

# Deterministic List Decoding of Reed Solomon Codes

---

Soham Chatterjee

January 29, 2026

# Introduction

---

# Introduction to Coding Theory

An Error-Correcting code or simply code,  $C \subseteq \Sigma^n$  for some fixed finite set of alphabets  $\Sigma$ . You have a set of messages  $M$  and encode them to  $C$ .

- Blocklength:  $n$
- Dimension of Code:  $k = \log |C|$
- Rate of Code:  $R(C) = \frac{k}{n \log |\Sigma|}$

The distance between two codewords  $c_1 \neq c_2 \in C$  is the hamming distance between them,  $\Delta(c_1, c_2)$ .

- Distance of Code:  $\Delta(C) = \min_{c_1 \neq c_2 \in C} \Delta(c_1, c_2)$
- Relative Distance:  $\delta(C) = \frac{\Delta(C)}{n}$

# Introduction to Coding Theory

**Goal:** Construct codes such that

- Codewords to be “far apart” from each other  $\implies$  High Distance
- Redundancy to be low  $\implies$  High Rate

## Relation between Rate and Distance

For any code  $C$

$$k \leq n - d + 1$$

Asymptotically  $R + \delta \leq 1$  as  $n$  becomes very large.

Codes achieving this bound are called **Maximum Distance Separable (MDS)** codes.

- Reed Solomon Codes are MDS codes.

# Unique Decoding

Let  $\Delta(C) = d$ . For any  $v \in \Sigma^n$  there is at most one codeword  $c \in C$  such that  $\Delta(v, c) \leq (d - 1)/2$ .

**Unique Decoding Problem:** Given a received word  $v \in \Sigma^n$ , find the unique codeword  $c \in C$  such that  $\Delta(v, c) < d/2$  if it exists.

- If we go more than  $d/2$  distance, multiple codewords may lie in the radius of hamming ball.

# List Decoding

## Definition ( $(\rho, L)$ -List Decodable)

$C$  is called  $(\rho, L)$ -list decodable if for every  $v \in \Sigma^n$ ,

$$|\{c \in C \mid \Delta(c, v) \leq \rho n\}| \leq L$$

We denote the list for  $v$  by  $L(v)$ .

**List Decoding Problem:** Given a received word  $v \in \Sigma^n$

- Combinatorial List Decoding: If  $|L(v)| = \text{poly}(n)$
- Algorithmic List Decoding: Find all codewords in  $L(v)$  in  $\text{poly}(n)$  time.

## Theorem (Johnson Bound)

*For a code  $C$  with rate  $R$  the list size remains polynomial in  $n$  for  $\rho \leq 1 - \sqrt{R}$*

We will talk in terms of agreement.

$$1 - t \text{ fraction of errors} \implies t \text{ fraction of agreement}$$

# Reed Solomon Codes

---

# Reed Solomon Codes

Fix the following

- Alphabets: finite field  $\mathbb{F}_q$  of size  $q$
- Subset  $S \subseteq \mathbb{F}_q$ ,  $|S| = n$ .  $S = \{\alpha_1, \alpha_2, \dots, \alpha_n\}$

$RS[n, k]_q$  encodes every message polynomial  $f(X) \in \mathbb{F}_q[X]$  with  $\deg(f) < k$  to

$$(f(\alpha_1), f(\alpha_2), \dots, f(\alpha_n))$$

- Rate  $R = \frac{k}{n}$
- Distance  $d = n - k + 1$



# Reed Solomon Decoding History

**Want:** unique and list decoding in  $\text{poly}(n, \log |\mathbb{F}_q|)$  time.

Deterministic unique decoding upto  $(n - k + 1)/2$  errors is possible using **Berlekamp-Welch** algorithm in  $\text{poly}(n, \log |\mathbb{F}|)$  time.

Johnson Bound gives polynomial list size for more than  $\sqrt{n(k-1)}$  agreement

- **Sudan (1997)** gave randomized list decoding for more than  $\sqrt{2n(k-1)}$  agreement in  $\text{poly}(n, \log q)$  time.
- **Guruswami-Sudan (1999)** improved it to more than  $\sqrt{n(k-1)}$  agreement using randomization in  $\text{poly}(n, \log q)$  time.

Their Deterministic variant has polynomial dependence on field characteristic

# Framework of Sudan and Guruswami-Sudan

Let  $w = \{(\alpha_j, \beta_j)\}_{j \in [n]}$  denote the received word. Both algorithms share the same two-step structure:

- **Interpolation:** Find a nonzero polynomial  $Q(X, Y) \in \mathbb{F}_q[X, Y]$  of  $(1, k-1)$ -degree at most  $D$  that vanishes at each  $(\alpha_j, \beta_j)$  with multiplicity at least  $m$ .
- **Factorization:** Factorize  $Q(X, Y)$  over  $\mathbb{F}_q$ ; for each factor of the form  $Y - f(X)$ , output  $f$  if  $\deg f < k$  and  $f$  agrees with  $w$  on at least  $t$  evaluations.

Sudan	Guruswami-Sudan
$m = 1, \quad D, t \approx \sqrt{2n(k-1)}$	$m = \sqrt{n(k-1)}, \quad D \approx m\sqrt{n(k-1)},$ $t = \sqrt{n(k-1)}$

Both algorithms we denote  $Q$  be the polynomial from interpolation step.

# Factorization Barrier

Sudan and Guruswami–Sudan algorithms rely on the factorization of bivariate polynomials over finite fields

- Barlekamp, Cantor-Zassenhaus, LLL, Kaltofen runs in polynomial time but randomized.
- Their deterministic variants have polynomial dependence on field characteristic.

Large field characteristic (super polynomial in  $n$ ) is a problem.

**Want:** Do the factorization step deterministically in  $\text{poly}(n, \log q)$  time.

# **Derandomization of Sudan**

---

# Newton Iteration

Let  $P(X, Y) \in \mathbb{F}_q[X, Y]$  and  $f(X) \in \mathbb{F}_q[X]$  such that

- $P(X, f(X)) \equiv 0$  and
- $\alpha \in \mathbb{F}_q$  such that  $\frac{\partial}{\partial Y} P(\alpha, f(\alpha)) \neq 0$ .
- $Y_t = f(X) \pmod{X^t}$  for all  $t \in \mathbb{N}$

Newton Iteration gives an efficient way to compute  $Y_{t+1}$  from  $Y_t$  as follows:

$$Y_{t+1} = Y_t - \frac{P(X, Y_t)}{\partial_Y P(X, Y_t)}$$

# Sudan Derandomization

Let  $f$  is in the list. And let  $j \in [n]$  such that  $f(\alpha_j) = \beta_j$ . Suppose  $Q$  is the polynomial from interpolation step.

If  $\partial_Y Q(\alpha_j, f(\alpha_j)) \neq 0$ : Then Newton Iteration from  $Y_0$  till  $Y_k$  gives  $f$ .

Else  $\partial_Y Q(\alpha_j, f(\alpha_j)) = 0$  for all  $j$  in agreement.

**Observe:**  $\partial_Y(Q(X, f(X)))$  has more than  $\sqrt{2n(k-1)}$  roots but degree at most  $\sqrt{2n(k-1)}$ . Implying  $\partial_Y(Q(X, f(X))) \equiv 0$

So recurse on  $\partial_Y(Q(X, f(X)))$ .

# Sudan Derandomization

## Algorithm:

1. Check for each  $j \in [n]$  if  $\partial_Y Q(\alpha_j, \beta_j) \neq 0$ . If yes, do Newton Iteration from there.
2. Else compute  $\partial_Y(Q(X, Y))$  and continue from step 1 with  $\partial_Y Q(X, Y)$  instead of  $Q$ .

## Theorem

*There is a deterministic algorithm that, for every finite field  $\mathbb{F}$  and parameters  $n, k \in \mathbb{N}$  runs in time  $\text{poly}(n, \log |\mathbb{F}|)$  list decodes Reed Solomon code  $RS[n, k]$  from agreement more than  $\sqrt{2n(k-1)}$ .*

# **Derandomization of Guruswami-Sudan**

---



# Local Splitting

Let  $P(X, Y) \in \mathbb{F}_q[X, Y]$  with no-pure  $X$ -factors. Let  $(\alpha, \beta) \in \mathbb{F}_q^2$  any point.

Suppose we are given the factorization  $P = \prod_{i=1}^s P_i$  into irreducibles (with multiplicity)

Eg:  $P(X, Y) = (Y^2 + X)(Y^2 + X + 1)^2$  then

$$P_1 = Y^2 + X, \quad P_2 = Y^2 + X + 1, \quad P_3 = Y^2 + X + 1$$

We partition  $[s]$  into four sets which defines four types of factors at  $(\alpha, \beta)$ :

$$A(\alpha, \beta), \quad B(\alpha, \beta), \quad C(\alpha, \beta), \quad D(\alpha, \beta)$$

# Local Splitting

Let  $P(X, Y) = (Y^2 + X + 1)(Y^2 + X)(Y^2 + X^2 + Y)(XY + 1)$  and  $(\alpha, \beta) = (0, 0)$

- $A(\alpha, \beta) = \{i \in [s] \mid P_i(\alpha, \beta) \neq 0, \deg(P_i(\alpha, Y)) \geq 1\}$   
So  $Y^2 + X + 1 \in A(0, 0)$
- $B(\alpha, \beta) = \{i \in [s] \mid P_i(\alpha, Y) = \gamma(Y - \beta)^m, m \geq 1, \gamma \neq 0\}$   
So  $Y^2 + X \in B(0, 0)$
- $C(\alpha, \beta) = \{i \in [s] \mid P_i(\alpha, Y) = (Y - \beta)^m \hat{P}_i(Y), m \geq 1, \hat{P}_i(\beta) \neq 0, \deg \hat{P}_i \geq 1\}$   
So  $Y^2 + X^2 + Y \in C(0, 0)$
- $D(\alpha, \beta) = \{i \in [s] \mid P_i(\alpha, \beta) \neq 0, \deg(P_i(\alpha, Y)) = 0\}$   
So  $XY + 1 \in D(0, 0)$

I will use  $A, B, C, D$  to denote these. Define  $P_A = \prod_{i \in A} P_i$  and similarly  $P_B, P_C, P_D$ .

**Observe:** If  $P$  is monic in  $Y$  then  $D$  is empty.

## A Nice Observation

$w = \{(\alpha_j, \beta_j)\}_{j \in [n]}$  denote the received word.

$Q$  is the polynomial from Interpolation step of Guruswami-Sudan algorithm. Let  $f$  is in the list.

For any  $j \in [n]$ :

If  $f(\alpha_j) = \beta_j$ : Then  $Y - f(X) \mid P_B$

If  $f(\alpha_j) \neq \beta_j$ : Then  $Y - f(X) \mid P_A$

# Derandomization

## Remark

We will assume  $Q$  is monic in  $Y$  and has no pure  $X$ -factors.

**If only we had:** An efficient  $\text{poly}(n, \log q)$  time algorithm `SPLIT` to find  $P_A, P_B$  from  $P, (\alpha, \beta)$  then:

## Algorithm:

1.  $S \leftarrow \{Q\}$
2. Choose  $j \in [n]$  and for all  $g \in S$  compute  $(g_A, g_B) = \text{SPLIT}(g, (\alpha_j, \beta_j))$
3. Remove  $g$  from  $S$  and put  $g_A, g_B$  in  $S$ .
4. Continue from step 2 till  $S$  stabilizes.
5. Do some interpolations to recover list

# Recover List from Stable Set

**Observe:** If  $f$  is in the list there is one factor  $g \in S$ ,  $Y - f(X) \mid g$ .

## Lemma

For all  $j \in [n]$ ,

$$g(\alpha_j, \beta_j) = 0 \iff f(\alpha_j) = \beta_j$$

So go over all  $g \in S$ , find  $j \in [n]$  such that  $g(\alpha_j, \beta_j) = 0$ , do interpolation to find appropriate  $f$ .

But stabilization can take long time !!

Simple potential function:

$$\Phi(S) = \sum_{i=1}^{\deg_Y(Q)} (i-1) \times \#(\text{polynomials with } Y\text{-deg} = i \text{ in } S)$$

You will notice  $\Phi(S)$  decreases by at least 1 in each update of  $S$ .

# Finally

Final Algorithm:

1.  $S \leftarrow \{Q\}$
2. Choose  $j \in [n]$  and for all  $g \in S$  compute  $(g_A, g_B) = \text{SPLIT}(g, (\alpha_j, \beta_j))$
3. Remove  $g$  from  $S$  and put  $g_A, g_B$  in  $S$ .
4. Continue from step 2 till  $S$  stabilizes.
5. Go over all  $g \in S$  and do interpolation on the set  $\{j \in [n] \mid g(\alpha_j, \beta_j) = 0\}$  and recover list

## Theorem

*There is a deterministic algorithm that, for every finite field  $\mathbb{F}$  and parameters  $n, k \in \mathbb{N}$  runs in time  $\text{poly}(n, \log |\mathbb{F}|)$  list decodes Reed Solomon code  $RS[n, k]$  from agreement more than  $\sqrt{n(k-1)}$ .*

# Splitting Algorithm

---

**Thank You**