通过部分高斯消元降低 ORBGRAND 时延

Li Wan、Wenyi Zhang 电子工程与信息科学系 中国科学技术大学 邮箱：wan li@mail.ustc.edu.cn, wenyizha@ustc.edu.cn

arXiv:2602.01174v1 [cs.IT] 1 Feb 2026

摘要—**猜测随机加性噪声译码（GRAND）是一种通用译码框架，通过测试候选错误图样（EP）来译码所有分组码。有序可靠性比特GRAND（ORBGRAND）利用对数似然比（LLR）排序，便于并行实现 GRAND，但在不利信道条件下仍存在较高的尾延迟，限制了其在实时系统中的应用。**

我们提出一种消元辅助 ORBGRAND 方案，将最可靠错误比特的秩（RMRE）与部分高斯消元（GE）筛选机制相结合，从而降低译码时延。该方案对具有相同 RMRE 的错误模式进行分组并联合验证；一旦识别出有效 EP，便恢复 ORBGRAND 搜索。通过利用先前的 GE 步骤筛除不必要的猜测，该方法显著减少了需要测试的错误模式数量，从而降低平均时延和最坏情况时延，同时保持纠错性能。

仿真结果表明，与原始 OR-BGRAND 相比，消元辅助 ORBGRAND 可过滤掉超过 50% 的错误模式，并相应降低总体计算复杂度，同时不会造成分组错误率损失。这表明，该方法适用于超可靠低时延通信场景。

I. 引言

近年来，人们提出了多种统称为猜测随机加性噪声译码（GRAND）的译码方法。这些方法通常被称为通用方法，因为它们可应用于所有分组码（例如 BCH、LDPC 和极化码）。其核心思想是从含噪信道输出中依次减去候选错误图样（错误模式），以确定是否恢复出有效码字。GRAND 特别适用于短码长高码率码 [1], ，因此成为超可靠低时延通信（URLLC） [2]的理想候选方案。

在 GRAND 的各种变体中，有序可靠性比特 GRAND（ORBGRAND）仅根据对数似然比（LLR）幅值的排序生成查询 [3], [4], ，从而便于并行处理和实现硬件友好型设计 [5], [6], [7]。因此，ORBGRAND 是实现低时延通信的有前景方案之一。然而，ORBGRAND 译码器仍容易受到时延不稳定性的影响：尽管平均译码时间通常较短，但在不利的信道条件下，最坏情况下的猜测次数可能会大幅增加，导致尾部时延可能无法满足实时系统的要求。

本工作得到中国国家自然科学基金项目编号62231022的部分资助。

关于加速 ORBGRAND 的已有研究大致可分为两类。第一类着重设计特定的预生成错误模式集合，以提升译码性能，这通常能同时减少平均猜测次数和整体复杂度 [3], [4], [8]。第二类旨在设计更快的验证方案，以降低译码延迟：有些方法利用校验矩阵的结构，提前排除某些错误模式 [9]，或加速基于伴随式的搜索 [10], ；另一些方法则通过部分评估校验方程，实时生成错误模式 [11]。

我们从最可靠错误位的秩（RMRE）这一角度提出一种新视角——它指错误模式中被翻转的最可靠比特的索引。RMRE 索引较小的错误模式在统计上更有可能出现，并主导译码的早期阶段。基于这一观察，我们将部分高斯消元（GE）融入 ORBGRAND 译码过程，以联合验证多个具有相同 RMRE 的错误模式，从而加速译码。

对于信息长度为 K 、码字长度为 N 的码，若要实现消元辅助 ORBGRAND，译码器会根据比特可靠性对校验矩阵 $H^{(N-K)\times N}$ 重新排序，然后从可靠性最低的比特开始，逐列进行部分高斯消元。类似的消元过程也常用于有序统计译码（OSD） [12], ，但在我们的方法中，消元经过优化，以降低计算复杂度。具体而言：

- 在每一列 k，译码器都会检查能否通过翻转前 k 个位置中的比特得到一个码字。这样可以联合验证所有 RMRE 为 $= k$ 的错误模式，从而筛除冗余猜测。
- 消元仅针对前 k 列，忽略其余无关列，以降低复杂度。

这两种操作将高斯消元的复杂度从$\mathcal{O}(N(N-K)^2)$ 降至 $\mathcal{O}((N-K)M^2)$，其中 M（通常为 $N-K$ 的 20–30%）表示部分高斯消元所需的列数，从而显著降低计算开销。一旦在高斯消元模块中识别出有效 EP，译码器便返回 ORBGRAND 序列并继续译码，同时安全地跳过所有 RMRE 小于或等于已确定 EP 的错误模式。这一策略显著减少了猜测总数，加快了译码速度。

仿真与复杂度分析表明，借助消元的 ORBGRAND 能够加快译码，同时不影响纠错性能。该方法对硬件友好，仅依赖逐比特异或操作；嵌入式部分高斯消元还使其能够与有序统计译码无缝集成，以应对少见且难以译码的情况。这些特性共同为基于 GRAND 的架构提供了可扩展且高效的框架。

本文其余部分安排如下：第二节介绍传输模型和 OR-BGRAND，第三节详述消元辅助的 ORB-GRAND 并进行复杂度分析，第四节给出仿真结果，以验证所提方法的有效性。最后，第五节总结全文。

II. 预备知识

A. 传输模型

我们考虑一种分组编码系统，其中信息向量 $U \in \mathbb{F}_2^K$ 被编码为码字 $W \in \mathcal{C} \subseteq \mathbb{F}_2^N$，码率为 $R = K/N$。码字通过反极性调制映射为发射信号向量 $X \in \{-1,+1\}^N$，其中当 $W_i = 0$ 时，$X_i = +1$；当 $W_i = 1$ 时，$X_i = -1$。所得加性高斯白噪声信道的输出向量满足 $Y \mid X \sim \mathcal{N}(X,\ \sigma^2 \mathbf{I}_{N\times N})$。给定 Y，对数似然比为

$$L_i = \log \frac{p(Y_i \mid W_i = 0)}{p(Y_i \mid W_i = 1)} = \frac{2}{\sigma^2} Y_i,\ i = 1, \ldots, N, \quad (1)$$

并且我们用 ℓ_i 表示 L_i 的实现。

我们将硬判决函数 $\theta(\cdot)$ 定义为：若 $y \geq 0$，则 $\theta(y) = 0$；否则为 1。对于 Y，为简洁起见，将 $[\theta(Y_1), \ldots, \theta(Y_n)] \in \mathbb{F}_2^N$ 组成的向量记为 $\theta(Y)$。

B. ORBGRAND

GRAND 由两个部分组成：错误模式生成器和码字校验器。给定最大查询预算 T，译码器生成错误模式序列 $\{e(1), \ldots, e(T)\}$，其中 $e(t) \in \mathbb{F}_2^N$，并依次检验 $\theta(y) \oplus e(t)$ 是否为有效码字。对于校验矩阵为 $H^{(N-K)\times N} = [h_1, \ldots, h_N]$ 的线性分组码，该检验可表示为

$$H \cdot \big(\theta(y) \oplus e(t)\big) = 0 \quad \Leftrightarrow \quad H \cdot e(t) = s\,, \quad (2)$$

其中，我们定义伴随式 $s \triangleq H \cdot \theta(y)$。如果所得 $\theta(y) \oplus e(t)$ 对应一个有效码字，则译码以该输出终止，并称此类错误模式为有效错误模式；否则，流程将继续，直到找到码字或预算 T 耗尽，此时译码被判定为失败。

ORBGRAND 是 GRAND 的一种变体，采用高效的错误模式生成策略。具体而言，它会预先计算并在本地存储一组错误模式，记为 $\mathcal{E}_{\text{ORB}} = \{\tilde{e}(1), \ldots, \tilde{e}(T)\}$[1]。对于每个接收对数似然比向量 ℓ，将其元素排序以获得排序向量 r，使得

$$|\ell_{r_1}| \leq |\ell_{r_2}| \leq \cdots \leq |\ell_{r_N}|. \quad (3)$$

[1]我们用 e 表示实际猜测的错误模式，而 $\tilde{e}$ 表示 ORBGRAND 中本地预存的错误模式。二者通过式 (4) 中的变换相关联。

每个预存的错误模式 $\tilde{e}(t)$ 在测试前都会根据 r 进行置换，从而得到

$$e(t) = \mathcal{P} \cdot \tilde{e}(t), \quad (4)$$

其中，置换矩阵 $\mathcal{P} \in \{0,1\}^{N\times N}$ 定义为 $\mathcal{P}_{i,j} = \mathbf{1}(i = r_j)$。因此，式 (2) 中的校验可重写为

$$H \cdot (\mathcal{P} \cdot \tilde{e}(t)) = s. \quad (5)$$

将 H 和 $\mathcal{P}$ 分组，定义 $H_\pi \triangleq H \cdot \mathcal{P}$，得到

$$H_\pi = \left[h_{r_1}, \ldots, h_{r_N}\right], \quad \text{Eq. (5)} \ \Leftrightarrow \ H_\pi \cdot \tilde{e}(t) = s. \quad (6)$$

为清晰起见，ORBGRAND 的整体译码流程总结于**算法 1**中。

算法1： ORBGRAND [3]

输入： y, T, H，预存的 $\mathcal{E}_{\text{ORB}}$。
输出： 译码结果 $\hat{w}$。

$\mathcal{P} \leftarrow$ 排序 y，初始化 $t = 0$, $flag = 0$;
$s \leftarrow H \cdot \theta(y)$　$H_\pi = H \cdot \mathcal{P}$;
当 $flag = 0$ 且 $t < T$ **执行**
　$t = t + 1$;
　$\tilde{e}(t) \leftarrow \mathcal{E}_{\text{ORB}}[t]$;
　标志 $\leftarrow \mathbf{1}(H_\pi \cdot \tilde{e}(t) = s)$;　// 标志 = 1如果成功
结束

$$\textbf{return } \hat{w} = \begin{cases} \theta(y) \oplus (\mathcal{P} \cdot \tilde{e}(t)) & \text{if } \textit{flag} = 1, \\ \emptyset & \text{if } \textit{flag} = 0. \end{cases}$$

C. 最可靠错误位的秩

在这项工作中，我们关注码字中可靠性最高的错误比特。为了形式化这一概念，我们首先定义错误模式（EP）的最可靠错误位的秩（RMRE）。

定义 *1*：令 $e \in \mathbb{F}_2^n$ 表示一个错误模式，并令 r 为式 (3) 中的可靠性排序向量。e 的 RMRE 定义为

$$\text{RMRE}(e) = \max\{i \mid e_{r_i} = 1\}. \quad (7)$$

特别是，能够导致成功译码的唯一错误模式的 RMRE 可表示为

$$\begin{aligned} \text{RMRE}(\theta(y) \oplus w) &= \max\{i \mid \theta(y_{r_i}) \oplus w_{r_i} = 1\} \\ &= \max\{i \mid \theta(y_{r_i}) \neq w_{r_i}\}. \end{aligned} \quad (8)$$

对于 ORBGRAND，由于每个被测试的错误模式都写作 $e = \mathcal{P} \cdot \tilde{e}$，根据 $\mathcal{P}$ 的定义，可得

$$\text{RMRE}(\mathcal{P} \cdot \tilde{e}) = \max\{i \mid \tilde{e}_i = 1\}.$$

III. 主要结果

本节介绍本文的主要结果。第三节 A 详细描述算法流程，第三节 B 对复杂度进行分析，并提出进一步优化算法的方法。第三节 C 进一步分析 RMRE，以帮助理解所提算法的底层机制。

A. 消元辅助 *ORBGRAND* 算法

所提消元辅助 ORBGRAND 算法旨在通过执行部分高斯消元，求解 $H_\pi \tilde{e} = s$，从而更高效地运行 ORBGRAND。整体流程如下。由于涉及迭代矩阵变换，我们使用上标（例如 $H_\pi^{(k)}$）表示第 k 轮后的结果。

*a)*步骤***1.*** 初始化：构造增广矩阵 $[H_\pi^{(0)} \mid \underline{s}^{(0)}] \in \mathbb{F}^{(N-K)\times(N+1)}$。将候选阈值初始化为 $t^* = T$，表示错误模式的最大索引。设置列索引 $n = 1$ 和主元索引 $k = 1$。

*b)*步骤***2.*** 按列消元：对于第 n 列，在 $k, \ldots, N-K$ 中搜索主元。

情况 I： 如果找到非零元素 $h_{pn}^{(k-1)} (k \le p \le N-K)$，交换第 p 行和第 k 行，以在 (k,n) 处建立主元，并对所有其他行执行消元：

$$\forall\, i \neq k \text{ and } \forall\, j: \quad m_{in}^{(k-1)} = \frac{h_{in}^{(k-1)}}{h_{kn}^{(k-1)}},$$

$$h_{ij}^{(k)} \leftarrow h_{ij}^{(k-1)} - m_{in} h_{kj}^{(k-1)}, s_i^{(k)} \leftarrow s_i^{(k-1)} - m_{in}^{(k-1)} s_k^{(k-1)}.$$

由于 $H_\pi^{(k-1)}$ 是二进制且 $h_{kn}^{(k-1)} = 1$，乘数可简化为 $m_{in}^{(k-1)} = h_{in}^{(k-1)}$，得到

$$h_{ij}^{(k)} \leftarrow h_{ij}^{(k-1)} \oplus h_{in}^{(k-1)} h_{kj}^{(k-1)},\ s_i^{(k)} \leftarrow s_i^{(k-1)} \oplus h_{in}^{(k-1)} s_k^{(k-1)}. \tag{9}$$

消元后，更新 $k \leftarrow k+1$，并继续执行步骤***3***进行一致性检查。

情况 II： 如果没有找到主元，则更新 $n \leftarrow n+1$，并继续执行步骤***2***。

*c)*步骤***3***：约化系统验证：每当找到主元时，检查前 n 列中是否存在解。这等价于验证

$$\operatorname{rank}\Big([\underline{h}_1^{(k)}, \ldots, \underline{h}_n^{(k)}]\Big) = \operatorname{rank}\Big([\underline{h}_1^{(k)}, \ldots, \underline{h}_n^{(k)}, \underline{s}^{(k)}]\Big), \tag{10}$$

这表明存在一个 $\tilde{e}$，它只翻转前 n 个比特并满足 $H_\pi^{(0)} \cdot \tilde{e} = s^{(0)}$。

如果式 (10) 成立，则求解约化系统，以获得所有 RMRE 为 $= n$ 的有效错误模式，记为 $\mathcal{E}_n$，然后继续执行步骤 ***4***；否则，更新 $n \leftarrow n+1$ 并返回步骤***2***。

d) 步骤 ***4***：最终解搜索：回到 ORB-GRAND 算法，我们已确立：

- 仅翻转前 $n-1$ 个比特无法得到有效码字。
- RMRE 为 $= n$ 的有效 EP 已经识别，其在 $\mathcal{E}_{\text{ORB}}$ 中的位置可通过哈希查找获取。如果 $\mathcal{E}_n \cap \mathcal{E}_{\text{ORB}} \neq \emptyset$，则可得到一个解：

$$t^* = \min\{t \mid \tilde{\underline{e}}(t) \in \mathcal{E}_n \cap \mathcal{E}_{\text{ORB}}\}. \tag{11}$$

否则，仍保留 $t^* = T$。

因此，只需通过检查进一步考察 RMRE 为 $> n$ 的 EP：

$$H_\pi^{(0)} \cdot \tilde{\underline{e}}(t) = \underline{s}^{(0)}. \tag{12}$$

由于错误模式已预先生成，因此可以提前存储其 RMRE 值，避免在译码过程中重复计算。

总而言之，所提算法以按列递增的方式执行高斯消元，并在每个阶段检查一致性。在最后阶段，它会筛除不可行的候选 EP，然后再将其传递给 ORBGRAND 搜索。完整的伪代码见 **算法 2**。

算法 2： 消元辅助 ORBGRAND

输入： $\underline{y}, T, H$，预存的 $\mathcal{E}_{\text{ORB}}$。
输出： 译码结果 $\hat{w}$。

利用 (2)-(6) 构造 $[H_\pi \mid \underline{s}]$，初始化 $k \leftarrow 1, t^* \leftarrow T$;
对于 $n \leftarrow 1$ **到** N，**执行**
  找到一行 $p \in \{k, \ldots, N-K\}$，使得 $h_{pn} = 1$;
  如果存在这样的 p **则**
    交换行 p 和 k（在 (k,n) 处选取主元）;
    对于 $i \neq k$ 和 $\forall\, j$，**执行**
      $h_{ij}^{(k)} \leftarrow h_{ij}^{(k-1)} \oplus h_{in}^{(k-1)} h_{kj}^{(k-1)}$;
      $s_i^{(k)} \leftarrow s_i^{(k-1)} \oplus h_{in}^{(k-1)} s_k^{(k-1)}$;
    end
    如果(10) 成立**则**
      $\mathcal{E}_n \leftarrow$ 使用前 n 列的有效 EP;
      如果 $\mathcal{E}_n \cap \mathcal{E}_{ORB} \neq \emptyset$ **则**
        $t^* \leftarrow \min\{t \mid \tilde{\underline{e}}(t) \in \mathcal{E}_n \cap \mathcal{E}_{\text{ORB}}\}$;
      end
      跳出;
    end
    $k \leftarrow k+1$;
  end
结束
对于 $t \leftarrow 1$ **到** t^*-1，**执行**
  如果 $RMRE(\tilde{e}(t)) > n$ 且 $H_\pi \cdot \tilde{\underline{e}}(t) = \underline{s}$ **则**
    返回 $\theta(\underline{y}) \oplus (\mathcal{P} \cdot \tilde{e}(t))$;
  end
结束
返回 $\theta(\underline{y}) \oplus (\mathcal{P} \cdot \tilde{e}(\underline{t}^*))$;

例 ***1***：为清晰地演示所提出的算法，我们以 (7,4) 汉明码为例。假设在接收到 ℓ 之后，我们计算出伴随式 $s = (0,0,1)^{\mathsf{T}}$ 以及排序后的校验矩阵如下：

$$H_\pi = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 1 \end{bmatrix}.$$

则相应的增广矩阵为

$$[H_\pi^{(0)} \mid \underline{s}^{(0)}] = \left[\begin{array}{ccccccc|c} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 \end{array}\right]. \tag{13}$$

然后我们继续执行步骤***2*** 和 步骤***3***，并发现当 $(n,k) = (3,3)$ 时，我们可以完成消元并满足 (10)。此时的增广矩阵如下：

$$[H_\pi^{(3)} \mid \underline{s}^{(3)}] = \left[\begin{array}{ccccccc|c} 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 \end{array}\right]. \tag{14}$$

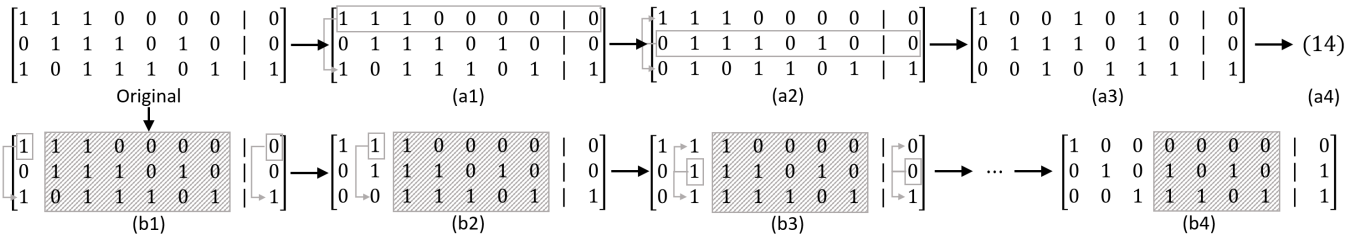


图 1. 完全消元与简化GE方案的比较。(a1)–(a4)：完全消元，其中每一步都会更新所有剩余列。(b1)–(b4)：所提出的约化消元——(b1)仅处理第一列以及s。(b2)–(b3)应用先前变换（$I + Q^{(1)}$），然后消去第二列；(b4)展示了所得的约化GE，该变换仅作用于前三列和 s，而所有后续列保持不变。

根据(14)，由于目标综合为 $s^{(3)} = (0,\ 1,\ 1)^{\mathsf{T}}$，因此在前三列上有支撑的相应解为$\mathcal{E}_3 = \{(0,\ 1,\ 1,\ 0,\ 0,\ 0,\ 0)^{\mathsf{T}}\}$。

优先级 t	错误模式 $\tilde{e}(t)^{\mathsf{T}}$	RMRE	状态
1	(1, 0, 0, 0, 0, 0, 0)	1	已过滤
2	(0, 1, 0, 0, 0, 0, 0)	2	已过滤
3	(0, 0, 1, 0, 0, 0, 0)	3	已过滤
4	(1, 1, 0, 0, 0, 0, 0)	2	已筛除
5	(0, 0, 0, 1, 0, 0, 0)	4	已验证
6	(1,0,1,0,0,0,0)	3	已筛除
7	(0,0,0,0,1,0,0)	5	已验证（输出）
8	(1, 0, 0, 1, 0, 0, 0)	4	待定
9	(0, 1, 1, 0, 0, 0, 0)	3	由高斯消元确定

表 I：采用更新后的状态描述的(7, 4) 汉明码错误模式筛选过程。

然后我们进入步骤 ***4***。筛选过程总结于表 I，其中前三列已预先存储在本地。由高斯消元确定的错误模式为 $\tilde{e}$ (9) =(0,1,1,0,0,0,0)$^{\mathsf{T}}$。然后，我们依次检查候选 EP：具有 RMRE ≤ 3 的错误模式被滤除，因此只需验证 $\tilde{e}$(5)和 $\tilde{e}$(7)。 $\tilde{e}$(7)可被判定为有效 EP，因此我们将其输出，并确认最终解码码字为 $\theta(y) \oplus \mathcal{P} \cdot \tilde{e}$(7)。在本例中，猜测次数从 7 减少到 2（仅$\tilde{e}$(5)和 $\tilde{e}$(7)）。

B. 复杂度分析

所提算法可分为两部分：部分高斯消元和过滤后的ORBGRAND。本小节分析其计算复杂度，并提出一种进一步降低高斯消元过程开销的优化设计。

我们首先考虑**算法 2**中的高斯消元复杂度。式(9)中每一消元轮次大约需要$(N+1)(N-K)$次运算。由于仅执行部分消元，当完成 k 轮消元后，我们记录结束轮次 $m = k$，最大复杂度为

$$\mathcal{O}_{\text{Partial GE}} = m\,(N+1)(N-K). \tag{15}$$

然而，只有前 m比特与构造有效 EP 相关。一旦找到可行的 RMRE，对后续列的消元便成为冗余。因此，我们将上一小节中**算法 2**的步骤***2***简化如下。

1) **第一个主元：** $(n,\ k) = (1,\ 1)$，在 $h_1^{(0)}$ 中找到一个非零主元 $h_{p1}^{(0)}$。将其交换到第一行后，记录行变换矩阵 $(I + Q^{(1)})$，其中 $Q_{i1}^{(1)} = h_{i1}^{(0)}$ 及其他位置均设为零。该操作仅作用于第一列和伴随式向量：

$$h_{i1}^{(1)} = 0, \quad s_i^{(1)} = s_i^{(0)} \oplus h_{i1}^{(0)} s_1^{(0)}, \quad \forall\ i \neq 1. \tag{16}$$

然后设置 $k \leftarrow k+1$ 并进入步骤***3***。

2）**后续主元：**在处理第 n列之前，应用累积的行变换：

$$\underline{h}_n^{(k-1)} \leftarrow \left(\prod_{j=k-1}^{1} (I + Q^{(j)}) \right) \underline{h}_n^{(0)}. \tag{17}$$

随后对更新后的列 $h_n^{(k-1)}$ 执行高斯消元：如果找到主元，则按照式 (16) 对 $h_n^{(k-1)}$和$s^{(k-1)}$ 执行消元，存储新的变换矩阵 $(I+Q^{(k)})$ $k \leftarrow k+1$ 并继续执行步骤***3***；否则，转到 $(n+1,\ k)$。

例 *2*：我们通过图 1 展示矩阵 (13) 上的原始和简化部分高斯消元过程，该矩阵见例 *1*。

在完整高斯消元方法（图 1(a1)–(a4)）中，每轮都会更新所有列。相比之下，所提出的方案（图 1(b1)–(b4)）仅更新活动列和s。在处理下一列之前，会将累积的变换应用一次，从而避免冗余操作，同时确保主元正确形成。

现在我们估算这一简化过程的复杂度。第一轮中，式 (16) 的计算开销至多为 $2(N-K)$。对于第 k 个主元，式 (17) 中的累积变换与消元合计需要

$$(k-1)(N-K) + 2(N-K) = (k+1)(N-K) \text{ operations.}$$

如果总共找到 m 个主元，则总体复杂度变为

$$\mathcal{O}_{\text{Reduced GE}} = \sum_{k=1}^{m} [(k+1)(N-K)] = \frac{m(m+3)}{2}(N-K).$$

最后，我们总结整体译码复杂度。令T_1 表示原始 OR-BGRAND 所需的猜测次数，每次试验的验证成本为 cN，其中c表示验证过程中使用的奇偶校验方程平均数量。对于所提出的消元辅助方案，所需的猜测次数记为 T_2，其中 $T_2 \leq T_1$ 是由于筛选机制所致。考虑到

部分高斯消元阶段的额外成本，总译码复杂度可表示为

$$\mathcal{O}_{\text{ORBGRAND}} = T_1 cN, \tag{18}$$

$$\mathcal{O}_{\text{With partial GE}} = \mathbb{E}[M]\,(N+1)(N-K) + T_2 cN, \tag{19}$$

$$\mathcal{O}_{\text{With reduced GE}} = \mathbb{E}\left[\frac{M(M+3)}{2}\right](N-K) + T_2 cN. \tag{20}$$

C. 讨论

由式 (18)–(20) 可知，所提算法的有效性取决于消元带来的额外开销与通过滤除部分错误模式所实现的复杂度降低之间的权衡。

为进一步理解这种权衡，我们通过 RMRE($\theta(Y)\oplus W$) 分析 M 的统计特性。由于对于每个接收向量，我们有

$$m \le \min\{\text{RMRE}(\underline{e}) \mid H_\pi \cdot \underline{e} = \underline{s}\}, \tag{21}$$

并且 $\theta(y)_\oplus w_$是式 (8) 中的目标错误模式，因此我们有 $m \le$RMRE($\theta(y) \oplus w$)，且对于任意单调递增函数 $g(\cdot)$:

$$\mathbb{E}[g(M)] \le \mathbb{E}[g(\text{RMRE}(\theta(\underline{Y}) \oplus \underline{W}))].$$

随后可利用次序统计量理论来刻画 RMRE($\theta(Y) \oplus W$) 的分布，推导见 [13]。具体而言，令 $P_i = \Pr\left(\text{RMRE}(\theta(Y)\oplus W) = i\right)$。那么在对称信道下，有 $P_0 = (1 - F(0))^N$，且

$$P_i = \frac{N!}{(i-1)!(N-i)!} \int_0^\infty \left[F(t) - F(-t)\right]^{i-1} \cdot f(-t)\left[1 - F(t)\right]^{N-i} \mathrm{d}t. \tag{22}$$

$F(\cdot)$和 $f(\cdot)$分别是 $X = +1$时 L 的累积分布函数和概率密度函数。

因此，当信道统计特性使 RMRE($\theta(Y) \oplus W$) 集中在较小的秩上时，期望值 $\mathbb{E}[M]$ 远小于码长 N。因此，在我们的设置中，与奇偶校验操作相比，部分高斯消元的额外开销仍然很小。

IV. 性能评估

本节使用 BCH(127,113)码给出仿真结果，以验证所提算法的有效性。最大猜测次数设为 $T = 5 \times 10^4$，评估的主要译码方法为 ORBGRAND[3]及其近期改进变体 RS-ORBGRAND [8]。

图 2 比较了多种译码器在 BCH (127, 113) 码上的译码性能，包括 Berlekamp-Massey (BM) 译码 [14]，OSD，以及多种 GRAND 变体。其中，软输入GRAND (SGRAND) [15] 实现了最大似然解码 (MLD)[2]。我们分别将加速算法应用于 ORBGRAND 和 RS-ORBGRAND。结果表明，消元辅助的 ORBGRAND 达到与.....相同的分组错误率

[2]当允许解码器枚举所有错误模式，且不对最大猜测次数设限时，即可实现最大似然解码。

原始 ORBGRAND 算法。此外，尽管仅依赖软信息的排序值，RS-ORBGRAND 的性能仍非常接近最大似然解码，并且几乎达到一阶有序统计译码的相同性能。

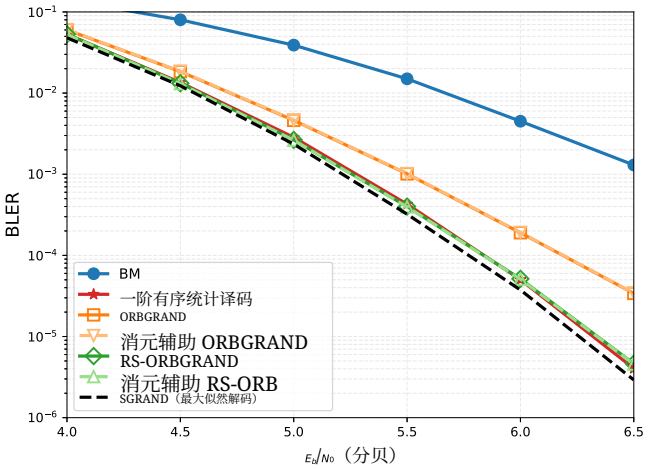


图 2. 不同 E_b/N_0下块错误率曲线的比较。

表 II 总结了应用加速算法后得到的平均猜测次数，结果表明，对于不同的 E_b/N_0 值，所提方法始终能将猜测次数减少约 40%–50%。图3进一步展示了猜测次数的分布。可以看出，所提方法有效地将猜测次数集中在较小的数值范围内，从而显著降低平均猜测次数，并缓解高尾延迟问题。

译码方法	每个 E_b/N_0 的平均猜测次数		
	4 dB	**5 dB**	**6 dB**
ORBGRAND	1.04e+3	9.67e+1	7.30e+0
消元辅助 ORBGRAND	5.92e+2	4.49e+1	3.27e+0
猜测次数减少（% ↓）	**43.1**	**53.5**	**55.2**
RS-ORB	9.52e+2	7.02e+1	4.52e+0
消元辅助 RS-ORB	6.09e+2	4.16e+1	2.40e+0
猜测次数减少（% ↓）	**36.0**	**40.7**	**46.9**

表 II 不同译码方法的平均猜测次数比较

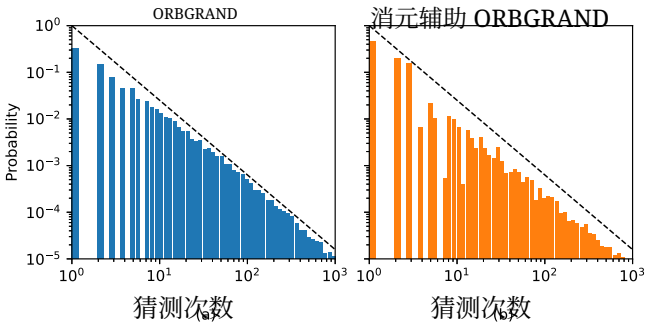


图3. BCH(127, 113) 的猜测次数直方图，E_b/N_0 = 5 分贝。

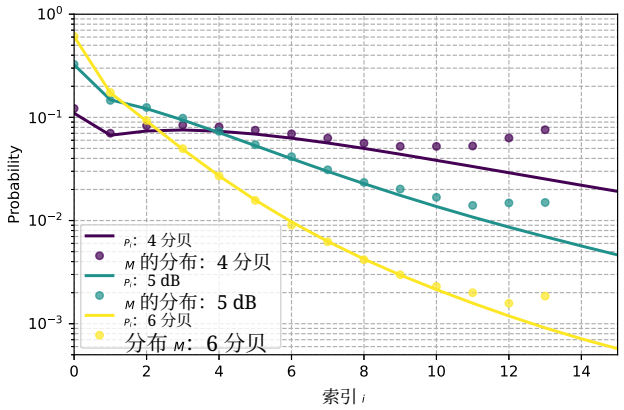


图4. RMRE($\theta(Y) \oplus W$)和 M的分布。

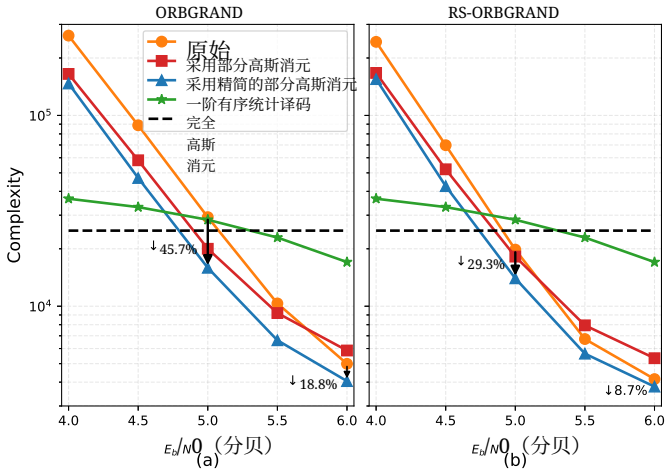


图5. 不同E_b/N_0 取值下的平均计算复杂度比较（1次浮点运算 = 8次异或）。

关于部分高斯消元的复杂度分析，我们首先验证第三节-C中给出的结果。图4分别展示了 RMRE($\theta(Y) \oplus W$) 和 M 的分布。RMRE 值由式 (22) 得出，而M 则通过对重复解码试验进行蒙特卡罗抽样来估计。结果表明，大多数 RMRE 值都相对较小，这说明在所考虑的设置下，成功译码通常只需要进行少量比特翻转。

图5(a)和(b)比较了不同实现下 ORBGRAND 和 RS-ORBGRAND 的译码复杂度。纵轴表示异或运算的平均次数，其中一次浮点运算按八次异或运算计数。尽管部分高斯消元可以在较低的 E_b/N_0下降低复杂度，但在较高的 E_b/N_0下，其收益会减弱，甚至可能消失，因为每个消元步骤仍然作用于完整行。相比之下，简化的部分高斯消元方法进一步降低了消元复杂度，从而显著提升整体性能。与此同时，完整高斯消元仍然开销较大，限制了随着 E_b/N_0增加而降低有序统计译码复杂度的效果。因此，在高E_b/N_0 区间，ORBGRAND 仍保持明显的复杂度优势。

在图5的 $E_b/N_0 = 5$ 分贝处，猜测次数减少53.5%和40.7%（见表 II），分别使整体译码复杂度降低45.7%和29.3%。即使在6分贝处，由于猜测次数极少，排序在总开销中占比更大，我们仍观察到18.1%和8.7%的加速，这证实了通过简化的部分高斯消元移除冗余错误模式，可显著降低端到端时延。

V. 结论

我们提出了一种消元辅助 ORBGRAND 方案，利用 RMRE 加速译码。通过集成部分高斯消元，可联合验证多个具有相同RMRE的EP，从而过滤掉冗余的EP检查，并将高斯消元的复杂度从 $\mathcal{O}(N(N-K)^2)$ 降低至 $\mathcal{O}((N-K)M^2)$。仿真结果表明，该方法在保持相同纠错能力的同时，显著加快了译码速度。该方法对硬件友好，并可无缝集成有序统计译码，以处理罕见且难以译码的情况，为基于 GRAND 的架构提供可扩展的框架。

参考文献

[1] K. R. Duffy, J. Li, and M. M´edard, “容量逼近猜测随机加性噪声译码”，*IEEE*信息论汇刊, 第65卷，第7期，第4023–4040页，2019年。[2] C. Yue, V. Miloslavskaya, M. Shirvanimoghaddam, B. Vucetic, and Y. Li, “面向6G超可靠低时延通信的短码长码高效译码器”，*IEEE*通信杂志, 第61卷，第4期，第84–90页，2023年。[3] K. R. Duffy, W. An, and M. M´edard, “有序可靠位猜测随机加性噪声译码”，*IEEE*信号处理汇刊, 第70卷，第4528–4542页，2022年。[4] M. Liu, Y. Wei, Z. Chen, and W. Zhang, “ORBGRAND几乎达到容量”，*IEEE*信息论汇刊, 第69卷，第5期，第2830–2840页，2022年。[5] S. M. Abbas, T. Tonnellier, F. Ercan, M. Jalaleddine, and W. J. Gross, “用于有序可靠性比特GRAND的高吞吐量、节能型超大规模集成电路架构”，*IEEE*超大规模集成电路系统汇刊, 第30卷，第6期，第681–693页，2022年。[6] C. Ji, X. You, C. Zhang, and C. Studer, “采用并行噪声序列生成的高效ORBGRAND实现”，*IEEE*超大规模集成电路系统汇刊, 2024年。[7] L. Wan, H. Yin, and W. Zhang, “利用错误模式树表示实现具有最优性保证的GRAND并行化策略”，*arXiv*预印本 *arXiv:2510.01813*, 2026年。[8] L. Wan and W. Zhang, “通过重排ORBGRAND逼近最大似然解码性能”，载于*IEEE*国际信息论研讨会（*ISIT*）, 2024年，第31–36页。[9] M. Rowshan and J. Yuan, “面向GRAND的受约束错误模式生成”，载于*IEEE*国际信息论研讨会（*ISIT*）, 2022年，第1767–1772页。[10] R. Hadavian, D. Truhachev, K. El-Sankary, H. Ebrahimzad, and H. Na-jafi, “有序可靠性直接错误模式测试（ORDEPT）算法”，载于*IEEE*全球通信大会, 2023年，第6983–6988页。[11] Y. Wang, J. Liang, and X. Ma, “线性分组码的部分受约束GRAND”，载于第九届国际计算机与通信系统会议（*ICCCS*）, 2024年，第617–622页。[12] M. P. Fossorier and S. Lin, “基于有序统计的线性分组码软判决译码”，*IEEE*信息论汇刊, 第41卷，第5期，第1379–1396页，1995年。[13] I. Song, S. R. Park, W. Zhang, and S. Lee, 《序统计与秩统计基础》。施普林格, 2024年。[14] E. R. Berlekamp, 《代数编码理论》。世界科学出版社, 2015年。[15] A. Solomon, K. R. Duffy, and M. M´edard, “使用GRAND进行软最大似然译码”，载于*IEEE*国际通信会议（*ICC*）, 2020年，第1–6页。