in S}$ is nonsingular.*

The next lemma bounds the probability that Algorithm 2 outputs a particular sequence.

▶ **Lemma 38.** *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 $\mathbb{F}_q$. The probability that Algorithm 2 outputs a fixed sequence $(i_1^*, \ldots, i_r^*) \in \{0, \ldots, 2k-2\}^r$ is at most p^r, where $p = \frac{((1/\varepsilon_0)+1)\cdot 2(k-1)}{q-n+1}$.*

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

▶ **Corollary 39.** *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 $\mathbb{F}_q$. Also suppose $\ell, r \in \mathbb{N}^+$ and $\varepsilon_0 \in (0,1)$ satisfy (3). Then the probability that $V_{k,\ell,I,J}|_{X_1=\alpha_1,\ldots,X_n=\alpha_n}$ does not have full column rank is at most $2^{2k+r-2}p^r$, where $p = \frac{((1/\varepsilon_0)+1)\cdot 2(k-1)}{q-n+1}$.*

Finally, we state our main theorem below, whose proof is based on Algorithm 2.

▶ **Theorem 40** (Detailed version of Theorem 4)**.** *Let $\varepsilon \in (0,1)$ and $n, k \in \mathbb{N}^+$, where $k \leq n$. Let q be a prime power such that $q \geq n + 2^{c/\varepsilon^2}k$, where $c > 0$ is a large enough absolute constant. Suppose $(\alpha_1, \ldots, \alpha_n)$ is chosen uniformly at random from the set of all n-tuples of distinct elements in $\mathbb{F}_q$. Then with probability at least $1-2^{-n} > 0$, the code $RS_{n,k}(\alpha_1, \ldots, \alpha_n)$ 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 $\varepsilon^2 n \leq c_0$. Since $c/\varepsilon^2 \geq c_0 n$, it suffices to prove the claim for $q \geq n + 2^{c_0 n}k = 2^{\Theta(n)}$. In this case, the code can, in fact, correct at least $n - 2k + 1$ adversarial insdel errors. A proof was implicitly given in [14, proof of Theorem 16] and also sketched in our Section 1.3.

Next, assume $\varepsilon^2 n \geq c_0$. 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 (3) holds given that the constant c_0 is large enough.

By Lemma 13, 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 39 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. ◀

References

1 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.

2 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. doi:10.1145/3618260.3649634.

3 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. doi:10.1145/3618260.3649651.

4 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. doi:10.1145/3564246.3585128.

5 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. doi:10.1109/TIT.2017.2746566.

6 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. doi:10.1109/TIT.2016.2621044.

7 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. doi:10.1109/TIT.2022.3148185.

8 Hao Chen. Coordinate-ordering-free upper bounds for linear insertion-deletion codes. *IEEE Transactions on Information Theory*, 68(8):5126–5132, 2022. doi:10.1109/TIT.2022.3167662.

9 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. doi:10.1137/1.9781611976465.1.

10 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.

11 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. doi:10.1145/3561046.

12 Mahdi Cheraghchi and João Ribeiro. An overview of capacity results for synchronization channels. *IEEE Transactions on Information Theory*, 67(6):3207–3232, 2020. doi:10.1109/TIT.2020.2997329.

13 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. doi:10.1109/TIT.2022.3173185.

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

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

16 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. doi:10.1109/TIT.2021.3065618.

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

18 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.

19 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. doi:10.1109/TIT.2022.3222877.

20 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. doi:10.1038/NATURE11875.

21 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.

22 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. doi:10.48550/arXiv.2404.13230.

23 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. doi:10.1109/FOCS57990.2023.00019.

24 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. doi:10.1137/1.9781611976465.2.

25 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. doi:10.1109/TIT.2022.3223023.

26 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. doi:10.1109/ISIT.2016.7541373.

27 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.

28 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. doi:10.1109/TIT.2017.2659765.

29 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. doi:10.1109/FOCS.2019.00029.

30 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. doi:10.1109/TIT.2021.3056317.

31 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. doi:10.1145/3468265.

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

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

34 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.

35 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.

36 Selmer Johnson. A new upper bound for error-correcting codes. *IRE Transactions on Information Theory*, 8(3):203–207, 1962. doi:10.1109/TIT.1962.1057714.

37 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. doi:10.1109/TIT.2019.2961265.

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

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

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

41 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. doi:10.1016/J.FFA.2021.101841.

42 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. doi:10.1109/SURV.2010.020110.00079.

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

44 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. doi:10.1145/2591796.2591797.

45 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. doi:10.1007/978-3-540-44993-5_6.

46 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. doi:10.1109/TIT.2017.2661747.

47 Leonard J Schulman and David Zuckerman. Asymptotically good codes correcting insertions, deletions, and transpositions. *IEEE Transactions on Information Theory*, 45(7):2552–2557, 1999. doi:10.1109/18.796406.

48 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. doi:10.1145/3357713.3384295.

49 Claude Elwood Shannon. A mathematical theory of communication. *Bell system technical journal*, 27(3):379–423, 1948. doi:10.1002/J.1538-7305.1948.TB01338.X.

50 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. doi:10.1561/0100000117.

51 Jin Sima and Jehoshua Bruck. On optimal k-deletion correcting codes. *IEEE Transactions on Information Theory*, 67(6):3360–3375, 2020. doi:10.1109/TIT.2020.3028702.

52 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. doi:10.1109/ISIT44484.2020.9173986.

53 Madhu Sudan. Decoding of Reed Solomon codes beyond the error-correction bound. *Journal of Complexity*, 13(1):180–193, 1997. doi:10.1006/JCOM.1997.0439.

54 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. doi:10.1007/S10623-006-9032-7.

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

56 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.

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

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