Decoding Random Linear Codes in $\tilde{\mathcal{O}}(2^{0.054n})$

Alexander May[⋆], Alexander Meurer[⋆⋆], and Enrico Thomae[⋆⋆⋆]

Faculty of Mathematics
Horst Görtz Institute for IT-Security
Ruhr-University Bochum, Germany
{alex.may,alexander.meurer,enrico.thomae}@rub.de

Abstract. Decoding random linear codes is a fundamental problem in complexity theory and lies at the heart of almost all code-based cryptography. The best attacks on the most prominent code-based cryptosystems such as McEliece directly use decoding algorithms for linear codes. The asymptotically best decoding algorithm for random linear codes of length n was for a long time Stern's variant of information-set decoding running in time $\tilde{\mathcal{O}}\left(2^{0.05563n}\right)$. Recently, Bernstein, Lange and Peters proposed a new technique called *Ball-collision decoding* which offers a speed-up over Stern's algorithm by improving the running time to $\tilde{\mathcal{O}}\left(2^{0.05558n}\right)$.

In this paper, we present a new algorithm for decoding linear codes that is inspired by a representation technique due to Howgrave-Graham and Joux in the context of subset sum algorithms. Our decoding algorithm offers a rigorous complexity analysis for random linear codes and brings the time complexity down to $\tilde{\mathcal{O}}\left(2^{0.05363n}\right)$.

Keywords: Information set decoding, representation technique.

1 Introduction

Linear codes have various applications in information theory and in cryptography. Many problems for random linear codes such as the so-called *syndrome decoding* are known to be NP-hard [2] and thus coding-based cryptography hopes to transfer this hardness to an average case hardness for cryptographic constructions. Since it is unlikely that hard coding problems are efficiently solvable on quantum computers, coding-based constructions are also one of the most prominent candidates for quantum-resistant cryptography.

Even many of today's lattice-based constructions like Regev's cryptosystem [12] or the HB protocol [7] inherently rely on the hardness of syndrome decoding via a variant called Learning Parity with Noise (LPN) problem. Given the importance of the syndrome decoding problem, it is a major task to understand its complexity in order to properly define cryptographic parameters that offer a

[⋆] Supported by DFG project MA 2536/7-1 and by ICT-2007-216676 ECRYPT II.

[⋆⋆] Ruhr-University Research School Germany Excellence Initiative [DFG GSC 98/1].

[⋆⋆⋆] and by DFG through an Emmy Noether grant.

D.H. Lee and X. Wang (Eds.): ASIACRYPT 2011, LNCS 7073, pp. 107–124, 2011.

sufficient security level. Let us introduce some notion that helps to investigate the syndrome decoding problem for linear codes.

A binary linear $[n, k, d]$-code $\mathcal{C}$ of length n is a linear subspace of the vector space $\mathbb{F}_2^n$. The dimension k of $\mathcal{C}$ is the dimension of the subspace. The distance d of $\mathcal{C}$ is defined as the minimal Hamming distance between two codewords.

An $[n, k, d]$-code $\mathcal{C}$ can be defined via some basis matrix $\mathbf{G} \in \mathbb{F}_2^{k \times n}$ for the subspace, called a generator matrix, i.e. $\mathcal{C} = \{\mathbf{xG} : \mathbf{x} \in \mathbb{F}_2^k\}$. Alternatively, we can define $\mathcal{C}$ via a parity check matrix $\mathbf{H} \in \mathbb{F}_2^{(n-k)\times n}$ whose kernel equals $\mathcal{C}$, i.e. we have $\mathcal{C} = \{\mathbf{x} \in \mathbb{F}_2^n : \mathbf{Hx}^t = \mathbf{0}\}$. Moreover, let $\mathcal{C}$ have distance d and let $\mathbf{c} \in \mathcal{C}$ be a codeword. Assume that we transmit $\mathbf{x} = \mathbf{c}+\mathbf{e}$ for some error vector with Hamming weight $w := \text{wt}(\mathbf{e}) \leq \lfloor \frac{d-1}{2} \rfloor$. Then $\mathbf{c}$ is the unique closest codeword in $\mathcal{C}$ to $\mathbf{x}$.

The term $\mathbf{s}(\mathbf{x}) := \mathbf{Hx}^t = \mathbf{H}(\mathbf{c}^t + \mathbf{e}^t) = \mathbf{He}^t$ is called the *syndrome* of $\mathbf{x}$. Notice that $\mathbf{e}$ defines the unique linear combination of exactly w columns of $\mathbf{H}$ that sum to $\mathbf{He}^t$ over $\mathbb{F}_2^n$. Finding this linear combination allows to recover the closest codeword $\mathbf{c} = \mathbf{x} + \mathbf{e}$. Hence, the so-called *syndrome decoding* of linear codes amounts to finding a subset I of ω out of n vectors from $\mathbb{F}_2^{n-k}$ such that the vectors in I sum to a fixed target value $\mathbf{s}(\mathbf{x})$.

A naive linear decoding algorithm is thus to search over all $\binom{n}{w}$ linear combinations of columns in $\mathbf{H}$. Obviously $w < \frac{n}{2}$, therefore the search space $\binom{n}{w}$ is maximal for w as large as possible. Thus, in coding based cryptosystems like McEliece [11] one usually fixes the weight of the error vector $\mathbf{e}$ to $w := \lfloor \frac{d-1}{2} \rfloor$. Throughout the paper, we assume for simplicity that we know w. We would like to stress that our decoding algorithm also works with the same asymptotical running time for unknown w, if we incorporate a loop over all possible values of w within the interval $(0, \lfloor \frac{d-1}{2} \rfloor]$, since our asymptotical running time is dominated by the largest value of w.

The running time of a decoding algorithm is a function of the three code parameters $[n, k, d]$. A random $[n, k, d]$-code is defined via a random parity check matrix $\mathbf{H} \in_R \mathbb{F}_2^{(n-k)\times n}$. It is well-known that for sufficiently large n random linear codes reach the so-called Gilbert-Varshamov bound (see [6], Chapter 2 for an introduction). More precisely, the code rate $\frac{k}{n}$ of a random linear code asymptotically reaches $1-H(\frac{d}{n})$, where H is the binary entropy function. Solving for d allows us to express the asymptotical running time for random linear codes as a function of $[n, k]$ only. We obtain a worst case running time as a function of n if we take the maximum over all values of $0 \leq k \leq n$. For all decoding algorithms in this work the worst case appears for codes of rate $\frac{k}{n} \approx 0.47$.

Related Work. Let $\mathbf{s}(\mathbf{x}) = \mathbf{Hx}^t$ be the syndrome of some erroneous codeword $\mathbf{x} = \mathbf{c} + \mathbf{e}$ with $\mathbf{c} \in \mathcal{C}$ and weight-w error $\mathbf{e}$. We briefly show how to extract $\mathbf{e}$ from $\mathbf{s}(\mathbf{x})$ by an algorithm called *information set decoding*, that was already mentioned in the initial security analysis of McEliece [11] and further explored by Lee and Brickell [9].

The idea of information set decoding is to reduce the search space by linear algebra. The first step is to randomly permute the columns of $\mathbf{H}$, which basically permutes the coordinates of the error vector $\mathbf{e}$. Then, one transforms the permuted $\mathbf{H} \in \mathbb{F}_2^{(n-k)\times n}$ into systematic form $(\mathbf{Q}|\mathbf{I}_{n-k})$ with $\mathbf{Q} \in \mathbb{F}_2^{(n-k)\times k}$ and $\mathbf{I}_{n-k}$ the $(n-k)$-dimensional identity matrix. Next, one fixes a weight p and computes for all linear combinations of p columns in $\mathbf{Q}$ the sum with the given syndrome $\mathbf{s}(\mathbf{x})$. If this sum has Hamming weight exactly $w-p$, then we can simply choose another $w-p$ columns from the identity matrix $\mathbf{I}_{n-k}$ in order to obtain a weight-w linear combination of columns that sum to $\mathbf{s}(\mathbf{x})$.

Obviously, information set decoding succeeds if we permute $\mathbf{e}$ such that exactly p out of its w 1-entries are in the first k coordinates, and the remaining $(w-p)$ 1-entries fall into the last $n-k$ coordinates. Optimization of p yields a running time of $\tilde{\mathcal{O}}\left(2^{0.05751n}\right)$.

In 1988, Leon [10] and Stern [13] further improved information set decoding by enforcing a window of 0-entries of size ℓ in the last $n-k$ entries of $\mathbf{e}$. Assume that this length-ℓ window is e.g. in positions $k+1,\ldots,k+\ell$ of $\mathbf{e}$. Then the weight-p linear combination of $\mathbf{Q}$ has to exactly match the syndrome $\mathbf{s}(\mathbf{x})$ in the ℓ positions $1,\ldots,\ell$, since we are no longer allowed to use the first ℓ columns from $\mathbf{I}_{n-k}$. Stern [13] proposed to compute those weight-p linear combinations of $\mathbf{Q}$ by a birthday technique via the sum of two disjoint weight-$\frac{p}{2}$ sums of columns in $\mathbf{Q}$. This algorithm lowers the time complexity to $\tilde{\mathcal{O}}\left(2^{0.05563n}\right)$ by increasing the memory complexity to $\tilde{\mathcal{O}}\left(2^{0.013n}\right)$.

In this work, we study a variant of Stern's information set decoding algorithm which is an instantiation of an algorithm by Finiasz and Sendrier from 2009 [5]. We call this instantiation FS-ISD. In FS-ISD, the 0-window is removed by simply removing the corresponding ℓ columns, i.e., by adjusting the systematic form to $\left(\mathbf{Q} \mid \begin{smallmatrix}\mathbf{0}\\ \mathbf{I}_{n-k-\ell}\end{smallmatrix}\right)$ with $\mathbf{Q} \in \mathbb{F}_2^{(n-k)\times(k+\ell)}$.

A different approach for removing the length-ℓ 0-window restriction in Stern's algorithm was recently proposed by Bernstein, Lange and Peters [4], called *Ball-collision decoding* by the authors. In Ball-collision decoding, one allows to have a small non-zero weight q in the length-ℓ window. Both algorithms, FS-ISD and Ball-collision decoding, share the same time complexity $\tilde{\mathcal{O}}\left(2^{0.05558n}\right)$ and memory complexity $\tilde{\mathcal{O}}\left(2^{0.014n}\right)$.

As a sideline of our work we show that any parameter choice (p,q,ℓ) for Ball-collision decoding can be transformed into parameters (p',ℓ') for the FS-ISD algorithm with the *same* asymptotic time complexity. That is, FS-ISD is asymptotically at least as efficient as Ball-collision decoding. We conjecture that both algorithms actually behave asymptotically equivalent. Since FS-ISD offers a simpler description than Ball-collision, we focus on improving the FS-ISD variant in this work.

Our Contribution. We provide a new information set decoding algorithm based on FS-ISD. The major subproblem in FS-ISD is to find exactly p columns of an ℓ-row submatrix $\mathbf{Q}'$ of the $(n-k)\times(k+\ell)$ matrix $\mathbf{Q}$ that sum to the corresponding ℓ coordinates of the syndrome $\mathbf{s}(\mathbf{x})$.

More precisely, let $\mathbf{Q}' = [\mathbf{q}'_1 \cdots \mathbf{q}'_{k+\ell}]$ and $\mathbf{s}'(\mathbf{x})$ be the projections of $\mathbf{Q}$ and $\mathbf{s}(\mathbf{x})$ on the desired ℓ coordinates. Then we have to find an index set $I \subseteq \{1,\ldots,k+\ell\}$ with $|I| = p$ and $\sum_{i\in I}\mathbf{q}'_i = \mathbf{s}'(\mathbf{x})$. We call this problem the *submatrix matching problem*. Our improvement of information set decoding comes from a more efficient algorithm for the submatrix matching problem than the birthday algorithm of Stern. Our algorithm for the submatrix matching problem might be of independent interest as this problem is again a parametrized version of syndrome decoding.

In FS-ISD, the submatrix matching problem is solved by splitting the interval $[1, k + \ell]$ into the two disjoint intervals $[1, \frac{k+\ell}{2}]$ and $[\frac{k+\ell}{2} + 1, k + \ell]$. Then one searches in a birthday-type manner for two index sets $I_1 \subset [1, \frac{k+\ell}{2}]$ and $I_2 \subset [\frac{k+\ell}{2} + 1, k + \ell]$ of cardinality $\frac{p}{2}$ each, such that $\sum_{i \in I_1} \mathbf{q}'_i = \sum_{i \in I_2} \mathbf{q}'_i + \mathbf{s}'(\mathbf{x})$.

Our approach is inspired by a clever *representation technique* used in a recent subset sum algorithm of Howgrave-Graham and Joux from Eurocrypt 2010 [8]. We choose I_1 and I_2 in the submatrix matching problem both from the whole interval $[1, k + \ell]$ instead of taking two disjoint intervals of size $\frac{k+\ell}{2}$. Let I be a solution with $\sum_{i \in I} \mathbf{q}'_i = \mathbf{s}'(\mathbf{x})$ and $|I| = p$.

Then the major observation is that I has $\binom{p}{p/2}$ different representations of the form $I = I_1 \cup I_2$ with $|I_1| = |I_2| = \frac{p}{2}$. Thus, we also have $\binom{p}{p/2}$ identities of the form

$$\sum_{i \in I_1} \mathbf{q}'_i = \sum_{i \in I_2} \mathbf{q}'_i + \mathbf{s}'(\mathbf{x}), \tag{1}$$

instead of just one unique representation as in FS-ISD.

Interestingly, Finiasz and Sendrier also allow for non-disjoint splittings in [5]. However, their framework does not make use of different representations. It is precisely the representation technique that allow us to bypass their lower bound argument and to asymptotically beat the lower bound for information set decoding given in [5]. Our algorithms achieves an asymptotic running time of $\tilde{\mathcal{O}}\left(2^{0.05363n}\right)$ using memory $\tilde{\mathcal{O}}\left(2^{0.021n}\right)$.

The correctness of our algorithm is rigorously proven under the assumption that $\mathbf{H}$ is a uniformly random $\{0, 1\}$-matrix. This assumption is plausible in the cryptographic setting, since it is actually the goal of crypto designers to hide the structure of the underlying code, e.g. the Goppa code in McEliece, by linear transformations.

Table 1. Comparison of exponents in the asymptotic worst-case complexities

	time	space
Lee-Brickell	$0.05751n$	-
Stern	$0.05563n$	$0.013n$
FS-ISD / Ball-collision	$0.05558n$	$0.014n$
Lower bound from [5]	$0.05556n$	$0.014n$
Our algorithm with FS-ISD space	$0.05402n$	$0.014n$
Our algorithm	$0.05363n$	$0.021n$

Table 1 summarizes the worst-case complexity of decoding algorithms. Notice that Stern's algorithm, FS-ISD and Ball-collision are typical time-memory tradeoffs that decrease the running time complexity at the cost of an increased memory complexity. In contrast, our algorithm does not only benefit from a mere time-memory tradeoff. For example, if we restrict our memory complexity to $\tilde{\mathcal{O}}\left(2^{0.014n}\right)$ as in FS-ISD we still obtain an improved running time.

Roadmap. Our paper is organized as follows. We first introduce some useful notation in Section 2. In Section 3, we briefly recall the state of the art in information set decoding, including Stern's algorithm, FS-ISD and Ball-Collision decoding. In Section 4, we provide an algorithm for the submatrix matching problem. This leads to our new information set decoding algorithm in Section 5, for which we provide some experimental results in Section 6.

2 Notation

By $[k]$ we define the set of natural numbers between 1 and k, i.e. $[k] = \{1, \ldots, k\}$. The cardinality of a finite set I is denoted by $|I|$. For a better readability we represent matrices $\mathbf{Q}$ and vectors $\mathbf{e}$ by bold letters. For index sets $I \subset [n]$, $J \subset [k]$ and an $n \times k$ matrix $\mathbf{Q} = (q_{i,j})_{i\in[n],j\in[k]} \in \mathbb{F}_2^{n\times k}$, we denote by $\mathbf{Q}_J^I := (q_{i,j})_{i\in I,j\in J}$ the submatrix containing the $|I|$ rows and $|J|$ columns defined by I and J, respectively. When we consider submatrices of $\mathbf{Q}$ where *either* columns *or* rows are chosen, we simply write $\mathbf{Q}_J$ or $\mathbf{Q}^I$ meaning $\mathbf{Q}_J = \mathbf{Q}_J^{[n]}$ and $\mathbf{Q}^I = \mathbf{Q}_{[k]}^I$. We extend this notion to vectors $\mathbf{s} \in \mathbb{F}_2^n$ and write $\mathbf{s}_L \in \mathbb{F}_2^{|L|}$ for the projection of $\mathbf{s}$ onto the coordinates defined by L. Further, for a matrix $\mathbf{Q} = (q_{i,j})_{i\in[n],k\in[k]} \in \mathbb{F}_2^{n\times k}$ and index sets $L \subseteq [n]$ with $|L| = \ell$, we define a mapping $\pi_L : \mathbb{F}_2^{n\times k} \to \mathbb{F}_2^\ell$ where

$$\pi_L(\mathbf{Q}) := \textstyle\sum_{i=1}^k \mathbf{Q}_{\{i\}}^L \in \mathbb{F}_2^\ell$$

is the projection of the sum of $\mathbf{Q}'s$ columns onto the ℓ rows defined by L. As before, we sometimes omit the index set L which means that we consider the sum of $\mathbf{Q}$'s columns *without* projecting it to a certain number of rows, i.e. $\pi(\mathbf{Q}) = \pi_{[n]}(\mathbf{Q}) \in \mathbb{F}_2^n$.

By $\mathrm{wt}(\mathbf{x})$ we denote the Hamming weight of a vector $\mathbf{x} \in \mathbb{F}_2^n$, i.e., $\mathrm{wt}(\mathbf{x})$ counts the number of non-zero entries of $\mathbf{x}$. By $\mathrm{supp}(\mathbf{x}) := \{i \in [n] : x_i = 1\}$ we denote the support of a vector $\mathbf{x}$, i.e., the set of indices corresponding to non-zero coordinates of $\mathbf{x} \in \mathbb{F}_2^n$. We represent the n-dimensional identity matrix by $\mathbf{I}_n$ and the i-th unit vector by $\mathbf{u}_i$. Observe that $\sum_{i\in\mathrm{supp}(\mathbf{x})} \mathbf{u}_i = \mathbf{x}$ for every $\mathbf{x} \in \mathbb{F}_2^n$. For a set of natural number $I \subset \mathbb{N}$, we introduce the shifted set $k+I := \{k+i : i \in I\}$ for arbitrary $k \in \mathbb{N}$.

Throughout the asymptotic complexity analysis of our *exponential* algorithms we make use of the soft Landau notation $\widetilde{\mathcal{O}}$ which suppresses arbitrary *polynomial* factors, i.e., $p(n)2^n = \widetilde{\mathcal{O}}(2^n)$ for every polynomial $p(n)$. We often need to estimate binomial coefficients of the form $\binom{\alpha n}{\beta n}$ asymptotically. Stirling's formula yields

$$\binom{\alpha n}{\beta n} = \widetilde{\mathcal{O}}(2^{\alpha H(\beta/\alpha)n}), \tag{2}$$

where $H(x) = -x\log_2(x) - (1-x)\log_2(1-x)$ is the binary entropy function.

3 Information Set Decoding Algorithms

3.1 Information Set Decoding

Let $\mathcal{C}$ be an $[n, k, d]$-code with parity check matrix $\mathbf{H}$. Furthermore, let $\mathbf{x} = \mathbf{c}+\mathbf{e}$, $\mathbf{c} \in \mathcal{C}$ be an erroneous codeword with error $\mathbf{e}$, $\mathrm{wt}(\mathbf{e}) = \lfloor \frac{d-1}{2} \rfloor$. In order to find $\mathbf{e}$, information set decoding proceeds as follows.

Initially, we apply a random permutation to the columns of $\mathbf{H}$, resulting in a permuted matrix $\tilde{\mathbf{H}}$. Then we apply Gaussian Elimination on the right-hand square submatrix $\tilde{\mathbf{H}}_I$, $I = \{k+1, \ldots, n\}$. If $\tilde{\mathbf{H}}_I$ is invertible, Gaussian Elimination will succeed and we obtain a systematic form[1] $(\mathbf{Q}|\mathbf{I_{n-k}})$ of $\tilde{\mathbf{H}}$, see Figure 1.

After the first step all the work can be done within the k columns of submatrix $\mathbf{Q}$. In the Lee-Brickell algorithm [9] one checks for every $I \subseteq [k]$ with cardinality $|I| = p$ whether $\mathrm{wt}(\pi(\mathbf{Q}_I) + \mathbf{s}(\mathbf{x})) = \omega - p$. If so, we can easily choose $\omega - p$ columns in the $\mathbf{I}_{n-k}$ part of $\tilde{\mathbf{H}}$ indexed by $J = k + \mathrm{supp}(\pi(\mathbf{Q}_I) + \mathbf{s}(\mathbf{x})) \subseteq [k+\ell+1, n]$ which eliminate the remaining 1-entries. This in turn implies that $\sum_{i \in I} \mathbf{Q}_{\{i\}} + \sum_{j \in J} \mathbf{u}_{j-k} = \mathbf{s}(\mathbf{x})$.

Therefore, I and J determine the support of the permuted error vector $\tilde{\mathbf{e}} = \mathbf{e}\mathbf{U}_P$, i.e., we can set $\mathrm{supp}(\tilde{\mathbf{e}}) := I \cup J$ which finally reveals the error $\mathbf{e}$.

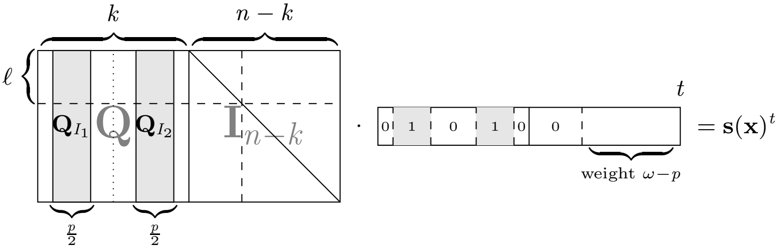

Fig. 1. Collision Decoding by Stern - $\mathbf{H}\mathbf{e}^t = \mathbf{s}(\mathbf{x})^t$. The error vector $\mathbf{e}$ contains two blocks each of $\frac{p}{2}$ 1's in its upper half corresponding to the columns of $\mathbf{Q}_{I_1}$ and $\mathbf{Q}_{I_2}$. Since $\mathbf{Q}_{I_1}$ and $\mathbf{Q}_{I_2}$ sum up to $\mathbf{s}(\mathbf{x})$ on the rows defined by $[\ell]$ we have to fix a corresponding zero-block in coordinates $\{k+1, \ldots, k+\ell\}$ of $\mathbf{e}$. The remaining $(\omega - p)$ 1's are then distributed over the remaining coordinates $\{k+\ell+1, \ldots, n\}$ of $\mathbf{e}$.

3.2 Stern's Algorithm

In the late 80s, Leon and Stern [13] introduced the idea of forcing the first ℓ coordinates of $\pi(\mathbf{Q}_I)$ already to the coordinates of $\mathbf{s}(\mathbf{x})$. Let $\mathbf{s}_{[\ell]}(\mathbf{x})$ be the projection of $\mathbf{s}(\mathbf{x})$ onto the coordinates in $[\ell]$.

We enumerate for all $I_1 \subseteq [1, \frac{k}{2}]$, $I_2 \subseteq [\frac{k}{2}+1, k]$ the projected vectors $\pi_{[\ell]}(\mathbf{Q}_{I_1})$ and $\pi_{[\ell]}(\mathbf{Q}_{I_2}) + \mathbf{s}_{[\ell]}(\mathbf{x})$ in two lists. Then we search for collisions in these lists,

[1] In more detail, we transform $\mathbf{H}$ by multiplying it by two invertible matrices $\mathbf{U}_P \in \mathbb{F}_2^{n \times n}$, $\mathbf{U}_G \in \mathbb{F}_2^{n-k \times n-k}$ corresponding to the initial column permutation and the Gaussian Elimination, respectively. Then $(\mathbf{Q}|\mathbf{I}) = \mathbf{U}_G(\mathbf{H}\mathbf{U}_P)$. Notice, that the transformation $\mathbf{U}_G$ also needs to be applied to the syndrome $\mathbf{s}(\mathbf{x})$, which we omit for simplicity of exposition.

meaning that we look for two weight-$\frac{p}{2}$ sums of columns that are equal to the syndrome $\mathbf{s}(\mathbf{x})$ within the coordinates of $[\ell]$.

If $\mathrm{wt}(\pi(\mathbf{Q}_{I_1}) + \pi(\mathbf{Q}_{I_2}) + \mathbf{s}(\mathbf{x})) = \omega - p$ holds for one of these collisions, we again set the corresponding $\omega - p$ coordinates in the second half of the permuted error vector $\tilde{\mathbf{e}}$ to 1, see Fig. 1 for an illustration.

To analyze Stern's algorithm we have to consider both the complexity of each iteration and the probability of success. The complexity of each iteration is dominated by the collision finding step in two lists. This can be done by a simple sort-and-match technique. Neglecting log factors, we obtain complexity

$$C_{\mathrm{Stern}}(p, \ell) := \max\left\{\binom{k/2}{p/2}, \frac{\binom{k/2}{p/2}^2}{2^\ell}\right\}. \tag{3}$$

In order to analyze the success probability, we need to compute the probability that a random permutation of the error $\mathbf{e} \in \mathbb{F}_2^n$ of weight $\mathrm{wt}\,(\mathbf{e}) = \omega$ has a good weight distribution, i.e., $\tilde{\mathbf{e}}$ needs to have weight $p/2$ both on its coordinates in $[1, k/2]$ and $[k/2+1, k]$ *and* zero-weight on all coordinates with indices in the set $\{k+1, \ldots, k+\ell\}$ as illustrated in Fig. 1. Thus, we obtain success probability

$$P_{\mathrm{Stern}}(p, \ell) := \frac{\binom{k/2}{p/2}^2 \binom{n-k-\ell}{\omega-p}}{\binom{n}{\omega}}. \tag{4}$$

The overall running time of Stern's algorithm is hence given by $C_{\mathrm{Stern}} \cdot P_{\mathrm{Stern}}^{-1}$. Optimizing this expression for p and ℓ under the natural constraints $0 \le p \le \omega$ and $0 \le \ell \le n - k - \omega + p$ we obtain time complexity $\tilde{\mathcal{O}}\left(2^{0.05563n}\right)$ and space complexity $\tilde{\mathcal{O}}\left(2^{0.013n}\right)$. The optimal parameter choice is given by $p = 0.003n$ and $\ell = 0.013n$.

3.3 The Finiasz-Sendrier ISD Algorithm

The idea of the FS-ISD algorithm is to increase the success probability for having a permuted error vector $\tilde{\mathbf{e}}$ of the desired form by allowing $\tilde{\mathbf{e}}$ to spread it's 1's over all coordinates, instead of fixing a certain ℓ-width 0-window. This is realized by changing the systematic form during the Gaussian Elimination process.

As before, we first randomly permute the columns of $\mathbf{H}$, which results in a permuted matrix $\tilde{\mathbf{H}} = \mathbf{H}\mathbf{U}_P$. Then we carry out a *partial Gaussian Elimination* on the right-hand lower square submatrix $\tilde{\mathbf{H}}_J^I \in \mathbb{F}_2^{(n-k-\ell)\times(n-k-\ell)}$ with index sets $I = \{\ell+1, \ldots, n-k\}$ and $J = \{k+\ell+1, \ldots, n\}$.

Next, we force an $\ell \times (n-k-\ell)$ zero block in the remaining ℓ rows of the submatrix $\tilde{\mathbf{H}}_J$ by adding rows of the identity matrix. Mathematically, we represent the partial Gaussian Elimination plus row elimination by a multiplication with an $(n-k) \times (n-k)$ invertible matrix $\mathbf{U}_G$. Therefore, the initial step in FS-ISD, which we denote $\mathsf{Init}(\mathbf{H})$, yields a modified systematic form

$$\left(\mathbf{Q} \middle| \begin{matrix} \mathbf{0} \\ \mathbf{I}_{n-k-\ell} \end{matrix}\right) = \mathbf{U}_G \mathbf{H} \mathbf{U}_P.$$

In Fig. 2, we illustrate the Birthday collision step of FS-ISD which is the same as in Stern's algorithm but for a submatrix $\mathbf{Q}^{[\ell]}$ which now has $k+\ell$ columns instead of k columns.

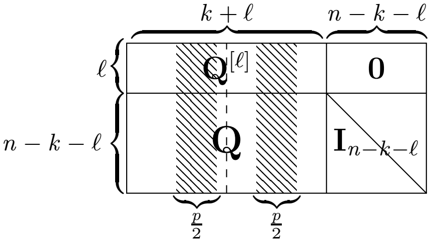

Fig. 2. Birthday collision search in FS-ISD

A straight-forward modification of the analysis of Stern's algorithm from Section 3 yields a complexity of

$$T_{\text{FS-ISD}}(p,\ell) := \max\left\{ S_{\text{FS-ISD}}(p,\ell), \frac{S_{\text{FS-ISD}}(p,\ell)^2}{2^\ell} \right\} \tag{5}$$

per iteration, where $S_{\text{FS-ISD}}(p,\ell) = \binom{(k+\ell)/2}{p/2}$ denotes the size of the initial lists and thus represents also the space complexity. Furthermore, the success probability of getting an error vector $\mathbf{e}$ of the desired form is now given by

$$P_{\text{FS-ISD}}(p,\ell) := \frac{\binom{(k+\ell)/2}{p/2}^2 \binom{n-k-\ell}{\omega-p}}{\binom{n}{\omega}} \ . \tag{6}$$

Thus, we obtain a total complexity of $C_{\text{FS-ISD}}(p,\ell) = T_{\text{FS-ISD}}(p,\ell) \cdot P_{\text{FS-ISD}}(p,\ell)^{-1}$. Optimizing this expression yields a worst-case running time of $\tilde{\mathcal{O}}\left(2^{0.05558n}\right)$ within space complexity $\tilde{\mathcal{O}}\left(2^{0.014n}\right)$. The optimal parameter choice is given by $p = 0.003n$ and $\ell = 0.014n$.

3.4 Ball-collision Decoding

In 2011, Bernstein, Lange and Peters [4] presented another information set decoding algorithm, which they called *Ball-collision decoding* (BCD for shorthand). The general idea of BCD is very similar to the idea of the FS-ISD algorithm, namely the authors increase the success probability of one iteration in Stern's algorithm by allowing an additional number of ones within the fixed width-ℓ 0-window.

Therefore, BCD allows for q additional 1's within the 0-window, or in other words for a Hamming ball of radius q within the 0-window. More precisely, let I be an index set with $|I| = \frac{p}{2}$ chosen from the intervals $[1, k/2]$ or $[k/2+1, k]$. Each entry $(I, \pi_{[\ell]}(\mathbf{Q}_I))$ in the initial lists of Stern's algorithm has to be expanded by all possible projected weight-$q/2$ column sums $\pi_{[\ell]}(\mathbf{I}_J)$ of the identity matrix

$\mathbf{I}$ – for index sets J of size $|J| = q/2$ contained either in $[k+1, k+\ell/2]$ or $[k+\ell/2+1, k+\ell]$.

Analogously to the analysis of Stern's algorithm in Sect. 3, we obtain an asymptotic time complexity for one iteration of BCD of

$$T_{\mathrm{BCD}}(p,\ell,q) := \max\left\{ S_{\mathrm{BCD}}(p,\ell,q), \frac{S_{\mathrm{BCD}}(p,\ell,q)^2}{2^\ell} \right\} . \tag{7}$$

The space consumption is $S_{\mathrm{BCD}}(p,\ell,q) = \binom{k/2}{p/2}\binom{\ell/2}{q/2}$. Similarly one obtains a success probability of

$$P_{\mathrm{BCD}}(p,\ell,q) := \frac{\left(\binom{k/2}{p/2}\binom{\ell/2}{q/2}\right)^2 \binom{n-k-\ell}{\omega-p-q}}{\binom{n}{\omega}} . \tag{8}$$

Eventually, the overall complexity of BCD is given by $C_{\mathrm{BCD}}(p,\ell,q) = T_{\mathrm{BCD}}(p,\ell,q) \cdot P_{\mathrm{BCD}}(p,\ell,q)^{-1}$.

Intuitively, FS-ISD and BCD proceed in a similar fashion by allowing $\tilde{\mathbf{e}}$ to spread its 1's in a more flexible way at the cost of slightly increasing the workload and space complexity per iteration. Indeed, the following theorem shows that FS-ISD is asymptotically at least as efficient as BCD.

Theorem 1. *Let (p,q,ℓ) be a parameter set for the BCD algorithm. Then $(p+q,\ell)$ is a parameter set for FS-ISD satisfying*

$$C_{\textit{FS-ISD}}(p+q,\ell) \le C_{\textit{BCD}}(p,\ell,q) .$$

Proof. See full version, available from the authors.

Due to Theorem 1, we take the FS-ISD algorithm as a starting point for our new construction, in which we improve on the birthday-collision step.

4 How to Solve the Submatrix Problem

Recall that in each iteration of the FS-ISD algorithm one has to find in a projected $\ell \times (k+\ell)$ - submatrix a weight-p sum of columns that sums to a target syndrome. We call this problem the *submatrix matching problem.*

Definition 1. *The* submatrix matching problem *with parameters ℓ, k and $p \le k+\ell$ is defined as follows. Given a random matrix $\mathbf{Q} = [\mathbf{q}_1 \ldots \mathbf{q}_{k+\ell}] \in_R \mathbb{F}_2^{\ell\times(k+\ell)}$ and a target vector $\mathbf{s} \in \mathbb{F}_2^\ell$, find an index set I of size at most p such that the corresponding columns of $\mathbf{Q}$ sum to $\mathbf{s}$, i.e., find $I \subset [k+\ell]$, $|I| \le p$ with*

$$\pi(\mathbf{Q}_I) = \textstyle\sum_{i\in I} \mathbf{q}_i = \mathbf{s} \in \mathbb{F}_2^\ell .$$

The submatrix matching problem is a vectorial variant of the well-known subset sum problem. In the following, we propose an algorithm ColumnMatch for the problem, based on a recently introduced *representation technique* for the subset sum problem by Howgrave-Graham and Joux [8].

When we use ColumnMatch in information set decoding, the input parameters p, ℓ are optimization parameters that guarantee that some solution I exists with a certain probability $P(p,\ell)$, compare e.g. with Eq.(6).

4.1 The ColumnMatch Algorithm

Let us briefly explain our ColumnMatch algorithm. We recommend the reader to follow our algorithm's description via the illustration given in Fig. 3 and the pseudocode description in Algorithm 1.

Let $\mathbf{Q} = [\mathbf{q}_1 \dots \mathbf{q}_{k+\ell}] \in_R \mathbb{F}_2^{\ell \times (k+\ell)}$ and $\mathbf{s} \in \mathbb{F}_2^\ell$ be an instance of the submatrix matching problem. Assume that I is a solution to the problem of size exactly p. Similar to FS-ISD we construct I from two sets I_1, I_2 of size $\frac{p}{2}$ each.

As opposed to FS-ISD, we do not choose I_1 and I_2 from disjoint sets of size $\frac{k+\ell}{2}$. Rather we choose both I_1, I_2 from the full set $[k+\ell]$. This choice of the index sets is similar to what we call the *representation technique* due to Howgrave-Graham and Joux [8]. The effect of the choice is that we obtain $\binom{p}{p/2} \approx 2^p$ different partitions $I = I_1 \dot{\cup} I_2$ and therefore the same number of identities

$$\sum_{i \in I_1} \mathbf{q}_i = \sum_{i \in I_2} \mathbf{q}_i + \mathbf{s} \text{ in } \mathbb{F}_2^\ell . \tag{9}$$

Our goal is to find one of these identities with constant success probability, where the probability is taken over the random choice of $\mathbf{Q}$. Therefore we do not construct all possible sums of elements in I_1, I_2 but only those that satisfy additional constraints. To establish the constraints, we introduce shortening parameters ℓ_1, ℓ_2 with $\ell_1 + \ell_2 = \ell$ that correspond to disjoint subsets $L_1, L_2 \subset [l]$ of size ℓ_1, ℓ_2, respectively.

Our construction now proceeds in two steps. In the first step, we construct partial solutions that already sum to the target value $\mathbf{s}$ on the ℓ_2 positions of L_2. More precisely, we construct two lists

$$\mathcal{L}_1 := \left\{ (I_1, \pi_{L_1}(\mathbf{Q}_{I_1})) : I_1 \subset [k+\ell], |I_1| = \tfrac{p}{2} \text{ and } \pi_{L_2}(\mathbf{Q}_{I_1}) = \mathbf{0} \in \mathbb{F}_2^{\ell_2} \right\} \text{ and}$$
$$\mathcal{L}_2 := \left\{ (I_2, \pi_{L_1}(\mathbf{Q}_{I_2}) + \mathbf{s}_{L_1}) : I_2 \subset [k+\ell], |I_2| = \tfrac{p}{2} \text{ and } \pi_{L_2}(\mathbf{Q}_{I_2}) = \mathbf{s}_{L_2} \in \mathbb{F}_2^{\ell_2} \right\} .$$

Notice that out of the 2^p possible identities that satisfy Eq. (9), we consider only those identities where $\sum_{i \in I_1} \mathbf{q}_i$ is equal to $\mathbf{0} \in \mathbb{F}_2^{l_2}$ on the bits of L_2. Thus we expect that we already remove a $2^{-\ell_2}$-fraction of all solutions, which lets an expected number of $2^{p-\ell_2}$ solutions survive.

Once we have constructed the lists $\mathcal{L}_1, \mathcal{L}_2$ in the first step, we sort $\mathcal{L}_2$ according to the labels $\pi_{L_1}(\mathbf{Q}_{I_2}) + \mathbf{s}_{L_1}$ and search for all elements $\pi_{L_1}(\mathbf{Q}_{I_1})$ in $\mathcal{L}_1$ for a matching element in $\mathcal{L}_2$. Notice that every matching (I_1, I_2) fulfills Eq. (9) and hence is a solution to the submatrix matching problem.

Since we constructed I_1, I_2 in a non-disjoint way, their intersection $J = I_1 \cap I_2$ might be non-empty. In this case, all vectors in J appear on both sides of Eq. (9) and thus cancel out when we compute $\sum_{i \in I_1} \mathbf{q}_i + \sum_{i \in I_2} \mathbf{q}_i$ over $\mathbb{F}_2^\ell$. This means that we have found a solution $I' = I_1 \Delta I_2 = (I_1 \cup I_2) \setminus (I_1 \cap I_2)$ to the submatrix matching problem with size $|I'| = p - 2|I_1 \cap I_2|$.

How to construct $\mathcal{L}_1$ and $\mathcal{L}_2$. The initial lists $\mathcal{L}_1$ and $\mathcal{L}_2$ can be easily constructed by a classical sort-and-match step. Let us show how to construct

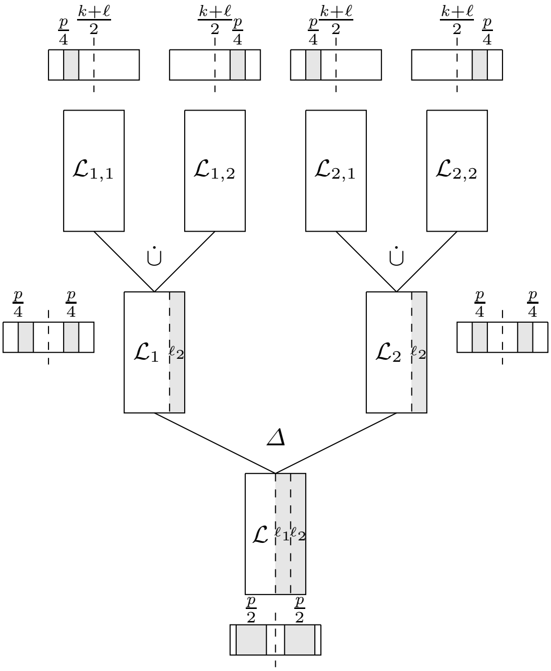

Fig. 3. Illustration of the ColumnMatch algorithm. The flat rectangles above, beside or below the lists represent the structure of the index sets $I_{i,j}$ contained in distinct lists, e.g., the level-2 list $\mathcal{L}_{1,1}$ contains index sets $I_{1,1}$ whose $\frac{p}{4}$ ones are spread over the first half of $[k+\ell]$ (as illustrated by the gray region).

$\mathcal{L}_1$, the construction of $\mathcal{L}_2$ is analogous. We partition $I_1 = I_{1,1} \dot\cup I_{1,2}$ with $|I_{1,1}| = |I_{1,2}| = \frac{p}{4}$ where $I_{1,1} \subset [1, \frac{k+\ell}{2}]$ and $I_{1,2} \subset [\frac{k+\ell}{2}+1, k+\ell]$. More precisely, we compute two lists

$$\mathcal{L}_{1,1} := \left\{ \left(I_{1,1}, \pi_{L_2}(\mathbf{Q}_{I_{1,1}}) \right) : I_{1,1} \subset [1, \tfrac{k+\ell}{2}], |I_{1,1}| = \tfrac{p}{4} \right\} \text{ and}$$
$$\mathcal{L}_{1,2} := \left\{ \left(I_{1,2}, \pi_{L_2}(\mathbf{Q}_{I_{1,2}}) \right) : I_{1,2} \subset [\tfrac{k+\ell}{2}+1, k+\ell], |I_{1,2}| = \tfrac{p}{4} \right\} \quad .$$

We then sort $\mathcal{L}_{1,2}$ with respect to the second component and search for all second components in $\mathcal{L}_{1,1}$ for matching elements in $\mathcal{L}_{1,2}$.

Remark 1. Notice that the construction of $\mathcal{L}_1$ and $\mathcal{L}_2$ via disjoint splittings $I_1 = I_{1,1} \dot\cup I_{1,2}$ and $I_2 = I_{2,1} \dot\cup I_{2,2}$ lowers the number of representations $\mathcal{R}(p)$. Instead of considering *every* subset $I_1 \subset I$ of size $\frac{p}{2}$ we take every I_1 with an equal number of $\frac{p}{4}$ indices coming from $[1, (k+\ell)/2]$ and $[(k+\ell)/2+1, k+\ell]$, respectively. Hence, we only have $\binom{p/2}{p/4}^2$ instead of $\binom{p}{p/2}$ many different representations per solution in Eq. (9). Asymptotically, this can be neglected since both terms equal $2^{p(1-o(1))}$.

Algorithm 1. ColumnMatch

Input: $\mathbf{Q} \in \mathbb{F}_2^{\ell \times (k+\ell)}$, $\mathbf{s} \in \mathbb{F}_2^{\ell}$, $p \leq k+\ell$
Output: I with $\pi(\mathbf{Q}_I) = \mathbf{s}$ or $\perp$ if no solution is found
Parameters: L_1, L_2 with $[l] = L_1 \dot\cup L_2$ and $|L_i| = \ell_i$ for $i = 1, 2$.

```
Construct $\mathcal{L}_{1,1}, \mathcal{L}_{1,2}, \mathcal{L}_{2,1}, \mathcal{L}_{2,2}$.
Sort $\mathcal{L}_{1,2}, \mathcal{L}_{2,2}$ according to their labels $\pi_{L_2}(\mathbf{Q}_{I_{1,2}})$, $\pi_{L_2}(\mathbf{Q}_{I_{2,2}}) + \mathbf{s}_{L_2}$.
Join $\mathcal{L}_{1,1}$ and $\mathcal{L}_{1,2}$ to $\mathcal{L}_1$, i.e., for all $(I_{1,1}, \pi_{L_2}(\mathbf{Q}_{I_{1,1}})) \in \mathcal{L}_{1,1}$ do
    for all $(I_{1,2}, \pi_{L_2}(\mathbf{Q}_{I_{1,2}})) \in \mathcal{L}_{1,2}$ with $\pi_{L_2}(\mathbf{Q}_{I_{1,1}}) = \pi_{L_2}(\mathbf{Q}_{I_{1,2}})$ do
        $I_1 = I_{1,1} \cup I_{1,2}$. Insert $(I_1, \pi_{L_1}(\mathbf{Q}_{I_1}))$ into $\mathcal{L}_1$.
Join $\mathcal{L}_{2,1}$ and $\mathcal{L}_{2,2}$ to $\mathcal{L}_2$, i.e., for all $(I_{2,1}, \pi_{L_2}(\mathbf{Q}_{I_{2,1}})) \in \mathcal{L}_{2,1}$ do
    for all $(I_{2,2}, \pi_{L_2}(\mathbf{Q}_{I_{2,2}}) + \mathbf{s}_{L_2}) \in \mathcal{L}_{2,2}$ with $\pi_{L_2}(\mathbf{Q}_{I_{2,1}}) = \pi_{L_2}(\mathbf{Q}_{I_{2,2}}) + \mathbf{s}_{L_2}$ do
        $I_2 = I_{2,1} \cup I_{2,2}$. Insert $(I_2, \pi_{L_1}(\mathbf{Q}_{I_2}) + \mathbf{s}_{L_1})$ into $\mathcal{L}_2$.
Sort $\mathcal{L}_2$ according to the label $\pi_{L_1}(\mathbf{Q}_{I_2}) + \mathbf{s}_{L_1}$.
Join $\mathcal{L}_1$ and $\mathcal{L}_2$ to $\mathcal{L}$, i.e., for all $(I_1, \pi_{L_1}(\mathbf{Q}_{I_1})) \in \mathcal{L}_1$ do
    for all $(I_2, \pi_{L_1}(\mathbf{Q}_{I_2}) + \mathbf{s}_{L_1}) \in \mathcal{L}_2$ with $\pi_{L_1}(\mathbf{Q}_{I_1}) = \pi_{L_1}(\mathbf{Q}_{I_2}) + \mathbf{s}_{L_1}$ do
        Output $I_1 \Delta I_2 = (I_1 \cup I_2) \setminus (I_1 \cap I_2)$.
Output $\perp$.
```

Time and space complexity. Throughout the analysis, we will again ignore low-order terms that are polynomial in the parameters p, ℓ. The space complexity of constructing the four level-2 lists $\mathcal{L}_{1,1}, \mathcal{L}_{1,2}, \mathcal{L}_{2,1}, \mathcal{L}_{2,2}$ is bounded by the length $\binom{(k+\ell)/2}{p/4}$ of these lists. The sort-and-match step of these lists can be done in time

$$\max\left\{ \binom{(k+\ell)/2}{p/4}, \binom{(k+\ell)/2}{p/4}^2 \cdot 2^{-\ell_2} \right\} .$$

Joining lists $\mathcal{L}_{1,1}$ and $\mathcal{L}_{1,2}$ to list $\mathcal{L}_1$ produces a list of expected size

$$\mathbb{E}[|\,\mathcal{L}_1\,|] = \binom{(k+\ell)/2}{p/4}^2 \cdot 2^{-\ell_2} = \widetilde{\mathcal{O}}(2^{(k+\ell)H(\frac{p}{2(k+\ell)}) - \ell_2}).$$

The final sort-and-match step of $\mathcal{L}_1$ and $\mathcal{L}_2$ on level 1 then takes expected time

$$\max\left\{ \mathbb{E}[|\,\mathcal{L}_1\,|], \frac{\mathbb{E}[|\,\mathcal{L}_1\,|] \cdot \mathbb{E}[|\,\mathcal{L}_2\,|]}{2^{\ell_1}} \right\} = \max\left\{ \binom{(k+\ell)/2}{p/4}^2 \cdot 2^{-\ell_2}, \binom{(k+\ell)/2}{p/4}^4 \cdot 2^{-2\ell_2 - \ell_1} \right\} .$$

The following table summarizes the exponents in the complexities for both levels of our algorithm ColumnMatch. This means that e.g. on level 2, we have space complexity $\widetilde{\mathcal{O}}(2^{S_2(k,p,\ell)})$. All binomial coefficients are estimated via Eq.(2).

The total time and space complexity for ColumnMatch is hence given by

$$S(k, p, \ell, \ell_2) = \max\{S_2(k, p, \ell), S_1(k, p, \ell, \ell_2)\} \text{ and}$$

$$T(k, p, \ell, \ell_1, \ell_2) = \max\{S_2(k, p, \ell), S_1(k, p, \ell, \ell_2), 2S_1(k, p, \ell, \ell_2) - \ell_1\} .$$

Table 2. Exponents of time and space complexities

level	space	time
2	$S_2(k,p,\ell) := \frac{k+\ell}{2} H(\frac{p}{2(k+\ell)})$	$\max\{S_2(k,p,\ell), 2S_2(k,p,\ell) - \ell_2\}$
1	$S_1(k,p,\ell,\ell_2) := 2S_2(k,p,\ell) - \ell_2$	$\max\{S_1(k,p,\ell,\ell_2), 2S_1(k,p,\ell,\ell_2) - \ell_1\}$

Theorem 2. *Let* $\mathbf{Q} \in_R \mathbb{F}_2^{\ell \times (k+\ell)}$, $\mathbf{s} \in \mathbb{F}_2^{\ell}$ *and* $p \leq k + \ell$. *Let* $\hat{I}$ *be a solution of the submatrix matching problem for* $\mathbf{Q}, \mathbf{s}$. *For sufficiently large* p ColumnMatch *finds* $\hat{I}$ *with probability at least* $\frac{1}{2}$ *in time* $\tilde{\mathcal{O}}(2^{T(k,p,\ell,\ell_1,\ell_2)})$ *and space* $\tilde{\mathcal{O}}(2^{S(k,p,\ell,\ell_2)})$ *as long as* $\ell_2 \leq p - 2$.

Proof. We already proved the claim about the time and space complexity. It remains to show that ColumnMatch succeeds with probability at least $\frac{1}{2}$.

To analyze the success probability of ColumnMatch we introduce a random variable X that counts the number of representations $I = I_1 \dot{\cup} I_2$ of the solution $\hat{I}$ in lists $\mathcal{L}_1$ and $\mathcal{L}_2$. Our goal is to show that at least one representation survives in our algorithm with probability at least $\frac{1}{2}$.

Notice that we have a total number of $\mathcal{R}(p) := \binom{p/2}{p/4}^2$ representations on level 1. To analyze X we introduce $\mathcal{R}(p)$ indicator variables X_I where $X_I = 1$ iff representation $I = I_1 \dot{\cup} I_2$ of $\hat{I}$ is contained in $\mathcal{L}_1$, i.e.,

$$X_I = \begin{cases} 1 & \text{if } \pi_{L_2}(\mathbf{Q}_{I_1}) = \mathbf{0} \\ 0 & \text{otherwise} \end{cases} .$$

Note that $X = \sum X_I$. The Second Moment Method [1] now lower bounds the success probability $\mathbf{Pr}[X \geq 1]$ by upper bounding $\mathbf{Pr}[X = 0] = 1 - \mathbf{Pr}[X \geq 1]$ using Chebyshev's inequality

$$\mathbf{Pr}[X = 0] \leq \frac{\mathrm{Var}[X]}{\mathbb{E}[X]^2} = \frac{\sum_I \mathrm{Var}[X_I] + \sum_{I \neq J} \mathrm{Cov}[X_I, X_J]}{\mathbb{E}[X]^2}. \quad (10)$$

Here the covariance has to be computed over all different representations $I \neq J$ of the solution $\hat{I}$. Essentially, for every representation I there is exactly one different representation J for which X_I and X_J are dependent, otherwise they are pairwise independent and hence $\mathrm{Cov}[X_I, X_J] = 0$.

We write $I = I_1 \dot{\cup} I_2$ with $|I_1|, |I_2| = \frac{p}{2}$ and analogously $J = J_1 \dot{\cup} J_2$. Notice that for all choices $J_1 \neq I \setminus I_1$, the random variables X_I and X_J are pairwise independent because $\mathbf{Q}$ contains randomly distributed columns.

Let $J_1 = I \setminus I_1$. Since $\pi(\mathbf{Q}_I) = \mathbf{s}$, we have

$$\pi_{L_2}(\mathbf{Q}_{I_1}) = \pi_{L_2}(\mathbf{Q}_{J_1}) + \mathbf{s}_{L_2}.$$

If $\mathbf{s}_{L_2} \neq \mathbf{0}$ then $\pi_{L_2}(\mathbf{Q}_{I_1}) \neq \pi_{L_2}(\mathbf{Q}_{J_1})$ which implies that $X_I X_J = 0$. Therefore $\mathrm{Cov}[X_I, X_J] = \mathbb{E}[X_I X_J] - \mathbb{E}[X_I]\,\mathbb{E}[X_J] = -\mathbb{E}[X_I]\,\mathbb{E}[X_J] < 0$. Hence we can bound Eq.(10) as $\mathbf{Pr}[X = 0] \leq \frac{\sum_I \mathrm{Var}[X_I]}{\mathbb{E}[X]^2}$.

If $\mathbf{s}_{L_2} = \mathbf{0}$ then $\pi_{L_2}(\mathbf{Q}_{I_1}) = \pi_{L_2}(\mathbf{Q}_{J_1})$ which implies $X_I = X_J$. This means that for every I there is exactly one $J \neq I$ such that $\mathrm{Cov}[X_I, X_J] = \mathrm{Cov}[X_I, X_I] = \mathrm{Var}[X_I]$. In this case, we can bound Eq.(10) as

$$\mathbf{Pr}\left[X = 0\right] \leq \tfrac{2\sum_I \mathrm{Var}[X_I]}{\mathbb{E}[X]^2}.$$

Example 1. Consider the case $k = 8$ and $p = 4$ with $\mathbf{Q} = (\mathbf{q}_1, \ldots, \mathbf{q}_8)$, $\mathbf{s} = \mathbf{0}$ and $\hat{I} = \{1, 2, 5, 6\}$. The representations $I = I_1 \dot{\cup} I_2 = \{1, 5\} \dot{\cup} \{2, 6\}$ and $J = J_1 \dot{\cup} J_2 = \{2, 6\} \dot{\cup} \{1, 5\}$ have identical indicator variables X_I, X_J. However I and $K = \{2, 5\} \dot{\cup} \{1, 6\}$ have independent indicator variables since $\mathbf{Pr}\left[X_K = 1 | X_I = 1\right] = \mathbf{Pr}\left[\mathbf{q}_2 + \mathbf{q}_5 = \mathbf{0} | \mathbf{q}_1 + \mathbf{q}_5 = \mathbf{0}\right] = \mathbf{Pr}\left[\mathbf{q}_2 = \mathbf{q}_1\right] = 2^{-\ell_2} = \mathbf{Pr}\left[X_K = 1\right]$.

We further observe that

$$\mathrm{Var}[X_I] = \mathbb{E}[X_I^2] - (\mathbb{E}[X_I])^2 = \mathbb{E}[X_I] - (\mathbb{E}[X_I])^2 \leq \mathbb{E}[X_I].$$

Therefore, we obtain

$$\mathbf{Pr}\left[X = 0\right] \leq \frac{2\sum_I \mathrm{Var}[X_I]}{\mathbb{E}[X]^2} \leq \frac{2\sum_I \mathbb{E}[X_I]}{\mathbb{E}[X]^2} \leq \frac{2\,\mathbb{E}[\sum_I X_I]}{\mathbb{E}[X]^2} = \frac{2\,\mathbb{E}[X]}{\mathbb{E}[X]^2} = \frac{2}{\mathbb{E}[X]} \ .$$

Since $\mathbb{E}[X] = \mathcal{R}(p)2^{-\ell_2} \geq 2^{p(1-o(1))-\ell_2}$, putting the restriction $\ell_2 \leq p - 2$ on the choice of the parameter ℓ_2 yields for large enough p

$$\mathbf{Pr}\left[X = 0\right] \leq 2^{1-p(1-o(1))+\ell_2} \rightarrow 2^{\ell_2-(p-1)} \leq \frac{1}{2} \ .$$

This in turn implies that our algorithm ColumnMatch succeeds in constructing at least one representation of the solution with probability at least $\frac{1}{2}$. □

5 Our New Decoding Algorithm

Let us start by giving a high-level description of our new information set decoding algorithm which we call Decode. Let $\mathbf{H} \in \mathbb{F}_2^{(n-k)\times n}$ be a parity check matrix of an $[n, k, d]$-code $\mathcal{C}$. Assume that we want to decode $\mathbf{x} = \mathbf{c} + \mathbf{e}$ with $\mathbf{c} \in \mathcal{C}$, $\omega := \mathrm{wt}(\mathbf{e}) = \lfloor \frac{d-1}{2} \rfloor$. That means we want to find ω columns in $\mathbf{H}$ that sum to the syndrome $\mathbf{s}(\mathbf{x}) = \mathbf{H}\mathbf{x}^t$. As described in Sect. 3.3, we start with the initial transformation on the parity check matrix $\mathbf{H}$ and obtain the modified systematic form

$$\tilde{\mathbf{H}} = \mathsf{Init}(\mathbf{H}) = \mathbf{U}_G \mathbf{H} \mathbf{U}_P = \left(\mathbf{Q} \middle| \begin{matrix} \mathbf{0} \\ \mathbf{I}_{n-k-\ell} \end{matrix} \right) \ .$$

This process also permutes $\mathbf{e}$ to $\tilde{\mathbf{e}} = \mathbf{U}_P \mathbf{e}$. Let $p \leq \omega$ be an optimization parameter. We need that the ω ones in $\tilde{\mathbf{e}}$ are distributed as $\frac{p}{2}, \frac{p}{2}, \omega - p$ in the coordinate intervals $[1, (k+\ell)/2], [(k+\ell)/2+1, k+\ell], [k+\ell+1, n]$ of $\tilde{\mathbf{e}}$, respectively.

Recall from Section 3.3 that $\tilde{\mathbf{e}}$ happens to have the correct form with probability

$$P_{ColumnMatch}(p,\ell) := \frac{\binom{(k+\ell)/2}{p/2}^2 \binom{n-k-\ell}{\omega-p}}{\binom{n}{\omega}} . \tag{11}$$

We now look within the submatrix $\mathbf{Q}^{[\ell]}$ of $\mathbf{Q}$ for a weight-p sum of the columns that exactly matches the projection of the syndrome to the first ℓ rows.

In the Decode algorithm, we now apply our ColumnMatch algorithm to $\mathbf{Q}^{[\ell]} \in \mathbb{F}_2^{\ell\times(k+\ell)}$ with the projected syndrome as target vector and a solution weight of p.

In each iteration of Decode, our ColumnMatch algorithm yields with probability at least $\frac{1}{2}\cdot P_{ColumnMatch}(p,\ell)$ at least one index set I, $|I| \leq p$ such that $\pi_{[\ell]}(\mathbf{Q}_I)$ exactly matches the projected syndrome. Thus we already match the syndrome on ℓ coordinates using a weight-$|I|$ linear combination of columns from $\mathbf{Q}$. If the remaining coordinates of $\pi(\mathbf{Q}_I)$ differ from the syndrome only by $w-|I|$ 1-entries, then we can correct these entries by choosing $w-|I|$ unit vectors from $\mathbf{I}_{n-k-\ell}$. Let us summarize our decoding algorithm by giving a pseudo-code description in Algorithm 2.

Algorithm 2. Decode

```
Input: Parity check matrix $\mathbf{H} \in \mathbb{F}_2^{(n-k)\times n}$, syndrome $\mathbf{s}(\mathbf{x}) = \mathbf{He}^t$ with $\mathrm{wt}(\mathbf{e}) = \omega$.
Output: Error $\mathbf{e} \in \mathbb{F}_2^n$
Parameters: $p$, $\ell$, $\ell_1, \ell_2$ with $\ell = \ell_1 + \ell_2$

Repeat
    Compute $\tilde{\mathbf{H}} \leftarrow \mathsf{Init}(\mathbf{H})$ where $\tilde{\mathbf{H}} = \mathbf{U}_G\mathbf{H}\mathbf{U}_P$.
    For all (solutions $I$ found by ColumnMatch($\mathbf{Q}^{[\ell]}, (\mathbf{U}_G\mathbf{s}^t(\mathbf{x}))_{[\ell]}, p, \ell_1, \ell_2$)) do
            If $\mathrm{wt}(\pi(\mathbf{Q}_I) + \mathbf{U}_G\mathbf{s}^t(\mathbf{x})) = \omega - |I|$ then
                    Compute $\widetilde{\mathbf{e}} \in \mathbb{F}_2^n$ by setting
                    $\widetilde{e}_i = 1\ \forall i \in I$
                    $\widetilde{e}_{k+\ell+j} = 1\ \forall j \in \mathrm{supp}(\pi_{[n-k]\setminus[\ell]}(\mathbf{Q}_I + \mathbf{U}_G\mathbf{s}^t(\mathbf{x})))$
                    Output $\mathbf{e} = \widetilde{\mathbf{e}}\mathbf{U}_P^t$.
```

The correctness of Decode is implied by correctness of the ColumnMatch algorithm as we show in the following lemma.

Lemma 1. Decode *is correct, i.e., if* Decode *outputs error* $\mathbf{e}$ *then* $\mathbf{He^t} = \mathbf{s}(\mathbf{x})$ *and* $\mathrm{wt}\,(\mathbf{e}) = \omega$.

Proof. Let I be an output of ColumnMatch, i.e., $\pi_{[\ell]}(\mathbf{Q}_I) = (\mathbf{U}_G\mathbf{s}^t(\mathbf{x}))_{[\ell]}$ and $0 < |I| \leq p$. Furthermore, we have

$$\begin{aligned}\mathbf{U}_G\mathbf{He}^t &= \mathbf{U}_G\mathbf{H}\mathbf{U}_P\widetilde{\mathbf{e}}^t = \widetilde{\mathbf{H}}\widetilde{\mathbf{e}}^t = \left(\mathbf{Q}\middle|\begin{matrix}\mathbf{0}\\ \mathbf{I}_{n-k-\ell}\end{matrix}\right)\tilde{\mathbf{e}}^t = \mathbf{Q}\tilde{\mathbf{e}}^t_{[k+\ell]} + \begin{pmatrix}\mathbf{0}\\ \mathbf{I}\end{pmatrix}\tilde{\mathbf{e}}^t_{[n]\setminus[k+\ell]}\\ &= \pi(\mathbf{Q}_I) + \begin{pmatrix}\mathbf{0}\\ \pi_{[n-k]\setminus[\ell]}(\mathbf{Q}_I + \mathbf{U}_G\mathbf{s}^t(\mathbf{x}))\end{pmatrix} = \begin{pmatrix}\mathbf{U}_G\mathbf{s}^t_{[\ell]}(\mathbf{x})\\ \mathbf{U}_G\mathbf{s}^t_{[n-k]\setminus[\ell]}(\mathbf{x})\end{pmatrix} = \mathbf{U}_G\mathbf{s}^t(\mathbf{x}) \ .\end{aligned}$$

Since $\mathbf{U}_G$ is invertible, it follows that $\mathbf{He}^t = \mathbf{s}(\mathbf{x})$. Moreover, from line 03 of Decode we obtain that

$$\mathrm{wt}\,(\mathbf{e}) = \mathrm{wt}(\widetilde{\mathbf{e}}) = |I| + \mathrm{wt}(\pi(\mathbf{Q}_I) + \mathbf{U}_G \mathbf{s}^t(\mathbf{x})) = |I| + \omega - |I| = \omega. \qquad \square$$

In the remaining part of this section we explain how to derive optimal parameter choices for the Decode algorithm. We parametrize our code by $k = c_k n$, $\omega = c_\omega n$. We also parametrize the algorithm's optimization parameters as $\ell_1 = c_{\ell_1} n$, $\ell_2 = c_{\ell_2} n$ and $p = c_p n$.

Optimal parameters for the Decode algorithm. Recall that a randomly permuted error $\tilde{\mathbf{e}} \in \mathbb{F}_2^n$ of weight $\mathrm{wt}\,(\tilde{\mathbf{e}}) = \omega$ has the desired weight distribution of 1-entries with probability $P_{ColumnMatch}(p, \ell)$ from Eq. (11) as in the FS-ISD algorithm. Thus the inverse success probability is asymptotically $P^{-1}_{ColumnMatch}(p, \ell) = \tilde{\mathcal{O}}(2^{\alpha n})$ with

$$\alpha(c_p, c_\ell) = H(c_\omega) - \left((c_k + c_\ell) H\left(\frac{c_p}{c_k + c_\ell}\right) + (1 - c_k - c_\ell) H\left(\frac{c_\omega - c_p}{1 - c_k - c_\ell}\right)\right) \ .$$

For a fixed choice of the parameters ℓ_1, ℓ_2 and p, the asymptotic time and space complexities of one iteration of Decode are given by $2^{T(k,p,\ell,\ell_1,\ell_2)n}$ and $2^{S(k,p,\ell,\ell_2)n}$ from Theorem 2. In order to apply Theorem 2 we need to further ensure that $\ell_2 \le p - 2$, which asymptotically simplifies to $c_{\ell_2} \le c_p + \frac{2}{n} \to c_p$.

In total, we have to solve the following optimization problem

$$\begin{aligned} &\min\{T(c_k, c_p, c_\ell, c_{\ell_1}, c_{\ell_2}) + \alpha(c_p, c_{\ell_1} + c_{\ell_2})\} \\ &\text{s.t.} \quad 0 \le c_p \le c_\omega \\ &\qquad\ \ 0 \le c_{\ell_1} + c_{\ell_2} \le 1 - c_k - c_\omega + c_p \\ &\qquad\ \ 0 \le c_{\ell_2} \le c_p \\ &\qquad\ \ 0 \le c_{\ell_1}. \end{aligned} \tag{OPT}$$

We solve (OPT) numerically for various code rates $0 \le c_k \le 1$. Since random linear codes attain the Gilbert-Varshamov bound [6], we related the value c_w for the maximal error-correction capability to c_k by the identity $c_k = 1 - H(2c_\omega)$.

For every code rate $0 \le c_k \le 1$ on the x-axis we plotted the complexity of Decode in comparison with the FS-ISD algorithm, see Fig. 4 and Fig. 5. This shows that our Decode algorithm yields for all rates c_k an exponential improvement over the best-known decoding algorithms FS-ISD and Ball-collision decoding. If we additionally plot the lower bound curve from [5] in its asymptotical form, then this curve lies strictly below the FS-ISD curve and strictly above our new curve. This shows that the representation technique in our Decode algorithm allows to bypass the lower bound framework from [5].

We obtained the worst-case complexity for $c_k \approx 0.47n$ with the parameter choice as stated in the following main result.

Theorem 3. Decode *recovers* $\mathbf{e}$ *in time* $\widetilde{\mathcal{O}}(2^{0.05363n})$ *and space* $\widetilde{\mathcal{O}}(2^{0.021n})$, *where the optimal parameter choice is* $c_p = c_{\ell_2} = 0.006$ *and* $c_{\ell_1} = 0.028$.

Our formulation as an optimization problem (OPT) easily allows to specify additional space constraints. E.g. adding the restriction $S(c_k, c_p, c_\ell, c_{\ell_2}) \leq 0.014$ gives us a running time of $\tilde{\mathcal{O}}\left(2^{0.05402n}\right)$ using the same space $\tilde{\mathcal{O}}\left(2^{0.014n}\right)$ as in FS-ISD/Ball-collision decoding.

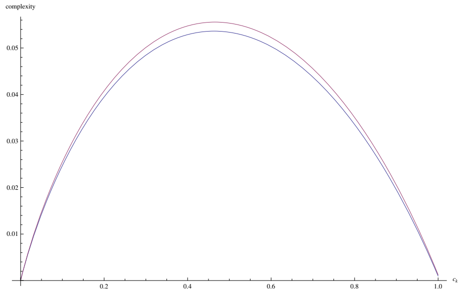

Fig. 4. Run time comparison with FS-ISD

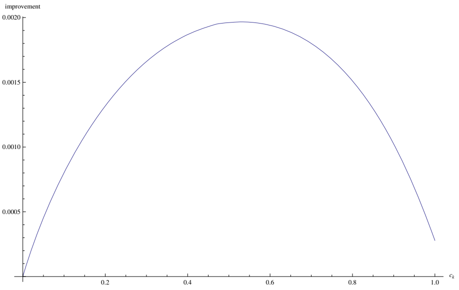

Fig. 5. Improvement over FS-ISD

6 Experiments

We implemented our Decode and ColumnMatch algorithms in C++ and tested them on three small McEliece instances with underlying $[n, k, \omega]$-Goppa codes. For each instance we computed optimal parameters p, ℓ_1, ℓ_2 (see second column of Table 3) using the *exact* formulas for the time and space complexities from Sect. 4 as well as for the respective probabilities from Eq. (11). We then carried out 10.000 experiments per McEliece instance with varying $\mathbf{Q}$. We computed the target syndrome $\mathbf{s} = \mathbf{Q}\mathbf{e}^t$ for an error vector $\mathbf{e}$ fulfilling the required weight distribution, i.e., we fixed $p/2$ coordinates to 1 in both intervals $[1, \frac{k+\ell}{2}]$ and $[\frac{k+l}{2}+1, k+\ell]$.

Recall that our sole heuristic assumption was that $\mathbf{Q}$ behaves as a uniformly random matrix, implying that the projected partial sums $\pi_{L_j}(\mathbf{Q}_I)$ are distributed uniformly at random as well. To verify this assumption experimentally, we determined the average list size of $\mathcal{L}_1$ on level 1 and compared it to the theoretically expected size (see columns three and four of Table 3).

Furthermore, we counted the number of successful iterations where the error vector $\mathbf{e}$ was found (see column five of table 3). The results approximately match the theoretically predicted success probability of at least $\frac{1}{2}$ for ColumnMatch. The slight discrepancy is due to the small value of p.

For the sake of completeness, we also give the time per iteration as well as the number of repetitions P^{-1} that would be needed for the complete Decode algorithm (see columns six and seven).

We would like to stress that the main goal of our implementation was to test the validity of the heuristic assumption, that $\mathbf{Q}$ behaves as a random matrix.

Table 3. Experimental results for the ColumnMatch algorithm

$[n, k, \omega]$	$[p, \ell_1, \ell_2]$	$\mid \mathcal{L}_1 \mid$ theo.	$\mid \mathcal{L}_1 \mid$ exp.	success prob.	time (ms)	P^{-1}
$[255, 135, 15]$	$[4, 11, 2]$	1369	1369.1	43.6%	11	$2^{8.12}$
$[511, 259, 28]$	$[4, 13, 2]$	4692.25	4692.08	44.2%	44	$2^{17.96}$
$[1024, 524, 50]$	$[4, 16, 2]$	18360	18360.4	43.3%	207	$2^{38.74}$

We did not put effort in optimizing our code for speed by e.g. using clever data structures or hash tables as it was done in [3]. We leave it has an open problem to implement an efficient version of our algorithm for determining the cut-off point with other variants of information set decoding, such as Stern, FS-ISD or Ball-collision decoding.

Acknowledgements. The authors would like to thank Antoine Joux for useful discussions and Jannik Pewny for carrying out the experiments.

References

1. Alon, N., Spencer, J.: The Probabilistic Method. Wiley (2008)
2. Berlekamp, E., McEliece, R., van Tilborg, H.: On the inherent intractability of certain coding problems (Corresp.). IEEE Transactions on Information Theory 24(3), 384–386 (1978)
3. Bernstein, D.J., Lange, T., Peters, C.: Attacking and Defending the McEliece Cryptosystem. In: Buchmann, J., Ding, J. (eds.) PQCrypto 2008. LNCS, vol. 5299, pp. 31–46. Springer, Heidelberg (2008)
4. Bernstein, D.J., Lange, T., Peters, C.: Smaller Decoding Exponents: Ball-Collision Decoding. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 743–760. Springer, Heidelberg (2011)
5. Finiasz, M., Sendrier, N.: Security Bounds for the Design of Code-Based Cryptosystems. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 88–105. Springer, Heidelberg (2009)
6. Guruswami, V.: Introduction to Coding Theory. Lecture Notes (2010)
7. Hopper, N.J., Blum, M.: Secure Human Identification Protocols. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 52–66. Springer, Heidelberg (2001)
8. Howgrave-Graham, N., Joux, A.: New Generic Algorithms for Hard Knapsacks. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 235–256. Springer, Heidelberg (2010)
9. Lee, P.J., Brickell, E.F.: An Observation on the Security of McEliece's Public-Key Cryptosystem. In: Günther, C.G. (ed.) EUROCRYPT 1988. LNCS, vol. 330, pp. 275–280. Springer, Heidelberg (1988)
10. Leon, J.S.: A probabilistic algorithm for computing minimum weights of large error-correcting codes. IEEE Transactions on Information Theory 34(5), 1354 (1988)
11. McEliece, R.J.: A public-key cryptosystem based on algebraic coding theory. In: DSN Progress Report 42–44 (1978)
12. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: Gabow, H.N., Fagin, R. (eds.) STOC, pp. 84–93. ACM (2005)
13. Stern, J.: A method for finding codewords of small weight. In: Wolfmann, J., Cohen, G.D. (eds.) Coding Theory 1988. LNCS, vol. 388, pp. 106–113. Springer, Heidelberg (1989)