A new signature scheme based on $(U|U+V)$ codes

Thomas Debris-Alazard, Nicolas Sendrier, and Jean-Pierre Tillich [⋆]

Inria, EPI SECRET
2 rue Simone Iff, Paris 75012, France
{thomas.debris,nicolas.sendrier,jean-pierre.tillich}@inria.fr

Abstract. We present here a new code-based digital signature scheme. This scheme uses $(U|U+V)$ codes, where both U and V are random. We prove that the scheme achieves *existential unforgeability under adaptive chosen message attacks* under two assumptions from coding theory, both strongly related to the hardness of decoding in a random linear code. The proof imposes a uniform distribution on the produced signatures, we show that this distribution is easily and efficiently achieved by rejection sampling. Our scheme is efficient to produce and verify signatures. For a (classical) security of 128 bits, the signature size is less than one kilobyte and the public key size a bit smaller than 2 megabytes. This gives the first practical signature scheme based on binary codes which comes with a security proof and which scales well with the security parameter: it can be shown that if one wants a security level of 2^λ, then signature size is of order $O(\lambda)$, public key size is of size $O(\lambda^2)$, signature generation cost is of order $O(\lambda^3)$, whereas signature verification cost is of order $O(\lambda^2)$.

Keywords: code-based cryptography, digital signature scheme, decoding algorithm.

1 Introduction

Code-based signature schemes. It is a long standing open problem to build an efficient and secure signature scheme based on the hardness of decoding a linear code which could compete in all respects with DSA or RSA. Such schemes could indeed give a quantum resistant signature for replacing in practice the aforementioned signature schemes that are well known to be broken by quantum computers. A first partial answer to this question was given in [CFS01]. It consisted in adapting the Niederreiter scheme [Nie86] for this purpose. This requires a linear code for which there exists an efficient decoding algorithm for a non-negligible set of inputs. This means that if $\mathbf{H}$ is an $r \times n$ parity-check matrix of the code, there exists for a non-negligible set of elements $\mathbf{s}$ in $\{0,1\}^r$ an efficient way to find a word $\mathbf{e}$ in $\{0,1\}^n$ of smallest Hamming weight such that $\mathbf{H}\mathbf{e}^T = \mathbf{s}^T$. In such a case, we say that $\mathbf{s}$, which is generally called a syndrome in the literature, can be decoded. To sign a message $\mathbf{m}$, a hash function $\mathscr{H}$ is used to produce a sequence $\mathbf{s}_0, \dots, \mathbf{s}_\ell$ of elements of $\{0,1\}^r$. For instance $\mathbf{s}_0 = \mathscr{H}(\mathbf{m})$ and $\mathbf{s}_i = \mathscr{H}(\mathbf{s}_0, i)$ for $i > 0$. The first $\mathbf{s}_i$ that can be decoded defines the signature of $\mathbf{m}$ as the word $\mathbf{e}$ of smallest Hamming weight such that

$$\mathbf{H}\mathbf{e}^T = \mathbf{s}_i^T.$$

The CFS signature scheme. The authors of [CFS01] noticed that very high rate Goppa codes are able to fulfill this task, and their scheme can indeed be considered as the first step towards a solution of the aforementioned problem. Moreover they gave a security proof of their scheme relying only on the assumption that two problems were hard, namely (i) decoding a generic linear code and (ii) distinguishing a Goppa code from a random linear code with the same parameters. However, afterwards it was realized that the parameters proposed in [CFS01] can be attacked by an unpublished attack of Bleichenbacher. The significant increase of parameters needed to thwart the Bleichenbacher attack was fixed by a slight variation [Fin10]. However, this modified scheme is not able to fix two other worrying drawbacks of the CFS scheme, namely

[⋆] This work was supported in part by the Commission of the European Communities through the Horizon 2020 program under project number 645622 PQCRYPTO.

(i) a lack of security proof in light of the distinguisher of high rate Goppa codes found in [FGO$^+$11] (see also [FGO$^+$13] for more details) which shows that the hypotheses used in [CFS01] to give a security proof of the signature scheme were not met,

(ii) poor scaling of the parameters when security has to be increased. [CFS01] is namely based on t-error correcting Goppa codes of length 2^m. The complexity $S = 2^\lambda$ of the best attack is $S = 2^{tm/2}$, the public key is of size $K = tm2^m$, obtaining a signature needs about $t!t^2m^3$ operations. Here the factorial $t!$ term accounts for the number of syndromes $\mathbf{s}_i$ that have to be computed before finding one of them that can be decoded. Keeping a reasonable signature cost requires that we fix t to be small (say smaller than 12). In this case, the security parameter S is only polynomial in the key size $K : S \approx K^{t/2}$.

Other code-based signature schemes. Other signature schemes based on codes were also given in the literature such as for instance the KKS scheme [KKS97, KKS05] or its variants [BMS11, GS12]. But they can be considered at best to be one-time signature schemes in the light of the attack given in [COV07] and great care has to be taken to choose the parameters of these schemes as shown by [OT11] which broke all the parameters proposed in [KKS97, KKS05, BMS11].

There has been some revival of the CFS strategy [CFS01], by choosing other code families. The new code families that were used are LDGM codes in [BBC$^+$13], i.e. codes with a Low Density Generator Matrix, or (essentially) convolutional codes [GSJB14]. There are still some doubts that there is a way to choose the parameters of the scheme [GSJB14] in order to avoid the attack [LT13] on the McEliece cryptosystem based on convolutional codes [LJ12] and the LDGM scheme was broken in [PT16].

A last possibility is to use the Fiat-Shamir heuristic to turn a zero-knowledge authentication scheme into a signature scheme. When based on the Stern authentication scheme [Ste93] this gives a code-based signature scheme. However this approach leads to really large signature sizes (of the order of thousand(s) of bits). This represents a complete picture of code-based signature schemes based on the Hamming metric. There has been some recent progress in this area for another metric, namely the rank metric [GRSZ14]. We provide a detailed discussion about this point in the conclusion. It should be added that the Hamming metric is particularly attractive for designing code-based schemes due to the fact that the difficulty of decoding for this metric is a topic that has been thoroughly studied over the years (for more details see the discussion that appears later in the introduction).

Moving from error-correcting codes to lossy source codes. It can be argued that the main problem with the CFS approach is to find a family of linear codes that are at the same time (i) indistinguishable from a random code and (ii) that have a non-negligible fraction of syndromes that can be decoded. There are not so many codes for which (ii) can be achieved and this is probably too much to ask for. However if we relax a little bit what we ask for the code, namely just a code such that the equation (in $\mathbf{e}$)

$$\mathbf{H}\mathbf{e}^T = \mathbf{s}^T \tag{1}$$

admits for most of the $\mathbf{s}$'s a solution $\mathbf{e}$ of small enough weight, then there are many more codes that are able to fulfill this task. This kind of codes are not used in error-correction but can be found in lossy source coding or source distortion theory where the problem is to find codes with an associated decoding algorithm which can approximate *any* word of the ambient space by a close enough codeword. In the case of linear codes, this means a code and a associated decoding algorithm that can find for any syndrome $\mathbf{s}$ a vector $\mathbf{e}$ of small enough weight satisfying (1) where $\mathbf{H}$ is a parity-check matrix of the code.

Solving (1) is the basic problem upon which all code-based cryptography relies. This problem has been studied for a long time and despite many efforts on this issue [Pra62, Ste88, Dum91, Bar97, MMT11, BJMM12, MO15, DAT17] the best algorithms for solving this problem [BJMM12, MO15] are exponential in the weight w of $\mathbf{e}$ as long as $w = (1 - \epsilon)r/2$ for any $\epsilon > 0$. Furthermore when w is sublinear in n, the exponent of the best known algorithms has not changed [CTS16] since the Prange algorithm [Pra62] dating back to the early sixties. Moreover, it seems very difficult to lower this exponent by a multiplicative factor smaller than $\frac{1}{2}$ in the quantum computation model as illustrated by [Ber10, KT17].

The exponent $c(\epsilon)r$ is maximal when w is required to be equal to the Gilbert-Varshamov bound, namely $w = nh^{-1}(r/n)$ where $h(x) \stackrel{\triangle}{=} -x\log_2(x) - (1-x)\log_2(1-x)$ and $h^{-1}(x)$ is the inverse function defined for x in $[0,1]$ and ranging over $[0, \frac{1}{2}]$. It represents the largest weight w for which we can typically expect that Problem (1) has a unique solution. The amount $c(\epsilon)$ goes slowly to 0 as ϵ goes to zero. There is a very simple algorithm solving this problem, namely the Prange algorithm [Pra62] which is of polynomial complexity when $\epsilon = 0$. In the context of this work, we are precisely in a case where Equation (1) admits numerous solutions.

Our contribution: a new signature scheme based on $(U|U+V)$ codes. Convolutional codes or LDGM codes are codes which come with a decoding algorithm which is polynomial for weights below $r/2$. They could theoretically be used in this context, since they provide an advantage over standard linear codes in this context. Permuting the coordinates and publishing a random parity-check matrix of the resulting code might seem enough to yield a secure signature scheme. However in the light of the attacks [LJ12, PT16], it seems very difficult to propose parameters which avoid the attacks given in these papers. Polar codes are also codes which fulfill (ii) but the attack [BCD$^+$16] found on the McEliece scheme based on polar codes [SK14] would also apply in this setting. We are instead introducing a new class of codes in this context namely $(U|U+V)$ codes. A $(U|U+V)$ code is just a way of building of code of length n when we have two codes U and V of length $n/2$. It consists in

$$(U|U+V) \stackrel{\triangle}{=} \{(\mathbf{u}|\mathbf{u}+\mathbf{v}) : \mathbf{u} \in U, \mathbf{v} \in V\}.$$

Generalized $(U|U+V)$ codes have already been proposed in the cryptographic context for building a McEliece encryption scheme [MCT16]. However, there it was suggested to take U and V to be codes that have an efficient decoding algorithm (this is namely mandatory in the encryption context). In the signature context, when we just need to find a small enough solution of (1) this is not needed. In our case, we can afford to choose *random* codes for U and V. It turns out that if we choose U and V random with the right choice of the dimension of U and V, then a suitable use of the Prange algorithm on the code U and the code V provides an advantage in this setting. It namely allows to solve (1) for weights w that are significantly below $r/2$, that is in the range of weights for which the best decoding algorithms are exponential.

Moreover, by tweaking a little bit the output of the Prange algorithm in our case and performing an appropriate rejection sampling, it turns out that the signatures are indistinguishable from a random word of weight w. This allows to give a security proof of our signature scheme which relies on two problems:

P1: Solving the decoding problem (1) when w is sufficiently below $r/2$
P2: Distinguishing a permuted $(U|U+V)$ code from a random code of the same length and dimension and this even when U and V are themselves random codes.

Problem P1 is the problem upon which all code-based cryptography relies. Here we are in a case where there are multiple solutions of (1) and the adversary may produce any number of instances of (1) with the same matrix $\mathbf{H}$ and various syndromes $\mathbf{s}$ and is interested in solving only one of them. This relates to the, so called, Decoding One Out of Many (DOOM) problem. This problem was first considered in [JJ02]. It was shown there how to modify slightly the known algorithms for decoding a linear code in order to solve this modified problem. This modification was later analyzed in [Sen11]. The parameters of the known algorithms for solving (1) can be easily adapted to this scenario where we have to decode simultaneously multiple instances which all have multiple solutions.

Problem P2 might seem to be an ad-hoc problem. However we are really in a situation where the resulting permuted $(U|U+V)$ code is actually very close to a random code. The only different behavior that can be found seems to be in the weight distribution for small weights. In this case, the permuted $(U|U+V)$ code has some codewords of a weight slightly smaller than the minimum distance of a random code of the same length and dimension. It is very tempting to conjecture that the best algorithms for solving Problem P2 come from detecting such codewords. This approach can be easily thwarted by choosing the parameters of the scheme in such a way that the best algorithms for solving this task are of prohibitive complexity. Notice that the best algorithms that we have for detecting such codewords are in essence precisely the generic algorithms for solving the first problem, namely Problem P1. In some sense, it seems that we might rely on the very same problem, namely solving Problem P1, even if our proof technique does not show this.

All in all, this gives the first practical signature scheme based on binary codes which comes with a security proof and which scales well with the parameters: it can be shown that if one wants a security level of 2^λ, then signature size is of order $O(\lambda)$, public key size is of order $O(\lambda^2)$, signature generation is of order $O(\lambda^3)$, whereas signature verification is of order $O(\lambda^2)$.

Organization of the paper. The paper is organized as follows, we present our scheme in §2, in §3 we prove it is secure under *existential unforgeability under an adaptive chosen message attack* (EUF-CMA), in relation with this proof we respectively examine in §4, §5, and §6, how to produce uniformly distributed signatures as well as the best message and key attacks. Finally we give some set of parameters on par with the security reduction and with the current state-of-the-art for decoding techniques.

Notation. We provide here some notation that will be used throughout the paper. Vectors will be written with bold letters (such as $\mathbf{e}$) and uppercase bold letters are used to denote matrices (such as $\mathbf{H}$). Vectors are in row notation. Let $\mathbf{x}$ and $\mathbf{y}$ be two vectors, we will write $(\mathbf{x}|\mathbf{y})$ to denote their concatenation. We also denote for a subset I of positions of the vector $\mathbf{x} = (x_i)_{1 \leq i \leq n}$ by $\mathbf{x}_I$ the vector whose components are those of $\mathbf{x}$ which are indexed by I, i.e.

$$\mathbf{x}_I = (x_i)_{i \in I}.$$

We define the support of $\mathbf{x}$ as

$$\mathrm{Supp}(\mathbf{x}) \stackrel{\triangle}{=} \{i \in \{1, \cdots, n\} \text{ such that } x_i \neq 0\}$$

The Hamming weight of $\mathbf{x}$ is denoted by $|\mathbf{x}|$. By some abuse of notation, we will use the same notation to denote the size of a finite set: $|S|$ stands for the size of the finite set S. It will be clear from the context whether $|\mathbf{x}|$ means the Hamming weight or the size of a finite set. Note that

$$|\mathbf{x}| = |\,\mathrm{Supp}(\mathbf{x})|.$$

The notation $x \stackrel{\triangle}{=} y$ means that x is defined to be equal to y. We denote by $\mathbb{F}_2^n$ the set of binary vectors of length n and S_w is its subset of words of weight w. Let S be a finite set, then $x \hookleftarrow S$ means that x is assigned to be a random element chosen uniformly at random in S. For a distribution $\mathscr{D}$ we write $\xi \sim \mathscr{D}$ to indicate that the random variable ξ is chosen according to $\mathscr{D}$. We denote the uniform distribution on S_w by $\mathscr{U}_w$.

A binary linear code $\mathscr{C}$ of length n and dimension k is a subspace of $\mathbb{F}_2^n$ of dimension k and is usually defined by a parity-check matrix $\mathbf{H}$ of size $r \times n$ as

$$\mathscr{C} = \left\{\mathbf{x} \in \{0,1\}^n : \mathbf{H}\mathbf{x}^T = \mathbf{0}\right\}.$$

When $\mathbf{H}$ is of full rank (which is usually the case) we have $r = n - k$. The rate of this code (that we denote by R) is defined as $R \stackrel{\triangle}{=} \frac{k}{n}$.

2 The $(U, U+V)$-signature Scheme

2.1 The general scheme $\mathscr{S}_{code}$

Our scheme can be viewed as a probabilistic version of the full domain hash (FDH) signature scheme as defined in [BR96] which is similar to the probabilistic signature scheme introduced in [Cor02] except that we replace RSA with a trapdoor function based upon the hardness of Problem P1. Let $\mathscr{C}$ be a binary linear code of length n defined by a parity-check matrix $\mathbf{H}$. The one way function f_w we consider is given by

$$\begin{array}{rcl} f_w : \mathscr{C} \times S_w & \longrightarrow & \mathbb{F}_2^n \\ (\mathbf{c}, \mathbf{e}) & \longmapsto & \mathbf{c} + \mathbf{e} \end{array}$$

Inverting this function on an input $\mathbf{y}$ amounts to solve Problem P1 with $\mathbf{s}^T \triangleq \mathbf{H}\mathbf{y}^T$ since we have that $\mathbf{s}^T = \mathbf{H}\mathbf{y}^T = \mathbf{H}(\mathbf{c}^T + \mathbf{e}^T) = \mathbf{H}\mathbf{e}^T$. Solving Problem P1 yields in this case $\mathbf{e}$ and we recover $\mathbf{c}$ with $\mathbf{c} = \mathbf{y} - \mathbf{e}$.

We are ready now to give the general scheme we consider. We assume that we have a family $\mathscr{F}$ of codes of length n for which we can invert for all codes $\mathscr{C}$ in the family the associated function f_w. Then we pick a code $\mathscr{C}_{\text{sec}}$ uniformly at random in $\mathscr{F}$. We then choose an $n \times n$ permutation matrix $\mathbf{P}$. This is the *secret key* (sk) of the signature scheme. The *public code* is defined as

$$\mathscr{C}_{\text{pub}} = \{\mathbf{c}\mathbf{P} : \mathbf{c} \in \mathscr{C}_{\text{sec}}\}.$$

We compute a parity-check matrix $\mathbf{H}_{\text{pub}}$ of $\mathscr{C}_{\text{pub}}$. This is the *public key* (pk) of the signature scheme. We also select a cryptographic hash function $\mathscr{H} : \{0,1\}^* \to \mathbb{F}_2^n$ and a parameter λ_0 for the random salt. The algorithms $\mathtt{Sgn}^{\text{sk}}$ and $\mathtt{Vrfy}^{\text{pk}}$ are defined as follows

```
Sgn^sk($\mathbf{m}$):
  $\mathbf{r} \hookleftarrow \{0,1\}^{\lambda_0}$
  $\mathbf{y} \leftarrow \mathscr{H}(\mathbf{m}|\mathbf{r})$
  $(\mathbf{c}, \mathbf{e}) \leftarrow f_w^{-1}(\mathbf{y}\mathbf{P}^{-1})\mathbf{P}$
  return($\mathbf{e}, \mathbf{r}$)
```

```
Vrfy^pk($\mathbf{m}, (\mathbf{e}', \mathbf{r})$):
  $\mathbf{y} \leftarrow \mathscr{H}(\mathbf{m}|\mathbf{r})$
  $w_0 \leftarrow |\mathbf{e}'|$
  if $\mathbf{H}_{\text{pub}}(\mathbf{y}^T + \mathbf{e}'^T) = \mathbf{0}$ and $w_0 = w$ return 1
  else return 0
```

The correction of the verification step (i.e. that the pair $(\mathbf{e}\mathbf{P}, \mathbf{r})$ passes the verification step) follows from the fact that by definition of f_w^{-1} we have $\mathbf{c}+\mathbf{e} = \mathbf{y}\mathbf{P}^{-1}$ with $\mathbf{c} \in \mathscr{C}_{\text{sec}}$ and $|\mathbf{e}| = w$. This implies that $\mathbf{c}\mathbf{P}+\mathbf{e}\mathbf{P} = \mathbf{y}$ and therefore $\mathbf{y}+\mathbf{e}\mathbf{P}$ is in $\mathscr{C}_{\text{pub}}$ which in turn implies that $\mathbf{H}_{\text{pub}}(\mathbf{y}^T+\mathbf{P}^T\mathbf{e}^T) = \mathbf{0}$. We also have $|\mathbf{e}\mathbf{P}| = |\mathbf{e}| = w$. Finding a valid signature pair clearly amounts to solve Problem P1 for the code $\mathscr{C}_{\text{pub}}$ and the syndrome $\mathbf{s}^T = \mathbf{H}_{\text{pub}}\mathbf{y}^T$ as explained before.

2.2 Source-distortion codes and decoders

Source-distortion theory is a branch of information theory which deals with obtaining a family of codes $\mathscr{F}$ of the smallest possible dimension which can be used in our setting (i.e. for which we can invert f_w). Recall that a linear code is a vector space and the dimension of the code is defined as the dimension of this vector space. For a linear code specified by a full rank parity-check matrix of size $r \times n$, the dimension k of the code is equal to $n - r$. It is essential to have the smallest possible dimension in our cryptographic application, since this makes the associated problem P1 harder: the smaller $n-r$ is, the bigger r is and the further away w can be from $r/2$ (where solving P1 becomes easy). This kind of codes is used for performing lossy coding of a source. Indeed assume that we can perform this task, then this means that we can find for every binary word $\mathbf{y}$ a codeword $\mathbf{c}$ which is at most at distance w from it. The word $\mathbf{y}$ is compressed with a compact description of $\mathbf{c}$. Since the code is dimension $n-r$ we just need $n-r$ bits to store a description of $\mathbf{c}$. We have replaced here $\mathbf{y}$ with a word which is not too far away from it. Of course, the smaller $n-r$ is, the smaller the compression rate $\frac{n-r}{n}$ is. There is some loss by replacing $\mathbf{y}$ by $\mathbf{c}$ since we are in general close to $\mathbf{y}$ but not equal to it. Finding a close codeword $\mathbf{c}$ of a given word $\mathbf{y}$

is equivalent to find a low weight "error" $\mathbf{e}$ such that $\mathbf{y}+\mathbf{e}$ is in the code. For our purpose it will be more convenient to adopt the error viewpoint than the codeword viewpoint. To stress the similarity with error-correction we will call the function which associates to $\mathbf{y}$ such an $\mathbf{e}$ a source distortion decoder.

Definition 1 (Source Distortion Decoder). *Let n, $k \leq n$ be integers and let $\mathscr{F}$ be a family of binary linear codes of length n and dimension k. A source distortion decoder for $\mathscr{F}$ is a probabilistic algorithm φ:*

$$\begin{aligned} \varphi : \mathscr{F} \times \mathbb{F}_2^n &\longrightarrow \mathbb{F}_2^n \\ (\mathscr{C}, \mathbf{y}) &\longmapsto \mathbf{e} \end{aligned}$$

such that $\mathbf{y}+\mathbf{e} \in \mathscr{C}$. When the weight of the error is fixed, we call it a decoder of fixed distortion w and we denote it by φ_w. We say that the distortion w is achievable if there exists a family of codes with a decoder of fixed distortion w.

This discussion raises a first question: for given n and k, what is the minimal distortion w which is achievable? We know from Shannon's rate-distortion theorem that the minimal w is given by the Gilbert-Varshamov bound $d_{\mathrm{GV}}(n,k)$ which follows:

Definition 2 (Gilbert-Varshamov's bound). *For given integers n and k such that $k \leq n$, the Gilbert-Varshamov bound $d_{\mathrm{GV}}(n,k)$ is given by:*

$$d_{\mathrm{GV}}(n,k) \overset{\triangle}{=} n h^{-1}\left(1 - k/n\right)$$

where h denotes the binary entropy: $h(x) = -x \log_2 x - (1-x)\log_2(1-x)$ and h^{-1} its inverse defined on $[0,1]$ and whose range is $[0, \frac{1}{2}]$.

Achieving distortion $w = (n-k)/2$ with the Prange technique. The study of random codes shows that they achieve this source-distortion bound in average. Nevertheless we do not know for them an efficient source-distortion algorithm. However, as the following proposition shows, it is not the case when the distortion w is higher. When $w = (n-k)/2$ there is a very efficient decoder using the Prange technique [Pra62] for decoding. To explain it consider a linear code $\mathscr{C}$ of dimension k, length n and a parity-check matrix $\mathbf{H}$ for it. We want to find for a given $\mathbf{y} \in \mathbb{F}_2^n$ an error $\mathbf{e}$ of low weight such that $\mathbf{y}-\mathbf{e}$ is in $\mathscr{C}$. This means that we should have $\mathbf{H}\mathbf{y}^T = \mathbf{H}\mathbf{e}^T$. $\mathbf{H}$ is a full-rank matrix and it therefore contains an invertible submatrix $\mathbf{A}$ of size $(n-k)\times(n-k)$. We choose a set of positions I of size $n-k$ for which $\mathbf{H}$ restricted to these positions is a full rank matrix. For simplicity assume that this matrix is in the first $n-k$ positions: $\mathbf{H} = \big(\mathbf{A}|\mathbf{B}\big)$. We look for an $\mathbf{e}$ of the form $\mathbf{e} = (\mathbf{e}'|\mathbf{0}_k)$ where $\mathbf{e}' \in \mathbb{F}_2^{n-k}$. We should therefore have $\mathbf{H}\mathbf{y}^T = \mathbf{H}\mathbf{e}^T = \mathbf{A}\mathbf{e}'^T$, that is $\mathbf{e}'^T = \mathbf{A}^{-1}\mathbf{H}\mathbf{y}^T$. The expected weight of $\mathbf{e}'$ is $\frac{n-k}{2}$ and it easy to check that by randomly picking a random set I of size $n-k$ we have to check a polynomial number of them until finding an $\mathbf{e}'^T$ of weight exactly $(n-k)/2$.

Notation. We denote by $\varphi^{\mathrm{Prange}}_{(n-k)/2}$ this fixed distortion decoder and by $\varphi^{\mathrm{Prange}}$ the decoder which picks a random subset until finding one for which $\mathbf{H}$ restricted to the columns corresponding to I is invertible and computes $\mathbf{e}'$ as explained above. $\varphi^{\mathrm{Prange}}$ does not necessarily output an error of weight $(n-k)/2$.

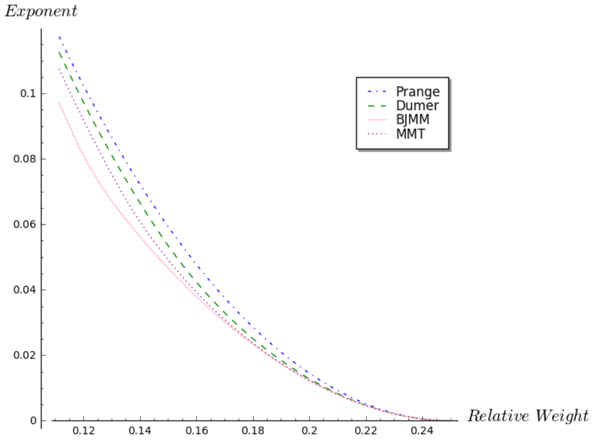
From the previous discussion we easily obtain

Proposition 1 (Generic Source Distortion Decoder). *Let integers n, $k \leq n$ and $\mathscr{F}$ the family of random codes of length n and dimension k. $\varphi^{\mathrm{Prange}}_{(n-k)/2}$ works on average polynomial time.*

When we consider in general the family of linear codes we speak about generic source-distortion decoders as there is no structure, except linearity of the code. In contrast to the distortion $(n-k)/2$, the only algorithms we know for linear codes for smaller values of w are all exponential in the distortion. This is illustrated by Figure 1 where we give the exponents (divided by the length n)

of the complexity in base 2 as a function of the distance, for the fixed rate $R = k/n = 0.5$, of the best generic fixed-w source distortion decoders. As we see, the normalized exponent is 0 for distortion $(n-k)/2$ and the difficulty increases as w approaches the Gilbert-Varshamov bound (which is equal approximately to $0.11n$ in this case).

Fig. 1: Normalized exponents in base 2 of the best generic fixed-w source distortion decoders.



Source distortion theory has found over the years several families of codes with an efficient source-distortion algorithm which achieves asymptotically the Gilbert-Varshamov source-distortion bound, one of the most prominent ones being probably the Arikan polar codes [Arı09] (see [Kor09]). The naive way would be to build our signature on such a code-family and hoping that permuting the code positions and publishing a random parity-check matrix of the permuted code would destroy all the structure used for decoding. All known families of codes used in this context have low weight codewords and this can be used to mount an attack. We will proceed differently here and introduce in this setting the $(U|U+V)$ codes mentioned in the introduction. The point is that they (i) have very little structure, (ii) have a very simple source-distortion decoder which is more performant than the generic source decoder, (iii) they do not suffer from low weight codewords as was the case with the aforementioned families. It will be useful to recall here that

Definition 3 ($(U|U+V)$-Codes). *Let U, V be linear binary codes of length $n/2$ and dimension k_U, k_V. We define the subset of $\mathbb{F}_2^n$:*

$$(U|U+V) \stackrel{\triangle}{=} \{(\mathbf{u}|\mathbf{u}+\mathbf{v}) \text{ such that } \mathbf{u} \in U \text{ and } \mathbf{v} \in V\}$$

which is a linear code of length n and dimension $k = k_U + k_V$. The resulting code is of minimum distance $\min(2d_U, d_V)$ where d_U is the minimum distance of U and d_V is the minimum distance of V.

We are now going to present a source-distortion for a $(U|U+V)$ code. The following definitions will be useful here.

Definition 4 (Punctured code). *Consider a code $\mathscr{C}$ of length n. The punctured code $\mathrm{Punc}_I(\mathscr{C})$ in a set of positions $I \subset \{1,\dots,n\}$ is a code of length $n - |I|$ defined as*

$$\underset{I}{\mathrm{Punc}}(\mathscr{C}) \stackrel{\triangle}{=} \{\mathbf{c}_{\bar{I}} : \mathbf{c} \in \mathscr{C}\}$$

where $\bar{I} = \{1,\dots,n\} \setminus I$. If $\mathbf{x} = (x_i)_{1 \le i \le n}$ is a binary vector, $\mathrm{Punc}_{\mathbf{x}}$ *will denote* $\mathrm{Punc}_{\mathrm{Supp}(\mathbf{x})}$.

We will also need the dual notion of extending a partial codeword. Assume that for a given set of positions J and a linear code $\mathscr{C}$ of length n we have that $\dim \mathscr{C} = \dim(\text{Punc}_J \mathscr{C})$, then if for $\mathbf{x} \in \mathbb{F}_2^{n-|J|}$ there exists a codeword $\mathbf{c}$ in $\mathscr{C}$ such that $\mathbf{c}_{\overline{J}} = \mathbf{x}$, such a codeword is necessarily unique and we say that $\mathbf{c}$ is the completion of $\mathbf{x}$ with respect to J and write $\mathbf{c} = \text{Comp}_J^{\mathscr{C}}(\mathbf{x})$. For a random code $\mathscr{C}$, a set J has this property with probability $1 - O(2^{-n+|J|+\dim \mathscr{C}})$. Moreover, for a vector $\mathbf{x} = (x_i)_{1 \leq i \leq n}$, $\text{Comp}_{\mathbf{x}}^{\mathscr{C}}$ will denote $\text{Comp}_{\text{Supp}(\mathbf{x})}^{\mathscr{C}}$.

We can use the generic source distortion decoder of Proposition 1 for source distortion decoding a $(U|U+V)$ code. Assume that we have a $(U|U+V)$ code of length n and a word $\mathbf{y} = (\mathbf{y}_1|\mathbf{y}_2)$ that we want to decode where the $\mathbf{y}_i$'s belong to $\mathbb{F}_2^{n/2}$. We assume that the dimension of U is k_U and that the dimension of V is k_V. Let us first try to recover a "good" $\mathbf{v}$. Such a $\mathbf{v}$ should clearly be a good approximation for $\mathbf{y}_1 + \mathbf{y}_2$. We use the (Prange) generic source decoder for V to find such a $\mathbf{v}$. That is $\mathbf{e}_V = \varphi^{\text{Prange}}_{(n/2-k_V)/2}(V, \mathbf{y}_1 + \mathbf{y}_2)$. Here $\mathbf{v} = \mathbf{e}_V + \mathbf{y}_1 + \mathbf{y}_2$. We then subtract $\mathbf{v}$ to $\mathbf{y}_2$ to get $\mathbf{y}'_2 = \mathbf{y}_2 + \mathbf{v} = \mathbf{y}_1 + \mathbf{e}_V$. We are now left with the task of approximating $\mathbf{y}' = (\mathbf{y}_1|\mathbf{y}_1 + \mathbf{e}_V)$ with a word of the form $(\mathbf{u}|\mathbf{u})$ where $\mathbf{u} \in U$. We clearly have to get the best approximation of $\mathbf{y}_1$ in the positions which do not correspond to the support of $\mathbf{e}_V$. In other words we compute $\mathbf{e}'_U = \varphi^{\text{Prange}}_{(n/2-k_U-|\mathbf{e}_V|)/2}(\text{Punc}_{\mathbf{e}_V}(U), \text{Punc}_{\mathbf{e}_V}(\mathbf{y}_1))$. This can be done as long as $(n/2 - k_U - |\mathbf{e}_V|)/2 \geq 0$, that is $n/2 \geq k_U + |\mathbf{e}_V|$ which amounts to $n/2 \geq k_U + (n/2 - k_V)/2$, that is $n/2 \geq 2k_U - k_V$. The word $\mathbf{u}$ we are looking is such that $\text{Punc}_{\mathbf{e}_V}(\mathbf{u}) = \text{Punc}_{\mathbf{e}_V}(\mathbf{y}_1) + \mathbf{e}'_U$. We compute the righthand term and we perform its completion to recover $\mathbf{u}$. This procedure is given in Algorithm 1.

Algorithm 1 UV**-sddV**1 : $(U|U+V)$**−Source Distortion Decoder**

```
Parameter: a $(U|U+V)$ code of length $n$ and dimension $k = k_U + k_V$
Input: $(\mathbf{y}_1|\mathbf{y}_2)$ with $\mathbf{y}_i \in \mathbb{F}_2^{n/2}$
Output: $\mathbf{e} \in \mathbb{F}_2^n$
Assumes: $2k_U - k_V \leq n/2$.
$\mathbf{e}_V \leftarrow \varphi^{\text{Prange}}_{(n/2-k_V)/2}(V, \mathbf{y}_1 + \mathbf{y}_2)$
$\mathbf{e}'_U \leftarrow \varphi^{\text{Prange}}_{(n/2-k_U-|\mathbf{e}_V|)/2}(\text{Punc}_{\mathbf{e}_V}(U), \text{Punc}_{\mathbf{e}_V}(\mathbf{y}_1))$
$\mathbf{u} \leftarrow \text{Comp}^U_{\mathbf{e}_V}(\text{Punc}_{\mathbf{e}_V}(\mathbf{y}_1) + \mathbf{e}'_U)$
$\mathbf{e}_U \leftarrow \mathbf{y}_1 + \mathbf{u}$
return $(\mathbf{e}_U|\mathbf{e}_U + \mathbf{e}_V)$
```

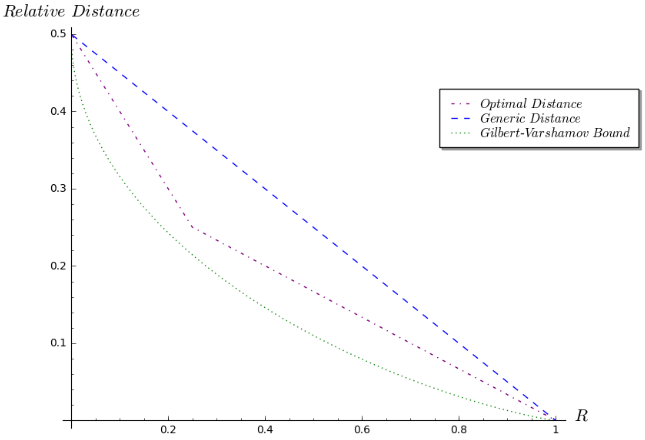
Proposition 2. *The algorithm* UV-**sddV**1 *is a fixed-*$(n/2 - k_U)$ *source distortion decoder which works in polynomial average-time when* $2k_U - k_V \leq n/2$.

Proof. Let $\mathbf{e}''_U$ be the element of $\mathbb{F}_2^{n/2}$ which coincides with $\mathbf{e}'_U$ on the positions outside the support of $\mathbf{e}_V$ and is zero elsewhere. We may write $\mathbf{e}_U = \mathbf{e}''_U + \mathbf{e}'''_U$ where the support of $\mathbf{e}'''_U$ is included in the support of $\mathbf{e}_V$.

$$\begin{aligned}
|(\mathbf{e}_U|\mathbf{e}_U + \mathbf{e}_V)| &= |\mathbf{e}_U| + |\mathbf{e}_U + \mathbf{e}_V| \\
&= |\mathbf{e}''_U| + |\mathbf{e}'''_U| + |\mathbf{e}''_U + \mathbf{e}'''_U + \mathbf{e}_V| \\
&= |\mathbf{e}''_U| + |\mathbf{e}'''_U| + |\mathbf{e}''_U| + |\mathbf{e}'''_U + \mathbf{e}_V| \\
&= 2|\mathbf{e}''_U| + |\mathbf{e}'''_U| - |\mathbf{e}'''_U| + |\mathbf{e}_V| \\
&= 2|\mathbf{e}''_U| + |\mathbf{e}_V| \\
&= n/2 - k_U - |\mathbf{e}_V| + |\mathbf{e}_V| \\
&= n/2 - k_U
\end{aligned}$$

We can now choose the parameters k_U and k_V in order to minimize the distortion $n/2 - k_U$ for a fixed dimension $k = k_U + k_V$ of the code. Figure 2 compares the distance that we obtain with this algorithm to $(n-k)/2$ which corresponds to what is achieved by the generic decoder and to the optimal distance (Gilbert-Varshamov bound) where R denotes the rate of the code. As we see there is a non-negligible gain. Nevertheless, UV-**sddV**1 approximates to a fixed distance in each step of its execution which leads to correlations between some bits that can be used to recover the structure of the secret key. In order to fix this problem and as it is asked in our proof of security, we will present a modified version of UV-**sddV**1 in §4 which uses a rejection sampling method to simulate uniform outputs. This comes at the price of slightly increasing the weight of the error output by the decoder.

Fig. 2: Comparison of the Optimal Signature Distance, the Gilbert-Varshamov Bound and Generic Distance



3 Security Proof

We give in this section a security proof of the signature scheme $\mathscr{S}_{\text{code}}$. This proof is in the spirit of the security proof of the FDH signatures in the random oracle model (see [BR93]). However as our scheme is probabilistic we were also inspired by the proof of [Cor02]. Our main result is to reduce the security to two major problems in code-based cryptography.

3.1 Basic tools

Basic definitions.

Definition 5 (Negligible Function). *A function $f : \mathbb{N} \to \mathbb{R}$ is negligible if for every positive polynomial p:*

$$\exists N,\ \forall n > N,\ |f(n)| \leq \frac{1}{p(n)}.$$

Definition 6 (Statistical Distance). *Let $\mathscr{D}^0$ and $\mathscr{D}^1$ be two discrete probability distributions over a same discrete space $\mathscr{E}$. Their statistical distance is defined as:*

$$\rho(\mathscr{D}^0, \mathscr{D}^1) \stackrel{\triangle}{=} \sum_{x \in \mathscr{E}} |\mathscr{D}^0(x) - \mathscr{D}^1(x)|.$$

We will need the following well known property for the statistical distance which can be easily proved by induction

Proposition 3. *Let $(\mathscr{D}_1^0, \dots, \mathscr{D}_n^0)$ and $(\mathscr{D}_1^1, \dots, \mathscr{D}_n^1)$ be two n-tuples of discrete probability distributions where $\mathscr{D}_i^0$ and $\mathscr{D}_i^1$ are distributed over a same space $\mathscr{E}_i$. For $a \in \{0,1\}$, let us denote by $\mathscr{D}_1^a \otimes \dots \otimes \mathscr{D}_n^a$ the product probability distribution of $\mathscr{D}_1^a, \dots, \mathscr{D}_n^a$, that is $\mathscr{D}_1^a \otimes \dots \otimes \mathscr{D}_n^a(x_1, \dots, x_n) = \mathscr{D}_1^a(x_1) \dots \mathscr{D}_n^a(x_n)$ with $x_i \in \mathscr{E}_i$ for $i \in \{1, \dots, n\}$. In such a case we have*

$$\rho\left(\mathscr{D}_1^0 \otimes \dots \otimes \mathscr{D}_n^0, \mathscr{D}_1^1 \otimes \dots \otimes \mathscr{D}_n^1\right) \leq \sum_{i=1}^n \rho(\mathscr{D}_i^0, \mathscr{D}_i^1).$$

We are now going to define the concept of distinguisher between two distributions and to relate it with the statistical distance.

Definition 7 (Distinguisher). *A distinguisher between two distributions $\mathscr{D}^0$ and $\mathscr{D}^1$ over the same space $\mathscr{E}$ is a randomized algorithm which takes as input an element of $\mathscr{E}$ that follows the distribution $\mathscr{D}^0$ or $\mathscr{D}^1$ and outputs $b \in \{0,1\}$.*

A distinguisher $\mathscr{A}$ between $\mathscr{D}^0$ and $\mathscr{D}^1$ is characterized by its advantage:

$$Adv^{\mathscr{D}^0,\mathscr{D}^1}(\mathscr{A}) \overset{\triangle}{=} \mathbb{P}_{\xi \sim \mathscr{D}^0}\left(\mathscr{A}(\xi) \text{ outputs } 1\right) - \mathbb{P}_{\xi \sim \mathscr{D}^1}\left(\mathscr{A}(\xi) \text{ outputs } 1\right)$$

where $\mathbb{P}_{\xi \sim \mathscr{D}^i}\left(\mathscr{A}(\xi) \text{ outputs } 1\right)$ is the probability that $\mathscr{A}(\xi)$ outputs 1 when its inputs are picked according to the distribution $\mathscr{D}^i$ and for each executions its internal coins are picked uniformly at random. We call this quantity the advantage of $\mathscr{A}$ against $\mathscr{D}^0$ and $\mathscr{D}^1$.

We are now able to define the computational distance between two distributions.

Definition 8 (Computational Distance and Indistinguishability). *The computational distance between two distributions $\mathscr{D}^0$ and $\mathscr{D}^1$ in time t is:*

$$\rho_c\left(\mathscr{D}^0, \mathscr{D}^1\right)(t) \overset{\triangle}{=} \max_{|\mathscr{A}| \leq t} \left\{ Adv^{\mathscr{D}^0,\mathscr{D}^1}(\mathscr{A}) \right\}$$

where $|\mathscr{A}|$ denotes the running time of $\mathscr{A}$ on its inputs.

The ensembles $\mathscr{D}^0 = (\mathscr{D}_n^0)$ and $\mathscr{D}^1 = (\mathscr{D}_n^1)$ are computationally indistinguishable in time (t_n) if their computational distance in time (t_n) is negligible in n.

In other words, the computational distance is the best advantage that any adversary could get in bounded time. It is well known that statistical distance is greater than computational distance as the following theorem claims.

Theorem 1. *Let $\mathscr{D}^0$ and $\mathscr{D}^1$ be two distributions, then $\rho\left(\mathscr{D}^0, \mathscr{D}^1\right)$ is the best advantage that any adversary could get, even with unbounded time:*

$$\forall t, \quad \rho_c\left(\mathscr{D}^0, \mathscr{D}^1\right)(t) \leq \rho\left(\mathscr{D}^0, \mathscr{D}^1\right).$$

Digital signature and games. Let us recall the concept of signature schemes, the security model that will be considered in the following and to recall in this context the paradigm of games in which we give a security proof of our scheme.

Definition 9 (Signature Scheme). *A signature scheme $\mathscr{S}$ is a triple of algorithms* `Gen`, `Sgn`, *and* `Vrfy` *which are defined as:*

- *The key generation algorithm* `Gen` *is a probabilistic algorithm which given 1^λ, where λ is the security parameter, outputs a pair of matching public and private keys (pk, sk);*
- *The signing algorithm is probabilistic and takes as input a message $\mathbf{m} \in \{0,1\}^*$ to be signed and returns a signature $\mathbf{s} = \mathtt{Sgn}^{\mathrm{sk}}(\mathbf{m})$;*
- *The verification algorithm takes as input a message $\mathbf{m}$ and a signature $\mathbf{s}$. It returns $\mathtt{Vrfy}^{\mathrm{pk}}(\mathbf{m}, \mathbf{s})$ which is 1 if the signature is accepted and 0 otherwise. It is required that $\mathtt{Vrfy}^{\mathrm{pk}}(\mathbf{m}, \mathbf{s}) = 1$ if $\mathbf{s} = \mathtt{Sgn}^{\mathrm{sk}}(\mathbf{m})$.*

For this kind of scheme one of the strongest security notion is *existential unforgeability under an adaptive chosen message attack* (EUF-CMA). In other words, the adversary has access to any signatures of its choice and its goal is to produce a valid forgery. A valid forgery is a message/signature pair $(\mathbf{m},\mathbf{s})$ such that $\texttt{Vrfy}^{\text{pk}}(\mathbf{m},\mathbf{s})=1$ whereas the signature of $\mathbf{m}$ has never been requested by the forger. More precisely, the following definition gives the EUF-CMA security of a signature scheme:

Definition 10 (EUF-CMA Security). *Let $\mathscr{S}$ be a signature scheme.*
A forger $\mathscr{A}$ is a $(t,q_{hash},q_{sign},\varepsilon)$-adversary in EUF-CMA against $\mathscr{S}$ if after at most q_{hash} queries to the hash oracle, q_{sign} signatures queries and t working time, it outputs a valid forgery with probability at least ε. We define the EUF-CMA success probability against $\mathscr{S}$ as:

$$Succ_{\mathscr{S}}^{EUF\text{-}CMA}(t,q_{hash},q_{sign}) \stackrel{\triangle}{=} \max\left(\varepsilon \,|\, it\ exists\ a\ (t,q_{hash},q_{sign},\varepsilon)\text{-}adversary\right).$$

The signature scheme $\mathscr{S}$ is said to be (t,q_{hash},q_{sign})-secure in EUF-CMA if the above success probability is a negligible function of the security parameter λ.

The game associated to our code-based signature scheme. The modern approach to prove the security of cryptographic schemes is to relate the security of its primitives to well-known problems that are believed to be hard by proving that breaking the cryptographic primitives provides a mean to break one of these hard problems. In our case, the security of the signature scheme is defined as a game with an adversary that has access to hash and sign oracles. It will be helpful here to be more formal and to define more precisely the games we will consider. They are games between two players, an *adversary* and a *challenger*. In a game G, the challenger executes three kind of procedures:

- an initialization procedure `Initialize` which is called once at the beginning the game.
- oracle procedures which can be requested at the will of the adversary. In our case, there will be two, `Hash` and `Sign`. The adversary $\mathscr{A}$ which is an algorithm may call `Hash` at most q_{hash} times and `Sign` at most q_{sign} times.
- a final procedure `Finalize` which is executed once $\mathscr{A}$ has terminated. The output of $\mathscr{A}$ is given as input to this procedure.

The output of the game G, which is denoted $G(\mathscr{A})$, is the output of the finalization procedure (which is a bit $b\in\{0,1\}$). The game G with $\mathscr{A}$ is said to be successful if $G(\mathscr{A})=1$. The standard approach for obtaining a security proof in a certain model is to construct a sequence of games such that the success of the first game with an adversary $\mathscr{A}$ is exactly the success against the model of security, the difference of the probability of success between two consecutive games is negligible until the final game where the probability of success is the probability for $\mathscr{A}$ to break one of the problems which is supposed to be hard. In this way, no adversary can break the claim of security with non-negligible success unless it breaks one of the problems that are supposed to be hard.

Definition 11 (challenger procedures in the EUF-CMA Game). *The challenger procedures for the EUF-CMA Game corresponding to $\mathscr{S}_{code}$ are defined as:*

proc Initialize(λ)	proc Hash$(\mathbf{m},\mathbf{r})$	proc Sign$(\mathbf{m})$	proc Finalize$(\mathbf{m},\mathbf{e},\mathbf{r})$
$(\mathbf{H}_{\text{pub}},\mathbf{P},\lambda_0)\leftarrow \texttt{Gen}(1^\lambda)$	return $\mathscr{H}(\mathbf{m}\vert\mathbf{r})$	$\mathbf{r}\hookleftarrow\{0,1\}^{\lambda_0}$	$\mathbf{y}\leftarrow \texttt{Hash}(\mathbf{m},\mathbf{r})$
return $\mathbf{H}_{\text{pub}}$		$\mathbf{y}\leftarrow$ Hash$(\mathbf{m},\mathbf{r})$	$\mathbf{c}\leftarrow\mathbf{y}+\mathbf{e}$
		$(\mathbf{c},\mathbf{e})\leftarrow f_w^{-1}(\mathbf{y}\mathbf{P}^{-1})\mathbf{P}$	return $\mathbf{H}_{\text{pub}}\mathbf{c}^T=\mathbf{0}\wedge\vert\mathbf{e}\vert=w$
		return $(\mathbf{e},\mathbf{r})$	

3.2 Code-Based Signatures

We introduce in this subsection the code-based problems that will be used in the security proof. The first is Decoding One Out of Many (DOOM) which was first considered in [JJ02] and later analyzed in [Sen11]. We will come back to the best known algorithms to solve this problem as a function of the distance w in §5.

Problem 1 (DOOM – Decoding One Out of Many).
Instance: $\mathbf{H} \in \mathbb{F}_2^{(n-k)\times n}$; $\mathbf{y}_1, \cdots, \mathbf{y}_q \in \mathbb{F}_2^n$; $w \in \{0, \cdots, n\}$
Output: $\mathbf{e} \in \mathbb{F}_2^n$ of Hamming weight w such that for some $i \in \{1, \cdots, q\}$, $\mathbf{He}^T = \mathbf{Hy}_i^T$

We will denote by $(\mathbf{H}, \mathbf{y}_1, \cdots, \mathbf{y}_q) \hookleftarrow \text{DOOM}(n, k, w)$ a randomly chosen instance of this problem.

Definition 12 (One-Wayness of DOOM). *We define the success of an algorithm $\mathscr{A}$ against* DOOM *with the parameters n, k, w as:*

$$Succ^{n,k,w}_{\text{DOOM}}(\mathscr{A}) = \mathbb{P}\big(\mathscr{A}(\mathbf{H}, \mathbf{y}_1, \cdots, \mathbf{y}_q) \textit{ solution } \textit{of } \text{DOOM} \mid (\mathbf{H}, \mathbf{y}_1, \cdots, \mathbf{y}_q) \hookleftarrow \text{DOOM}(n, k, w)\big).$$

where the probability is taken over uniformly instances and internal coins of $\mathscr{A}$.

The computational success in time t of breaking DOOM *with the parameters n, k, w is then defined as:*

$$Succ^{n,k,w}_{\text{DOOM}}(t) = \max_{|\mathscr{A}|\leq t} \left\{ Succ^{n,k,w}_{\text{DOOM}}(\mathscr{A}) \right\}$$

Another problem will appear in the security proof: distinguish random codes from a code drawn uniformly at random in the family used in the signature scheme. Let $\mathscr{F}$ be the family of public codes of length n and dimension k that we use in $\mathscr{S}_{\text{code}}$. We define the uniform distribution over $\mathscr{F}$ as $\mathscr{D}^{\mathscr{F}}$. On the other hand $\mathscr{D}^{\mathscr{R}}$ will denote the uniform distribution over the family of all binary linear codes of length n and dimension k.

Remark 1. In $\mathscr{S}_{\text{code}}$, the family $\mathscr{F}$ is $\{\mathscr{C}_{pub}\}$ where:

$$\mathscr{C}_{pub} = \{\mathbf{cP} : \mathbf{c} \in \mathscr{C}_{sec}\}$$

with $\mathbf{P}$ a permutation matrix and $\mathscr{C}_{sec}$ a $(U|U+V)$-code of length n and dimension k.

Let us denote by $\mathscr{D}_w$ the distribution of the w-source distortion decoder outputs which is used to sign and by $\mathscr{U}_w$ the uniform distribution over S_w (which is the set of words of weight w in $\mathbb{F}_2^n$).

3.3 EUF-CMA Security Proof

This Subsection is devoted to our main theorem and its proof

Theorem 2 (Security Reduction). *Let q_{hash} (resp. q_{sign}) be the number of queries to the hash (resp. signing) oracle. We assume that $\lambda_0 = \lambda + 2\log_2(q_{sign})$ where λ is the security parameter of the signature scheme. We have for all w, time t :*

$$Succ^{EUF-CMA}_{\mathscr{S}_{code}}(t, q_{hash}, q_{sign}) \leq 4 Succ^{n,k,w}_{\text{DOOM}}(t_c) + \frac{1}{2} q_{hash} \sqrt{\frac{2^{n-k}}{\binom{n}{w}}} + q_{sign}\rho\left(\mathscr{D}_w, \mathscr{U}_w\right) + 3\rho_c\left(\mathscr{D}^{\mathscr{R}}, \mathscr{D}^{\mathscr{F}}\right)(t_c)$$

where $t_c = t + O\left(q_{hash} \cdot n^2\right)$

Remark 2. In the paradigm of code-based signatures we have w greater than the Gilbert-Varshamov bound, which gives $2^{n-k} \ll \binom{n}{w}$.

Proof. Let $\mathscr{A}$ be a $(t, q_{\text{sign}}, q_{\text{hash}}, \varepsilon)$-adversary in the EUF-CMA model against $\mathscr{S}_{\text{code}}$. We will write $\mathbb{P}(S_i)$ to denote the probability of success for $\mathscr{A}$ of game G_i. Let $q = q_{\text{hash}} - q_{\text{sign}}$. If `proc Hash` is called several times with the same arguments, it returns the same output, we do not handle this in the games to keep the pseudo code simple.

Game 0 is the EUF-CMA game for $\mathscr{S}_{\text{code}}$.

Game 1 is identical to Game 0 unless the following failure event F occurs: there is a collision in a signature query (*i.e.* two signatures queries for a same message $\mathbf{m}$ lead to the same salt $\mathbf{r}$). By using the difference lemma (see for instance [Sho04, Lemma 1]) we get:

$$\mathbb{P}(S_0) \leq \mathbb{P}(S_1) + \mathbb{P}(F).$$

The following lemma (see A.2 for a proof) shows that in our case as $\lambda_0 = \lambda + 2\log_2(q_{\text{sign}})$, the probability of the event F is negligible.

Lemma 1. *For $\lambda_0 = \lambda + 2\log_2(q_{sign})$ we have:*

$$\mathbb{P}(F) \leq \frac{1}{2^\lambda}.$$

Game 2 is modified from Game 1 as follows:

```
proc Initialize($\lambda$)
$(\mathbf{H}_{\text{pub}}, \mathbf{P}, \lambda_0) \leftarrow$ Gen($1^\lambda$)
$\mathbf{H}_0 \hookleftarrow \mathbb{F}_2^{(n-k)\times n}$
$(\mathbf{y}_1, \dots, \mathbf{y}_q) \hookleftarrow (\mathbb{F}_2^n)^q$
$j \leftarrow 0$
return $\mathbf{H}_{\text{pub}}$
```

```
proc Hash($\mathbf{m}, \mathbf{r}$)
if $L_\mathbf{m}$.contains($\mathbf{r}$)
  $(\mathbf{c}, E_{\mathbf{m},\mathbf{r}}) \hookleftarrow \mathscr{C}_{pub} \times S_w$
  return $\mathbf{c} + E_{\mathbf{m},\mathbf{r}}$
else
  $j \leftarrow j + 1$
  return $\mathbf{y}_j$
```

```
proc Sign($\mathbf{m}$)
$\mathbf{r} \leftarrow L_\mathbf{m}$.next()
$\mathbf{y} \leftarrow$ Hash($\mathbf{m}, \mathbf{r}$)
$(\mathbf{c}, \mathbf{e}) \leftarrow f_w^{-1}(\mathbf{y}\mathbf{P}^{-1})\mathbf{P}$
return $(\mathbf{e}, \mathbf{r})$
```

To each message $\mathbf{m}$ we associate a list $L_\mathbf{m}$ containing q_{sign} random elements of $\mathbb{F}_2^{\lambda_0}$. It is constructed the first time it is needed. The call $L_\mathbf{m}$.contains($\mathbf{r}$) returns true if and only if $\mathbf{r} \in L_\mathbf{m}$. The call $L_\mathbf{m}$.next() returns elements of $L_\mathbf{m}$ sequentially. The list is large enough to satisfy all queries. The Hash procedure now creates the list $L_\mathbf{m}$ if needed, then, if $\mathbf{r} \in L_\mathbf{m}$ it returns $\mathbf{y} = \mathbf{c} + \mathbf{e}$ with $\mathbf{e} \hookleftarrow S_w$ and $\mathbf{c} \hookleftarrow \mathscr{C}_{pub}$. This leads to a valid signature $(\mathbf{e}, \mathbf{r})$ for $\mathbf{m}$. The error value is stored in $E_{\mathbf{m},\mathbf{r}}$. If $\mathbf{r} \notin L_\mathbf{m}$ it outputs one of $\mathbf{y}_j$ of the instance $(\mathbf{H}_0, \mathbf{y}_1, \dots, \mathbf{y}_q)$ of the DOOM problem. The Sign procedure is unchanged, except for $\mathbf{r}$ which is now taken in $L_\mathbf{m}$. The global index j is set to 0 in proc Initialize.

We can relate this game to the previous one through the following lemma.

Lemma 2.

$$\mathbb{P}(S_1) \leq \mathbb{P}(S_2) + \frac{1}{2} q_{hash} \sqrt{\frac{2^{n-k}}{\binom{n}{w}}} + 2\rho_c \left(\mathscr{D}^{\mathscr{F}}, \mathscr{D}^{\mathscr{R}}\right) \left(t \cdot O(n^2)\right)$$

The proof of this lemma is given in Appendix A.3 and relies among other things on the leftover hash lemma (see [BDK$^+$11]). We show in appendix how to emulate the lists $L_\mathbf{m}$ in such a way that list operations cost, including its construction, is at most linear in the security parameter λ. Since $\lambda \leq n$, it follows that the cost to a call to proc Hash cannot exceed $O(n^2)$ and the running time of the challenger is $t_c = t + O\left(q_{\text{hash}} \cdot n^2\right)$.

Game 3 differs from Game 2 by changing in proc Sign the call "$(\mathbf{c}, \mathbf{e}) \leftarrow f_w^{-1}(\mathbf{y}\mathbf{P}^{-1})\mathbf{P}$" by "$\mathbf{e} \leftarrow E_{\mathbf{m},\mathbf{r}}$". Any signature $(\mathbf{e}, \mathbf{r})$ produced by proc Sign is valid. The error $\mathbf{e}$ is drawn according to the uniform distribution $\mathscr{U}_w$ while previously it was drawn according to the source distortion decoder distribution, that is $\mathscr{D}_w$. By using Proposition 3 it follows that

$$\mathbb{P}(S_2) \leq \mathbb{P}(S_3) + q_{\text{sign}} \rho\left(\mathscr{U}_w, \mathscr{D}_w\right).$$

The running time of the challenger cannot increase by more than $O(q_{\text{sign}} \cdot \lambda)$ and thus we still have $t_c = t + O\left(q_{\text{hash}} \cdot n^2\right)$.

Game 4 is the game where we replace the public matrix $\mathbf{H}_{\text{pub}}$ by $\mathbf{H}_0$. In other words the Initialize procedure is now:

```
proc Initialize($\lambda$)
$(\mathbf{H}_{\texttt{pub}}, \mathbf{P}, \lambda_0) \leftarrow \texttt{Gen}(1^\lambda)$
$(\mathbf{H}_0, \mathbf{y}_1, \cdots, \mathbf{y}_q) \hookleftarrow \text{DOOM}(n, k, w)$
$j \leftarrow 0$
$\mathbf{H}_{\texttt{pub}} \leftarrow \mathbf{H}_0$
return $\mathbf{H}_{\texttt{pub}}$
```

In this way we will force the adversary to build a solution of the DOOM problem. Here if a difference is detected between games it gives a distinguisher between the distribution $\mathscr{D}^{\mathscr{R}}$ and $\mathscr{D}^{\mathscr{F}}$:

$$\mathbb{P}(S_3) \leq \mathbb{P}(S_4) + \rho_c\left(\mathscr{D}^{\mathscr{F}}, \mathscr{D}^{\mathscr{R}}\right)(t_c).$$

Game 5 differs in the finalize procedure.

```
proc Finalize($\mathbf{m}, \mathbf{e}, \mathbf{r}$)
$\mathbf{y} \leftarrow \texttt{Hash}(\mathbf{m}, \mathbf{r})$
$b \leftarrow \mathbf{H}_{\texttt{pub}}(\mathbf{y} + \mathbf{e})^T = 0 \wedge |\mathbf{e}| = w$
return $b \wedge \neg L_{\mathbf{m}}$.contains($\mathbf{r}$)
```

We assume the forger outputs a valid signature $(\mathbf{e}, \mathbf{r})$ for the message $\mathbf{m}$. The probability of success of Game 5 is the probability of the event "$S_4 \wedge (\mathbf{r} \notin L_{\mathbf{m}})$".

If the forgery is valid, the message $\mathbf{m}$ has never been queried by `Sign`, and the adversary never had access to any element of the list $L_{\mathbf{m}}$. This way, the two events are independent and we get:

$$\mathbb{P}(S_5) = (1 - 2^{-\lambda_0})^{q_{\text{sign}}} \mathbb{P}(S_4).$$

As we assumed $\lambda_0 = \lambda + 2\log_2(q_{\text{sign}}) \geq \log_2(q_{\text{sign}})$, we have:

$$\left(1 - 2^{-\lambda_0}\right)^{q_{\text{sign}}} \geq \left(1 - \frac{1}{q_{\text{sign}}}\right)^{q_{\text{sign}}} \geq \frac{1}{4}.$$

Therefore

$$\mathbb{P}(S_5) \geq \frac{1}{4}\mathbb{P}(S_4). \tag{2}$$

The probability $\mathbb{P}(S_5)$ is then exactly the probability for $\mathscr{A}$ to output $\mathbf{e} \in S_w$ such that $\mathbf{H}_0(\mathbf{y}_j + \mathbf{e})^T = \mathbf{0}$ for some j which gives

$$\mathbb{P}(S_5) \leq Succ_{\text{DOOM}}^{n,k,w}(t_c). \tag{3}$$

(2) together with (3) imply that

$$\mathbb{P}(S_4) \leq 4Succ_{\text{DOOM}}^{n,k,w}(t_c).$$

This concludes the proof of Theorem 2 by combining this together with all the bounds obtained for each of the previous games.

4 Achieving the Uniform Distribution of the Outputs

4.1 Rejection Sampling Method

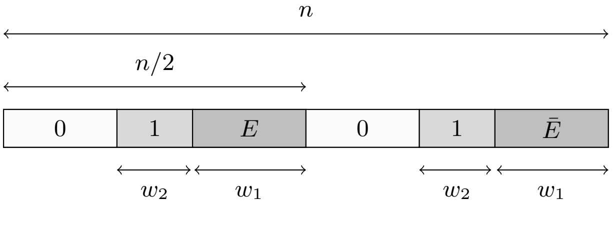
In our security proof, we use the fact that the distribution of the outputs of the $(U|U+V)$ decoder is close to the uniform distribution on the words of weight w. We will show how to modify a little bit the decoder by performing some moderate rejection sampling in order to meet this property. Note that ensuring such a property is actually not only desirable for the security proof, it is also more or less necessary since there is an easy way to attack the signature when it is based on the decoder UV-**sddV**1. Indeed, it is readily verified that with this decoder the probability $\mathbb{P}(e_i = 1, e_j = 1)$ we have on the output $\mathbf{e}$ of the decoder for certain i and j is larger than the same probability for a random word $\mathbf{e}$ of weight w. The pairs (i, j) which have this property correspond to the image by the permutation $\mathbf{P}$ of pairs of the form $(x, x+n/2)$ or $(x+n/2, x)$. In other words, signatures leak information in this case and this can be used to recover completely the permuted $(U|U+V)$ structure of the code.

It is insightful to consider more precisely the shape of the errors that are output by the $(U|U+V)$-source distortion decoder. This is given by Figure 3 where w_1 is the distortion achieved in the first step of the algorithm and w_2 is the second one. Algorithm UV-**sddV**1 has this behavior with $w_1 = (n/2 - k_V)/2$ and $w_2 = (n/2 - k_U - w_1)/2 = (n/2 - k_U - (n/2 - k_V)/2)/2 = n/8 - k_U/2 + k_V/4$. However we would have the same shape by applying to V and to U *any* source-distortion decoder. The weight w_1 would be in this case the distortion achieved for the decoder for V and w_2 would be the distortion achieved by the decoder for the punctured version of U with respect to the support of error output by the decoder for V. It turns out that by allowing more freedom in the distortion achieved by the decoders of U and V we are able to achieve a uniform distribution on the words of weight w. The corresponding procedure is given in Algorithm 2. Roughly speaking the idea is to use the Prange decoder but without fixing the weight for the V-decoder and performing some rejection sampling in order to achieve the uniform distribution over the outputs.

Algorithm 2 UV-**sddV**2 : $(U|U+V)-$**Source Distortion Decoder**

```
Parameter: a $(U|U+V)$ code of length $n$
Inputs: · $(\mathbf{y}_1|\mathbf{y}_2)$ with $\mathbf{y}_i \in \mathbb{F}_2^{n/2}$
· no-rejection probability vector $\mathbf{x} = (x_i)_{0 \le i \le n-k_V} \in [0,1]^{n-k_V}$
Output: $\mathbf{e} \in \mathbb{F}_2^n$ with $|\mathbf{e}| = w$.
Assumes: $2k_U - k_V \le n/2$.
repeat
    $\mathbf{e}_V \leftarrow \varphi(V, \mathbf{y}_1 + \mathbf{y}_2)$
    $p \hookleftarrow [0,1]$
until $|\mathbf{e}_V| \le w$, $w - |\mathbf{e}_V| \equiv 0 \pmod 2$ and $p \le x_{|\mathbf{e}_V|}$
$\mathbf{e}'_U \leftarrow \varphi_{(w-|\mathbf{e}_V|)/2}(\mathrm{Punc}_{\mathbf{e}_V}(U), \mathrm{Punc}_{\mathbf{e}_V}(\mathbf{y}_1))$
$\mathbf{u} \leftarrow \mathrm{Comp}^U_{\mathbf{e}_V}(\mathrm{Punc}_{\mathbf{e}_V}(\mathbf{y}_1) + \mathbf{e}'_U)$
$\mathbf{e}_U \leftarrow \mathbf{y}_1 + \mathbf{u}$
return $(\mathbf{e}_U|\mathbf{e}_U + \mathbf{e}_V)$
```

Fig. 3: Shape of the outputs of UV-**sddV**1



To explain the rejection method let us introduce some notation.

Notation. Let $\mathbf{e} \in \mathbb{F}_2^n$ and $w \in \{0, \cdots, n\}$,

$$w_1(\mathbf{e}) \stackrel{\triangle}{=} \#\left\{i \in \{1, \cdots, n/2\} \ : \ e_i \neq e_{i+n/2}\right\},$$
$$w_2(\mathbf{e}) \stackrel{\triangle}{=} \#\left\{i \in \{1, \cdots, n/2\} \ : \ e_i = e_{i+n/2} = 1\right\}.$$

The strategy for rejection sampling is that the distortion w_1 should follow the same law as $w_1(\mathbf{e})$ when $\mathbf{e}$ is drawn uniformly at random from the word of weight w. Note that we clearly have for any word $\mathbf{e} \in S_w$

$$w = w_1(\mathbf{e}) + 2w_2(\mathbf{e}).$$

It will be helpful to bring in now the following quantities. Let us first assume that $\mathbf{e}$ is chosen uniformly at random in S_w. For $i \in \{1, 2\}$ we define the quantities

$$p_i^u(j) \stackrel{\triangle}{=} \mathbb{P}_{\mathbf{e} \hookleftarrow S_w}\left(w_i(\mathbf{e}) = j\right).$$

It is straightforward to check that

Proposition 4 (Distribution of w_1 and w_2). *For all i in $\{0, \dots, w\}$ such that $w \equiv i \pmod 2$*

$$p_2^u\left(\frac{w-i}{2}\right) = p_1^u(i) = 2^i\,\frac{\binom{n/2}{(w-i)/2}\binom{n/2-(w-i)/2}{i}}{\binom{n}{w}}$$

and for other choices of i, $p_1(i)$ and $p_2(i)$ are equal to 0.

Similarly to the case of the uniform distribution, we define the following probability distributions for the outputs of the source-distortion decoder and for i in $\{1, 2\}$:

$$p_i^{sdd}(j) \stackrel{\triangle}{=} \mathbb{P}_{\mathbf{e}}\left(w_i(\mathbf{e}) = j\right)$$

where $\mathbf{e}$ is now the output of Algorithm 2. A simple formula for these probability distributions is given in Proposition 5 (see in the appendix §B.1 for a proof). These distributions can be derived from the rejection probabilities and the weight distribution of the source distortion decoder as follows:

Proposition 5. *Let $p(i) \stackrel{\triangle}{=} \mathbb{P}_{\mathbf{y},\theta}(|\varphi(V, \mathbf{y})| = i)$. If two executions of φ are independent, then for all i in $\{0, \dots, w\}$ such that $w - i \equiv 0 \pmod 2$ we have*

$$p_2^{sdd}\left(\frac{w-i}{2}\right) = p_1^{sdd}(i) = \frac{x_i\, p(i)}{p_w^1} \tag{4}$$

where

$$p_w^1 \stackrel{\triangle}{=} \sum_{\substack{0 \le j \le w \\ j \equiv w \pmod 2}} x_j\, p(j)$$

and $p_1^{sdd}(i) = 0$ for other choices of i.

Remark 3. We stress here that two executions of φ have to be independent. It is naturally the case for the Prange algorithm if we choose independently the different information sets.

The following definition will turn out be be useful.

Definition 13. *Let θ denote the internal coin used in the probabilistic algorithm φ. We say that φ behaves uniformly for a code $\mathscr{C}$ if $\mathbb{P}_{\mathbf{y},\theta}\left(\mathbf{e} = \varphi(\mathscr{C}, \mathbf{y})\right)$ only depends on the weight $|\mathbf{e}|$.*

In the case of a decoder φ that behaves uniformly, the no-rejection vector $\mathbf{x}$ can be chosen so that the output of Algorithm 2 is uniformly distributed as shown by the following proposition.

Proposition 6. *If the source decoder φ used in Algorithm 2 behaves uniformly for V and uniformly for $\text{Punc}_{\mathbf{e}_V}(U)$ for all error patterns $\mathbf{e}_V$ obtained as $\mathbf{e}_V = \varphi(V, \mathbf{y}_1 + \mathbf{y}_2)$, we have:*

$$\rho\left(\mathscr{D}_w, \mathscr{U}_w\right) = \rho\left(p_1^{sdd}, p_1^u\right)$$

The output of Algorithm 2 is the uniform distribution over S_w if in addition two executions of φ are independent and the no-rejection probability vector $\mathbf{x}$ is chosen for any i in $\{0, \dots, w\}$ as

$$x_i = \frac{1}{M_{rs}}\frac{p_1^u(i)}{p(i)} \text{ if } w \equiv i \pmod 2$$

and 0 *otherwise with* $M_{rs} \stackrel{\triangle}{=} \sup\limits_{\substack{0 \le i \le w \\ i \equiv w \pmod 2}} \frac{p_1^u(i)}{p(i)}$.

4.2 Application to the Prange source distortion decoder.

The Prange source decoder (defined in §2.2) is extremely close to behave uniformly for almost all linear codes. To keep this paper within a reasonable length we just provide here how the relevant distribution $p(i)$ is computed.

Proposition 7 (Weight Distribution of the Prange Algorithm). *Let* $p(i) = \sum_{\mathbf{e}:|\mathbf{e}|=i} \mathbb{P}_{\mathbf{y},\theta}\left(\mathbf{e} = \varphi^{\text{Prange}}(\mathscr{C}, \mathbf{y})\right)$*. For all* $w, k, n \in \mathbb{N}$ *with* $k \leq n$*,* $w \leq n-k$*, all codes* $\mathscr{C}$ *of length* n *and dimension* k*, we have:*

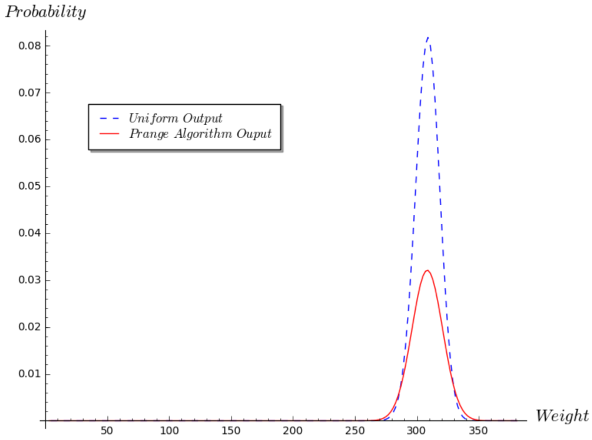
$$p(w) = \frac{\binom{n-k}{w}}{2^{n-k}}$$

By using Proposition 6 with this distribution p we can set up the no-rejection probability vector $\mathbf{x}$ in Algorithm 2. To have an efficient algorithm it is essential that the parameter M_{rs} is as small as possible (it is namely readily verified that the average number of calls in Algorithm 2 to $\varphi^{\text{Prange}}(V, \mathbf{y}_1 + \mathbf{y}_2)$ is M_{rs}). Let $\mathbf{e}$ be an error of weight w chosen uniformly at random. This average number of calls can be chosen to be small by imposing that the distributions of $w_1(e)$ and $|\varphi(V, \mathbf{y})|$ to have the same expectation. The expectation of $w_1(e)$ is approximately $w\left(1 - \frac{w}{n}\right)$ and the expectation of $|\varphi(V, \mathbf{y})|$ is $(n/2 - k_V)/2$. We choose therefore k_V such that

$$(n/2 - k_V)/2 \approx w\left(1 - \frac{w}{n}\right).$$

Thanks to this property, k_V is chosen to "align" both distributions and in this way M_{rs} is small. We can find in Figure 4 an example of the distributions p_1^u and p corresponding to the choice $n = 2000$, $k_V = 383$, $w = 381$. In this case, $M_{\text{rs}} = 2.54$.

Fig. 4: Comparison of Prange algorithm's output distribution with uniform outputs



This rejection sampling method comes at the price of slightly increasing the weight the decoder can output. Indeed, given parameters n, k what set of parameters do we have to choose? First, in order to avoid a prohibitive cost for the Prange algorithm we have to set k_U and w such that w corresponds to the expected weight of the Prange decoder. This implies:

$$n/2 - k_U = w \tag{5}$$

Moreover as explained above, the rejection sampling comes with a minimal cost when the following constraint is met:

$$k_V = n/2 - 2w\left(1 - \frac{w}{n}\right) \tag{6}$$

Finally we also have $k = k_U + k_V$. Combining this with (6) and (5) leads to the following system of equations:

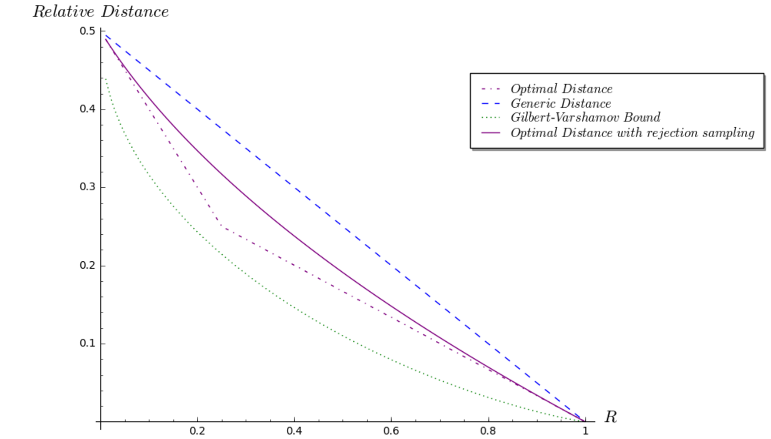
$$\begin{cases} n/2 - k_U = w \\ k_V = n/2 - 2w\left(1 - \frac{w}{n}\right) \end{cases} \iff \begin{cases} k_V = w - n/2 + k \\ 2w^2/n - 3w + n - k = 0 \end{cases}$$

Therefore $\omega \triangleq w/n$ and $R \triangleq k/n$ have to verify

$$2\omega^2 - 3\omega + 1 - R = 0 \tag{7}$$

Once k and n are fixed, ω is chosen and we deduce k_U and k_V thanks to (5) and (6). Figure 5 gives ω as a function of R. For instance for $R = 0.5$ we have $\omega = 0.1909$.

Fig. 5: Comparison of the Optimal Signature Distortion with or without the Rejection Sampling Method, the Gilbert-Varshamov Bound and the Generic Distortion



5 Best Known Algorithms for Solving the DOOM Decoding Problem

We consider here the best known techniques for solving Problem 1, namely decoding one out of many, i.e. the so called DOOM problem. This problem is a variation of the classical decoding problem where for a binary $[n, k]$ code and a vector $\mathbf{y}$ of length n we have to find an error vector $\mathbf{e}$ of Hamming weight w such that $\mathbf{y} + \mathbf{e}$ is in the code. This problem is equivalent to syndrome decoding

Problem 2 (SD – Syndrome Decoding).

Instance:	$\mathbf{H} \in \mathbb{F}_2^{(n-k)\times n}$, $\mathbf{s} \in \mathbb{F}_2^{n-k}$, w integer
Output:	$\mathbf{e} \in \mathbb{F}_2^n$ such that $\lvert\mathbf{e}\rvert = w$ and $\mathbf{H}\mathbf{e}^T = \mathbf{s}^T$

Information set decoding is the best known technique to solve the syndrome decoding problem, it can be traced back to Prange [Pra62]. It has been improved in [Ste88, Dum91] by introducing a birthday paradox. The current state-of-the-art can be found in [MMT11, BJMM12, MO15]. Existing literature usually assumes that there is a unique solution to the problem. This is true in particular when w is smaller than the Gilbert-Varshamov bound (see Definition 2). When w is larger, as it is the case here, we speak of source distortion decoding even though the problem statement is the same. The expected number of solutions grows as $M = \binom{n}{w}/2^{n-k}$ and the cost analysis must be adapted to take that into account. There is a second specificity in the current study which is the possibility for an attacker to consider several instances simultaneously. This is precisely the DOOM problem we referred to above.

The DOOM problem was first considered in [JJ02] then analyzed in [Sen11] for Dumer's variant of ISD. From this point and till the end of this section, the parameters n, k, w are fixed.

5.1 An Attack Using Multiple Instances

An attacker may produce many, say q, favorable messages and hash them to obtain $\mathbf{s}_1, \dots, \mathbf{s}_q$ submitted to a solver of Problem 1 together with a parity check matrix of the public key. If it is successful, it has forged a new signature.

Note that in the security reduction, the assumption related to DOOM is precisely the same, that is assuming key indistinguishability and a proper distribution of the signatures, the adversary has to solve a DOOM instance as described above and the reduction is tight in this respect.

5.2 ISD – Information Set Decoding

The skeleton of the ISD algorithm for solving DOOM is given by Algorithm 3.

Algorithm 3 (generalized) ISD

```
1: input: $\mathbf{H} \in \mathbb{F}_2^{(n-k)\times n}, S = \{\mathbf{s}_1, \dots, \mathbf{s}_q\} \subset \mathbb{F}_2^{(n-k)}, w$ integer
2: loop
3:     pick an $n \times n$ permutation matrix $\mathbf{P}$
4:     perform partial Gaussian elimination on $\mathbf{HP}$
       $$\mathbf{UHP} = \left[\begin{array}{c|c} \mathbf{I}_{n-k-\ell} & \mathbf{H}'' \\ \hline 0 & \mathbf{H}' \end{array}\right] \begin{array}{l} \updownarrow n-k-\ell \\ \updownarrow \ell \end{array} \quad \mathbf{Us}_i^T = \left[\begin{array}{c} \mathbf{s}_i''^T \\ \hline \mathbf{s}_i'^T \end{array}\right] \quad i = 1, \dots, q$$
5:     find $\mathcal{E}$ = all solutions of DOOM$(\mathbf{H}', S', p)$, $\mathbf{H}' \in \mathbb{F}_2^{\ell\times(k+\ell)}, S' = \{\mathbf{s}'_1, \dots, \mathbf{s}'_q\}$
6:     for all $(e', i) \in \mathcal{E}$ do
7:         $\mathbf{e}'' \leftarrow \mathbf{e}'\mathbf{H}''^T + \mathbf{s}_{i''}$ ; $\mathbf{e} \leftarrow (\mathbf{e}'' \mid \mathbf{e}')\mathbf{P}^T$
8:         if $|\mathbf{e}| = w$ then print $(\mathbf{e}, i)$
```

In all variants of ISD, the set $\mathcal{E}$ computed at Instruction 5 contains (up to a small polynomial factor) all the solutions of DOOM$(\mathbf{H}', S', p)$. The cost to produce $\mathcal{E}$ dominates the cost of one loop of Algorithm 3, we denote it by $C_q(p, \ell)$. As it is described, the loop is repeated forever and just prints a stream of solutions. The standard version corresponds to a single instance, that is $q = 1$. Below we explain how the cost estimate of the algorithm varies in various situations: when we have a single instance and a single solution, when the number of solutions increases and when the number of instances increases. For each value of n, k, w and q, the algorithm is optimized over the parameters p and ℓ. The optimal values of p and ℓ will change with the number of solutions and the number of instances.

Single Instance and Single Solution. We consider a situation where we wish to estimate the cost of the algorithm for producing one specific solution of Problem 2, that is $q = 1$. In that case, even when w is large and there are multiple solutions, the solution we are looking for, say $\mathbf{e}$, is printed if and only if the permutation $\mathbf{P}$ is such that $|\mathbf{e}'| = p$ and $|\mathbf{e}''| = w - p$ where $(\mathbf{e}'' \mid \mathbf{e}') \leftarrow \mathbf{e}\left(\mathbf{P}^{-1}\right)^T$. This will happen with probability $\mathcal{P}(p, \ell)$ leading to the workfactor WF$^{(1)}$

$$\mathcal{P}(p, \ell) = \frac{\binom{n-k-\ell}{w-p}\binom{k+\ell}{p}}{\binom{n}{w}}, \mathrm{WF}^{(1)} = \min_{p,\ell} \frac{C_1(p, \ell)}{\mathcal{P}(p, \ell)},$$

which is obtained by solving an optimization problem over p and ℓ. The exact expression of $C_1(p, \ell)$ depends on the variant, for instance, for Dumer's algorithm [Dum91] we have $C_1(p, \ell) =$

$\max\left(\sqrt{\binom{k+\ell}{p}}, \binom{k+\ell}{p}2^{-\ell}\right)$ up to a small polynomial factor. For more involved variants [BJMM12, MO15], the value of $C_1(p,\ell)$ is, for each (p,ℓ), the solution of another optimization problem.

Single Instance and Multiple Solutions. We now consider a situation where there are M solutions to a syndrome decoding problem ($q = 1$). If w is larger than the Gilbert-Varshamov bound we expect $M = \binom{n}{w}/2^{n-k}$ else $M = 1$. Assuming each of the M solutions can be independently produced, the probability that one particular iteration produces (at least) one of the solutions becomes $\mathcal{P}_M(p,\ell) = 1 - (1 - \mathcal{P}(p,\ell))^M$. The corresponding workfactor is

$$\mathrm{WF}^{(M)} = \min_{p,\ell} \frac{C_1(p,\ell)}{\mathcal{P}_M(p,\ell)}.$$

Let (p_0, ℓ_0) be the optimal value of the pair (p,ℓ) for a single instance.

Case 1: $\mathcal{P}(p_0,\ell_0) \leq 1/M$. Then, up to a small constant (at most $\exp(-1)$) we have $\mathcal{P}_M(p_0,\ell_0) = 1 - (1 - \mathcal{P}(p_0,\ell_0))^M \approx M\mathcal{P}(p_0,\ell_0)$. Also remark that $\mathcal{P}_M(p,\ell) \leq M\mathcal{P}(p,\ell)$ and thus $\mathrm{WF}^{(M)} \geq \mathrm{WF}^{(1)}/M$. We also have

$$\mathrm{WF}^{(M)} = \min_{p,\ell} \frac{C_1(p,\ell)}{\mathcal{P}_M(p,\ell)} \leq \frac{C_1(p_0,\ell_0)}{\mathcal{P}_M(p_0,\ell_0)} = \frac{1}{M}\frac{C_1(p_0,\ell_0)}{\mathcal{P}(p_0,\ell_0)} = \frac{1}{M}\mathrm{WF}^{(1)}$$

up to a small constant factor. In other words, the optimal parameters remain the same and the workfactor for multiple solutions is simply obtained by dividing the single solution workfactor by the number of solutions.

Case 2: $\mathcal{P}(p_0,\ell_0) > 1/M$. In this case the success probability $\mathcal{P}_M(p_0,\ell_0) < M\mathcal{P}(p_0,\ell_0)$ and the pair (p,ℓ) that minimizes the workfactor is going to be different. It gives a different optimization problem and we observe that the gain is much less than the factor M of Case 1.

In practice, and for the parameters we consider in this work, we are always in Case 2. In fact, for $k/n = 0.5$, with Dumer's algorithm Case 1 only applies when $w/n < 0.150$, while the Gilbert-Varshamov bound corresponds to $w/n = 0.110$. With BJMM's algorithm, Case 1 only happens when $w/n \leq 0.117$. In our signature scheme we have $w/n \approx 0.19$ and we always fall in Case 2, even with a single instance.

Multiples Instances with Multiple Solutions. We now consider the case where the adversary has access to q instances of Problem 2 for the same matrix $\mathbf{H}$ and various syndromes. This is Problem 1 that appears in the security reduction. For each instance, we expect $M = \max\left(1, \binom{n}{w}/2^{n-k}\right)$ solutions.

As before, the cost is dominated by Instruction 5, which we denote by $C_q(p,\ell)$, and the probability of success is $\mathcal{P}_{qM}(p,\ell) = 1 - (1 - \mathcal{P}(p,\ell))^{qM}$. Next this cost has to be minimized over p and ℓ

$$\mathrm{WF}_q^{(M)} = \min_{p,\ell} \frac{C_q(p,\ell)}{\mathcal{P}_{qM}(p,\ell)}.$$

Indeed, solving $\mathrm{DOOM}(\mathbf{H}', S', p)$ is not specified here. This is in fact what [JJ02, Sen11] are about. For instance with Dumer's algorithm, we have [Sen11]

$$C_q(p,\ell) = \max\left(\sqrt{q\binom{k+\ell}{p}}, \frac{q\binom{k+\ell}{p}}{2^\ell}\right),\ q \leq \binom{k+\ell}{p}$$

up to a small polynomial factor. Introducing multiple instances in advanced variants of ISD has not been done so far and is an open problem. We give in Table 1 the asymptotic exponent for various decoding distances and for the code rate 0.5. The third column gives the largest useful value of q. It is likely that BJMM's algorithm will have a slightly lower exponent when addressing multiple instances. Note that for Dumer's algorithm in this range of parameters, the improvement

		Dumer			BJMM
w/n	$\frac{1}{n}\log_2 M$	$\frac{1}{n}\log_2 q$	$\frac{1}{n}\log_2 \mathrm{WF}_q^{(M)}$	$\frac{1}{n}\log_2 \mathrm{WF}^{(M)}$	$\frac{1}{n}\log_2 \mathrm{WF}^{(M)}$
0.11	0.0000	0.0872	0.0872	0.1152	0.1000
0.15	0.1098	0.0448	0.0448	0.0535	0.0486
0.19	0.2015	0.0171	0.0171	0.0184	0.0175

Table 1: Asymptotic Exponent for Algorithm 3 for $k/n = 0.5$

from $\mathrm{WF}^{(M)}$ (single instance) to $\mathrm{WF}_q^{(M)}$ (multiple instances) is relatively small, there is no reason to expect a much different behavior for BJMM.

Finally, let us mention that the best asymptotic exponent among all known decoding techniques was proposed in 2015 by May and Ozerov [MO15]. However it is penalized by a big polynomial overhead which makes it unpractical at this point for the problems considered here.

5.3 Other Decoding Techniques.

As mentioned in [CJ04, FS09], the Generalized Birthday Algorithm (GBA) [Wag02] is a relevant technique to solve decoding problems, in particular when there are multiple solutions. However, it is competitive only when the ratio k/n tends to 1, and does not apply here. We refer the reader to [MS09] for more details on GBA and its usage.

6 Key Attack

6.1 The Idea of the Attack

A $(U|U+V)$ code where U and V are random seems very close to a random linear code. There is for instance only a very slight difference between the weight distribution of a random linear code and the weight distribution of a random $(U|U+V)$-code of the same length and dimension. This slight difference happens for small and large weights and is due to codewords of the form $(\mathbf{u}|\mathbf{u})$ where $\mathbf{u}$ belongs to U or codewords of the form $(\mathbf{0}|\mathbf{v})$ where $\mathbf{v}$ belongs to V. More precisely, we have the following proposition

Proposition 8. *Assume that we choose a $(U|U+V)$ code by picking the parity-check matrices of U and V uniformly at random among the binary matrices of size $(n/2-k_U)\times n/2$ and $(n/2-k_V)\times n/2$ respectively. Let $a_{(U|U+V)}(w)$, $a_{(U|U)}(w)$ and $a_{(0|V)}(w)$ be the expected number of codewords of weight w that are respectively in the $(U|U+V)$ code, of the form $(\mathbf{u}|\mathbf{u})$ where $\mathbf{u}$ belongs to U and of the form $(\mathbf{0}|\mathbf{v})$ where $\mathbf{v}$ belongs to V. These numbers are given for even w in $\{0,\dots,n\}$ by*

$$a_{(U|U+V)}(w) = \frac{\binom{n/2}{w/2}}{2^{n/2-k_U}} + \frac{\binom{n/2}{w}}{2^{n/2-k_V}} + \frac{1}{2^{n-k_U-k_V}}\left(\binom{n}{w} - \binom{n/2}{w} - \binom{n/2}{w/2}\right)$$

$$a_{(U|U)}(w) = \frac{\binom{n/2}{w/2}}{2^{n/2-k_U}} \quad ; \quad a_{(0|V)}(w) = \frac{\binom{n/2}{w}}{2^{n/2-k_V}}$$

and for odd w in $\{0,\dots,n\}$ by

$$a_{(U|U+V)}(w) = \frac{\binom{n/2}{w}}{2^{n/2-k_V}} + \frac{1}{2^{n-k_U-k_V}}\left(\binom{n}{w} - \binom{n/2}{w}\right)$$

$$a_{(U|U)}(w) = 0 \quad ; \quad a_{(0|V)}(w) = \frac{\binom{n/2}{w}}{2^{n/2-k_V}}$$

On the other hand, when we choose a code of length n with a random parity-check matrix of size $(n-k_U-k_V)\times n$ chosen uniformly at random, then the expected number $a(w)$ of codewords of weight $w>0$ is given by

$$a(w) = \frac{\binom{n}{w}}{2^{n-k_U-k_V}}.$$

Remark 4. When the $(U|U+V)$ code is chosen in this way, its dimension is k_U+k_V with probability $1-O\left(\max(2^{k_U-n/2},2^{k_V-n/2})\right)$. This also holds for the random codes of length n.

We have plotted in Figure 6 the normalized logarithm of the density of codewords of the form $(\mathbf{u}|\mathbf{u})$ and $(\mathbf{0}|\mathbf{v})$ of relative *even* weight $x \overset{\triangle}{=} \frac{w}{n}$ against x in the case U is of rate $\frac{k_U}{n/2} = 0.6$ and V is of rate $\frac{k_V}{n/2} = 0.4$. These two relative densities are defined respectively by

$$\alpha_{(U|U)}(w/n) = \frac{\log_2(a_{(U|U)}(w)/a(w))}{n} \quad ; \quad \alpha_{(0|V)}(w/n) = \frac{\log_2(a_{(0|V)}(w)/a(w))}{n}$$

We see that for a relative weight w/n below approximately 0.18 almost all the codewords are of the form $(\mathbf{0}|\mathbf{v})$ in this case.

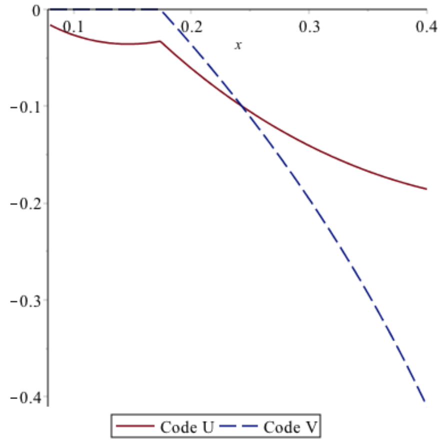


Fig. 6: $\alpha_{(U|U)}(w/n)$ and $\alpha_{(0|V)}(w/n)$ against $x \overset{\triangle}{=} \frac{w}{n}$.

Since the weight distribution is invariant by permuting the positions, this slight difference also survives in the permuted version of $(U|U+V)$. These considerations lead to the best attack we have found for recovering the structure of a permuted $(U|U+V)$ code. It consists in applying known algorithms aiming at recovering low weight codewords in a linear code. We run such an algorithm until getting at some point either a permuted $(\mathbf{u}|\mathbf{u})$ codeword where $\mathbf{u}$ is in U or a permuted $(\mathbf{0}|\mathbf{v})$ codeword where $\mathbf{v}$ belongs to V. The rationale behind this algorithm is that the density of codewords of the form $(\mathbf{u},\mathbf{u})$ or $(\mathbf{0},\mathbf{v})$ is bigger when the weight of the codeword gets smaller.

Once we have such a codeword we can bootstrap from there very similarly to what has been done in [OT11, Subs. 4.4]. Note that this attack is actually very close in spirit to the attack that was devised on the KKS signature scheme [OT11]. In essence, the attack against the KKS scheme really amounts to recover the support of the V code. The difference with the KKS scheme is that the support of V is much bigger in our case. As explained in the conclusion of [OT11] the attack against the KKS scheme has in essence an exponential complexity. This exponent becomes really prohibitive in our case when the parameters of U and V are chosen appropriately as we will now explain.

6.2 Recovering the V code up to a permutation

The aforementioned attack recovers V up to some permutation of the positions. In a first step it recovers a basis of

$$V' \overset{\triangle}{=} (0|V)\mathbf{P} = \{(\mathbf{0},\mathbf{v})\mathbf{P} : \mathbf{v} \in V\}.$$

Once this is achieved, the support $\mathrm{Supp}(V')$ of V' can be obtained. Recall that this is the set of positions for which there exists at least one codeword of V' that is non-zero in this position. This allows to recover the code V up to some permutation. The basic algorithm for recovering the support of V' and a basis of V' is given in Algorithm 4.

Algorithm 4 ComputeV: algorithm that computes a set of independent elements in V'.

```
Parameters: (i) $\ell$ : small integer ($\ell \leqslant 40$),
(ii) $p$ : very small integer (typically $1 \leqslant p \leqslant 10$).
Input: (i) $\mathscr{C}_{\mathrm{pub}}$ the public code used for verifying signatures.
(ii) $N$ a certain number of iterations
Output: an independent set of elements in $V'$
 1: function ComputeV($\mathscr{C}_{\mathrm{pub}}$,$N$)
 2:     for $i = 1, \dots, N$ do
 3:         $B \leftarrow \emptyset$
 4:         Choose a set $I \subset \{1, \dots, n\}$ of size $n-k-\ell$ uniformly at random
 5:         $\mathscr{L} \leftarrow$ Codewords($\mathrm{Punc}_I(\mathscr{C}_{\mathrm{pub}}), p$)
 6:         for all $\mathbf{x} \in \mathscr{L}$ do
 7:             $\mathbf{x} \leftarrow$ Complete($\mathbf{x}, I, \mathscr{C}_{\mathrm{pub}}$)
 8:             if CheckV($\mathbf{x}$) then
 9:                 add $\mathbf{x}$ to $B$ if $\mathbf{x} \notin < B >$
10:     return $B$
```

It uses other auxiliary functions

- Codewords$(\mathrm{Punc}_I(\mathscr{C}_{\mathrm{pub}}), p)$ which computes all (or a big fraction of) codewords of weight p of the punctured public code $\mathrm{Punc}_I(\mathscr{C}_{\mathrm{pub}})$. All modern [Dum91, FS09, MMT11, BJMM12, MO15] algorithms for decoding linear codes perform such a task in their inner loop.
- Complete$(\mathbf{x}, I, \mathscr{C}_{\mathrm{pub}})$ which computes the codeword $\mathbf{c}$ in $\mathscr{C}_{\mathrm{pub}}$ such that its restriction outside I is equal to $\mathbf{x}$.
- CheckV$(\mathbf{x})$ which checks whether $\mathbf{x}$ belongs to V'.

Choosing N appropriately. Let us first analyze how we have to choose N such that ComputeV returns $\Omega(1)$ elements. This is essentially the analysis which can be found in [OT11, Subsec 5.2]. This analysis leads to

Proposition 9. *The probability P_{succ} that one iteration of the for loop (Instruction 2) in Algorithm 4 adds elements to the list B is lower-bounded by*

$$P_{succ} \geq \sum_{w=0}^{n/2} \frac{\binom{n/2}{w}\binom{n/2}{n-k-\ell-w}}{\binom{n}{n-k-\ell}} f\left(\binom{n/2-w}{p} 2^{k_V+w-n/2}\right) \tag{8}$$

where f is the function defined by $f(x) \overset{\triangle}{=} \max\left(x(1-x/2), 1-\frac{1}{x}\right)$. Algorithm 4 returns a non zero list with probability $\Omega(1)$ when N is chosen as $N = \Omega\left(\frac{1}{P_{succ}}\right)$.

Proof. It will be helpful to recall [OT11, Lemma 3]

Lemma 3. *Choose a random code $\mathscr{C}_{rand}$ of length n from a parity-check matrix of size $r \times n$ chosen uniformly at random in $\mathbb{F}_2^{r \times n}$. Let X be some subset of $\mathbb{F}_2^n$ of size m. We have*

$$\mathbb{P}(X \cap \mathscr{C}_{rand} \neq \emptyset) \geq f\left(\frac{m}{2^r}\right).$$

To lower-bound the probability P_{succ} that an iteration is successful, we bring in the following random variables

$$I' \stackrel{\triangle}{=} I \cap \text{Supp}(I'') \quad \text{and} \quad W \stackrel{\triangle}{=} |I'|$$

where I'' is the set of positions that are of the images of the permutation $\mathbf{P}$ of the $n/2$ last positions. $\mathsf{ComputeV}$ outputs at least one element of V' if there is an element of weight p in $\text{Punc}_{I'}(V')$. Therefore the probability of success P_{succ} is given by

$$P_{\text{succ}} = \sum_{w=0}^{n/2} \mathbb{P}(W = w)\mathbb{P}\left(\exists \mathbf{x} \in V' : |\mathbf{x}_{\bar{I}'}| = p \mid W = w\right) \tag{9}$$

where $\bar{I}' \stackrel{\triangle}{=} \text{Supp}(V') \setminus I'$. On the other hand, by using Lemma 3 with the set

$$X \stackrel{\triangle}{=} \left\{\mathbf{x} = (x_j)_{j \in \text{Supp}(V')} : |\mathbf{x}_{\bar{I}'}| = p\right\}$$

which is of size $\binom{n/2-w}{p}2^w$, we obtain

$$\mathbb{P}\left(\exists \mathbf{x} \in V' : |\mathbf{x}_{\bar{I}'}| = p | W = w\right) \geq f(x). \tag{10}$$

with

$$x \stackrel{\triangle}{=} \frac{\binom{n/2-w}{p}2^w}{2^{n/2-k_V}} = \binom{n/2-w}{p} 2^{k_V+w-n/2}$$

The first quantity is clearly equal to

$$\mathbb{P}(W = w) = \frac{\binom{n/2}{w}\binom{n/2}{n-k-\ell-w}}{\binom{n}{n-k-\ell}}. \tag{11}$$

Plugging in the expressions obtained in (10) and (11) in (9) we have an explicit expression of a lower bound on P_{succ}

$$P_{\text{succ}} \geq \sum_{w=0}^{n/2} \frac{\binom{n/2}{w}\binom{n/2}{n-k-\ell-w}}{\binom{n}{n-k-\ell)}} f\left(\binom{n/2-w}{p} 2^{k_V+w-n/2}\right) \tag{12}$$

The claim on the number N of iterations follows directly from this. □

Complexity of recovering a permuted version of *V*. The complexity of a call to $\mathsf{ComputeV}$ can be estimated as follows. The complexity of computing the list of codewords of weight p in a code of length $k + \ell$ and dimension k is equal to $C_1(p, \ell)$ (this quantity is introduced in §5). It depends on the particular algorithm used here [Dum91, FS09, MMT11, BJMM12, MO15]. This is the complexity of the call $\mathsf{Codewords}(\text{Punc}_I(\mathscr{C}_{\text{pub}}), p)$ in Step 5 in Algorithm 4. The complexity of $\mathsf{ComputeV}$ and hence the complexity of recovering a permuted version of V is clearly lower bounded by $\Omega\left(\frac{C_1(p,\ell)}{P_{\text{succ}}}\right)$. It turns out that the whole complexity of recovering a permuted version of V is actually of this order, namely $\Theta\left(\frac{C_1(p,\ell)}{P_{\text{succ}}}\right)$. This can be done by a combination of two techniques

- Once a non-zero element of V' has been identified, it is much easier to find other ones. This uses one of the tricks for breaking the KKS scheme (see [OT11, Subs. 4.4]). The point is the following: if we start again the procedure $\mathsf{ComputeV}$, but this time by choosing a set I on which we puncture the code which contains the support of the codeword that we already found, then the number N of iterations that we have to perform until finding a new element is negligible when compared to the original value of N.

- The call to CheckV can be implemented in such a way that the additional complexity coming from all the calls to this function is of the same order as the N calls to Codewords. The strategy to adopt depends on the values of the dimensions k and k_V. In certain cases, it is easy to detect such codewords since they have a typical weight that is significantly smaller than the other codewords. In more complicated cases, we might have to combine a technique checking first the weight of $\mathbf{x}$, if it is above some prescribed threshold, we decide that it is not in V', if it is below the threshold, we decide that it is a suspicious candidate and use then the previous trick. We namely check whether the support of the codeword $\mathbf{x}$ can be used to find other suspicious candidates much more quickly than performing N calls to CheckV.

To keep the length of this paper within some reasonable limit we avoid here giving the analysis of those steps and we will just use the aforementioned lower bound on the complexity of recovering a permuted version of V.

6.3 Recovering the U code up to permutation

We consider here the permuted code

$$U' \stackrel{\triangle}{=} (U|U)\mathbf{P} = \{(\mathbf{u},\mathbf{u})\mathbf{P} : \mathbf{u} \in U\}.$$

The attack in this case consists in recovering a basis of U'. Once this is done, it is easy to recover the U code up to permutation by matching the pairs of coordinates which are equal in U'. The algorithm for recovering U' is the same as the algorithm for recovering V'. We call the associated function ComputeU though since they differ in the choice for N. The analysis is slightly different indeed.

Choosing N appropriately. As in the previous subsection let us analyze how we have to choose N in order that ComputeU returns $\Omega(1)$ elements of U'. We have in this case the following result.

Proposition 10. *The probability P_{succ} that one iteration of the for loop (Instruction 2) in ComputeU adds elements to the list B is lower-bounded by*

$$P_{succ} \geq \sum_{w=0}^{n/2} \frac{\binom{n/2}{w}\binom{n/2-w}{k+\ell-2w}2^{k+\ell-2w}}{\binom{n}{k+\ell}} \max_{i=0}^{\lfloor p/2 \rfloor} f\left(\frac{\binom{k+\ell-2w}{p-2i}\binom{w}{i}}{2^{\max(0,k+\ell-w-k_U)}}\right) \tag{13}$$

where f is the function defined by $f(x) \stackrel{\triangle}{=} \max\left(x(1-x/2), 1-\frac{1}{x}\right)$. ComputeU returns a non zero list with probability $\Omega(1)$ when N is chosen as $N = \Omega\left(\frac{1}{P_{succ}}\right)$.

Proof. Here the crucial notion is the concept of *matched positions*. We say that two positions i and j are matched if and only if $c_i = c_j$ for every $\mathbf{c} \in U'$. There are clearly $n/2$ pairs of matched positions. W will now be defined by the number of matched pairs that are included in $\{1,\dots,n\}\setminus I$. We compute the probability of success as before by conditioning on the values taken by W:

$$P_{\text{succ}} = \sum_{w=0}^{n/2} \mathbb{P}(W = w)\mathbb{P}\left(\exists \mathbf{x} \in U' : |\mathbf{x}_{\bar{I}}| = p \mid W = w\right) \tag{14}$$

where $\bar{I} \stackrel{\triangle}{=} \{1,\dots,n\}\setminus I$. Notice that we can partition $\bar{I}$ as $\bar{I} = J_1 \cup J_2$ where J_2 consists in the union of the matched pairs in $\bar{I}$. Note that $|J_2| = 2w$. We may further partition J_2 as $J_2 = J_{21} \cup J_{22}$ where the elements of a matched pair are divided into the two sets. In other words, neither J_{21} nor J_{22} contains a matched pair. We are going to consider the codes

$$U" \stackrel{\triangle}{=} \underset{I}{\text{Punc}}(U')$$

$$U''' \stackrel{\triangle}{=} \underset{I \cup J_{22}}{\text{Punc}}(U')$$

The last code is of length $k+\ell-w$. The point of defining the first code is that

$$\mathbb{P}\left(\exists \mathbf{x} \in U' : |\mathbf{x}_{\bar{I}}| = p \mid W = w\right)$$

is equal to the probability that U" contains a codeword of weight p. The problem is that we can not apply Lemma 3 to it due to the matched positions it contains. This is precisely the point of defining U'''. In this case, we can consider that it is a random code whose parity-check matrix is chosen uniformly at random among the set of matrices of size $\max(0, k+\ell-w-k_U) \times (k+\ell-w)$. We can therefore apply Lemma 3 to it. We have to be careful about the words of weight p in U" though, since they do not have the same probability of occurring in U" due to the possible presence of matched pairs in the support. This is why we introduce for i in $\{0, \dots, \lfloor p/2 \rfloor\}$ the sets X_i defined as follows

$$X_i \overset{\triangle}{=} \{\mathbf{x} = (x_i)_{i \in \bar{I} \setminus J_{22}} \mathbb{F}_2^{k+\ell-w} : |\mathbf{x}_{J_1}| = p - 2i,\ |\mathbf{x}_{J_{21}}| = i\}$$

A codeword of weight p in U" corresponds to some word in one of the X_i's by puncturing it in J_{22}. We obviously have the lower bound

$$\mathbb{P}\left\{\exists \mathbf{x} \in U' : |\mathbf{x}_{\bar{I}}| = p \mid W = w\right\} \geq \max_{i=0}^{\lfloor p/2 \rfloor} \left\{\mathbb{P}(X_i \cap U''' \neq \emptyset)\right\} \tag{15}$$

By using Lemma 3 we have

$$\mathbb{P}(X_i \cap U''' \neq \emptyset) \geq f\left(\frac{\binom{k+\ell-2w}{p-2i}\binom{w}{i}}{2^{\max(0,k+\ell-w-k_U)}}\right). \tag{16}$$

On the other hand, we may notice that $\mathbb{P}(W = w) = \mathbb{P}(w_2(\mathbf{e}) = w)$ when $\mathbf{e}$ is drawn uniformly at random among the binary words of weight $k+\ell$ and length n. By using Proposition 4 we deduce

$$\mathbb{P}(W = w) = \frac{\binom{n/2}{w}\binom{n/2-w}{k+\ell-2w}2^{k+\ell-2w}}{\binom{n}{k+\ell}}.$$

These considerations lead to the following lower bound on P_{succ}

$$P_{\text{succ}} \geq \sum_{w=0}^{n/2} \frac{\binom{n/2}{w}\binom{n/2-w}{k+\ell-2w}2^{k+\ell-2w}}{\binom{n}{k+\ell}} \max_{i=0}^{\lfloor p/2 \rfloor} f\left(\frac{\binom{k+\ell-2w}{p-2i}\binom{w}{i}}{2^{\max(0,k+\ell-w-k_U)}}\right) \tag{17}$$

□

Complexity of recovering a permuted version of *U*. As for recovering the permuted V code, the complexity for recovering the permuted U is of order $\Omega\left(\frac{C_1(p,\ell)}{P_{\text{succ}}}\right)$.

6.4 Distinguishing a $(U|U+V)$ code

It is not clear in the first case that from the single knowledge of V' and a permuted version of V we are able to find a permutation of the positions which gives to the whole code the structure of a $(U|U+V)$-code. However in both cases as single successful call to ComputeV (resp. ComputeU) is really distinguishing the code from a random code of the same length and dimension. In other words, we have a distinguishing attack whose complexity is given by $\min(O(C_U), O(C_V))$ where

$$C_U \overset{\triangle}{=} \frac{C_1(p,\ell)}{\sum_{w=0}^{n/2} \frac{\binom{n/2}{w}\binom{n/2-w}{k+\ell-2w}2^{k+\ell-2w}}{\binom{n}{k+\ell}} \max_{i=0}^{\lfloor p/2 \rfloor} f\left(\frac{\binom{k+\ell-2w}{p-2i}\binom{w}{i}}{2^{\max(0,k+\ell-w-k_U)}}\right)}$$

$$C_V \overset{\triangle}{=} \frac{C_1(p,\ell)}{\sum_{w=0}^{n/2} \frac{\binom{n/2}{w}\binom{n/2}{n-k-\ell-w}}{\binom{n}{n-k-\ell}} f\left(\binom{n/2-w}{p}2^{k_V+w-n/2}\right)}$$

and $f(x) \overset{\triangle}{=} \max\left(x(1-x/2), 1-1/x\right)$. As for the decoders of §5 the above numbers are minimized (independently) over p and ℓ.

We end this section by remarking that the dual of a code $(U|U+V)$ is $(U^{\perp}+V^{\perp}|V^{\perp})$ thus we have the same attack with the dual. With $k/n = 0.5$, these two attacks have the same complexity as $C_U = C_{V^{\perp}}$ and $C_V = C_{U^{\perp}}$.

7 Parameter Selection

In the light of the security proof in §3 and the rejection sampling method in §4, we need to derive parameters which lead to negligible success for the two following problems:

1. Solve a syndrome decoding problem with multiple instances (DOOM) for parameters n, k, w and an arbitrarily large number of instances.
2. Distinguish public matrices of the code family $(U|U+V)$ from random matrices of same size.

In the security proof we required a salt size $\lambda_0 = \lambda + 2\log_2(q_{sig})$ where q_{sig} is the number of signature queries allowed to the adversary. Since $q_{sig} \leq 2^{\lambda}$ (λ the security parameter) we choose a conservative $\lambda_0 = 3\lambda$. We gave in §5 and §6 state-of-the-art algorithms for the two problems mentioned above. This served as a basis for the parameters proposed in Table 2. For the key security, the estimates C_U and C_V are derived from the formulas at the end of §6. In those formulas the $C_1(p,\ell)$ term derives from Dumer's algorithm. Using more involved techniques [MMT11, BJMM12, MO15] will reduce the key security but will leave it above the security claims. For the message security ($\log_2$ WF), it is based on the DOOM variant of Dumer's algorithm, which is the current state-of-the-art. Algorithmic improvements, like adapting DOOM to BJMM, may lower the message security and require an adjustment of the sizes.

λ (security)	80	128	256
n	4678	7486	14970
k	2339	3743	7485
k_V	899	1439	2878
w	889	1422	2844
Signature length (bits)	4918	7870	15738
Public key size (MBytes)	0.683	1.75	7.00
$\log_2 C_V$ (§6)	163	260	521
$\log_2 C_U$ (§6)	250	400	800
$\log_2$ WF (§5)	80	128	256

Table 2: Proposed Parameters for the $(U \mid U+V)$ Signature Scheme

Note also that there is an additive term $q_{hash}/2\sqrt{2^{n-k}/\binom{n}{w}}$ in the adversary's advantage (see Theorem 2 in the security proof §3). With the current parameters, we have $k = n/2$ and $\binom{n}{w} \approx 2^{0.70n}$, and the above term is of order $q_{hash}2^{-n/10}$ which is always negligible.

Implementation. In Table 2 the ratio w/n is chosen close to 0.19 to minimize the rejection probability (see §4). For the three security levels we need to perform on average 27, 37, or 75 Gaussian eliminations to produce a signature. Most of those Gaussian elimination are performed on parity check matrices of shortened codes. Finally, let us mention that the signature length ($n + 3\lambda$ in the table) can be reduced (by about 30%) by choosing a compact representation of the sparse error vector.

8 Concluding remarks

We have presented a new code-based signature scheme which is EUF-CMA secure under two assumptions from coding theory. Both of those assumptions relate closely to hard decoding problems. Two nice features of this scheme are its simplicity and the fact that it is very close to rely on random linear codes: it is based on $(U|U+V)$ codes where U and V are random linear codes of appropriate dimensions. Using rejection sampling, we have shown how to efficiently avoid key leakage from any number of signatures. The main purpose of our work was to propose this new scheme, give a proof security and assess its security against current knowledge. It scales well with respect to the security parameter λ expressed in bits ($S = 2^{\lambda}$ measures the complexity of the best known attack), since the key size, signature size, signature generation time and signature verification time are respectively of order $O(\lambda^2)$, $O(\lambda)$, $O(\lambda^3)$ and $O(\lambda^2)$. This is the first code-based signature scheme with a security parameter which scales polynomially in key size. By code-based scheme, we mean here the restricted case where we are interested in binary linear codes and we use the Hamming metric for expressing the decoding problem. This setting presents the advantage that we are in the case where the decoding problem has been thoroughly studied for many decades and where it can be considered that the complexity of the best known attacks has not dramatically changed since the early sixties.

Comparison with RankSign. Recently another code-based signature scheme whose security relies on decoding codes with respect to the rank metric has been proposed in [GRSZ14]. It is called RankSign. Strictly speaking, the rank metric consists in viewing an element in $\mathbb{F}_q^N$ (when N is a product $N = m \times n$) as an $m \times n$ matrix over $\mathbb{F}_q$ and the rank distance between two elements $\mathbf{x}$ and $\mathbf{y}$ is defined as the rank of the matrix $\mathbf{x} - \mathbf{y}$. This depends of course on how N is viewed as a product of two elements. Decoding in this metric is known to be an NP hard problem [BFS99, Cou01]. In the particular case of [GRSZ14], the codes which are considered are not $\mathbb{F}_q$-linear but, as is customary in the setting of rank metric based cryptography, $\mathbb{F}_{q^m}$-linear. This allows to reduce the keysize by a factor of m when compared to the $\mathbb{F}_q$-linear setting (for more details see for instance §2, just below Def. 2.3 in [HT15]).

This is generally the key explanation why rank cryptographic systems (when the codes are actually linear on an extension field $\mathbb{F}_{q^m}$ of $\mathbb{F}_q$) have generally shorter keysizes than Hamming metric schemes. This is also the key reason why compared to the scheme proposed here, RankSign enjoys much shorter key sizes: it is of order tens of thousands bits for 128 bits of security. In some sense, the codes used in [GRSZ14] are more structured than just $\mathbb{F}_q$ linear codes. Decoding such codes for the rank metric is not known to be NP-hard anymore. There is however a randomized reduction of this problem to decoding an $\mathbb{F}_q$-linear code for the Hamming metric [GZ16] when the degree m of the extension field is sufficiently big. This situation is in some sense reminiscent to the current thread in cryptography based on codes or on lattices where structured codes (for instance quasi-cyclic codes) or structured lattices (corresponding to an additional ring structure) are taken. In the case at hand, there is however a randomized reduction to an NP complete problem (even if this reduction seems really loose and is not used to devise secure parameters for those schemes). Note however that contrarily to the quasi-cyclic case where it is not known whether decoding a quasi-cyclic code can be done faster than with a polynomial speedup when compared to a generic linear code, it is known how to use the $\mathbb{F}_{q^m}$-linearity of the code to decrease the exponent of the rank metric analogue of the Prange algorithm for decoding [GRS16]. There are also other algorithms for decoding in the rank metric that really use the $\mathbb{F}_{q^m}$-linearity (see [GRS16] again).

Our signature scheme offers however a significant advantage over RankSign when it comes to the security proof. First even if RankSign has a partial security proof showing that when the number of signatures is smaller than $q/2$ it does not leak information, there is no overall reduction of the security scheme to well identified problems in (rank metric) coding theory. Second, the fact that the number of signatures has to be smaller than $q/2$ to avoid information leakage represents a strong constraint on the parameters of the RankSign scheme. The problem of information leakage coming from signatures originating from a same public key is in general the main threat on signature schemes and it is handled in our case in a very satisfying fashion by a rejection sampling method which ensures statistical indistinguishability of the signatures when compared to random words of the same weight. It should also be added that our signature relies on the Hamming metric and that irrespective of the merits of a signature scheme based on the rank metric it is probably desirable to have also a signature scheme working for the Hamming metric due to the general faith in the hardness of decoding with this metric.

Reducing the keysize. The previous comparison with RankSign raises the issue whether it would be possible to reduce the keysize of the signature scheme proposed here. One obvious approach which comes to mind is to use more structured codes such as quasi-cyclic codes. There are two issues with this research thread. The first is that this degrades key security and this has to be taken into account into the distinguisher of Section 6. This certainly requires to use only quasi-cyclic codes with small circulant blocks and will allow only for some moderate gains in the keysize. Second, this also changes somehow the security proof and in particular the arguments used for instance in Game 2. Another related way to reduce the keysize would be to take advantage of the gap of the best attack against the key and the best attack for forging a signature to change a little bit the scheme to degrade a little bit key security but by improving the security against forgeries. This suggests that there might exist other choices of code parameters with different and possibly better features.

Implementation issues. Though rejection sampling in our algorithm is relatively unobtrusive (a few samples per signature) we did not examine how much impact a loss of accuracy in the calculations could have on the amount of key leakage. Also, we need to implement generic decoding for producing signatures and we use a plain Prange algorithm for that. Efficiency could be improved by using Dumer [Dum91] or MMT [MMT11] algorithm, but they must first be adapted to produce provably independent outputs.

References

[Arı09] Erdal Arıkan. Channel polarization: a method for constructing capacity-achieving codes for symmetric binary-input memoryless channels. *IEEE Trans. Inform. Theory*, 55(7):3051–3073, 2009.

[Bar97] Alexander Barg. Complexity issues in coding theory. *Electronic Colloquium on Computational Complexity*, October 1997.

[BBC$^+$13] Marco Baldi, Marco Bianchi, Franco Chiaraluce, Joachim Rosenthal, and Davide Schipani. Using LDGM codes and sparse syndromes to achieve digital signatures. In *Post-Quantum Cryptography 2013*, volume 7932 of *LNCS*, pages 1–15. Springer, 2013.

[BCD$^+$16] Magali Bardet, Julia Chaulet, Vlad Dragoi, Ayoub Otmani, and Jean-Pierre Tillich. Cryptanalysis of the McEliece public key cryptosystem based on polar codes. In *Post-Quantum Cryptography2016*, LNCS, pages 118–143, Fukuoka, Japan, February 2016.

[BDK$^+$11] Boaz Barak, Yevgeniy Dodis, Hugo Krawczyk, Olivier Pereira, Krzysztof Pietrzak, François-Xavier Standaert, and Yu Yu. Leftover hash lemma, revisited. In *Advances in Cryptology - CRYPTO 2011 - 31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2011. Proceedings*, pages 1–20, 2011.

[Ber10] Daniel J. Bernstein. Grover vs. McEliece. In Nicolas Sendrier, editor, *Post-Quantum Cryptography 2010*, volume 6061 of *LNCS*, pages 73–80. Springer, 2010.

[BFS99] Jonathan F. Buss, Gudmund S. Frandsen, and Jeffrey O. Shallit. The computational complexity of some problems of linear algebra. *J. Comput. System Sci.*, 58(3):572–596, June 1999.

[BJMM12] Anja Becker, Antoine Joux, Alexander May, and Alexander Meurer. Decoding random binary linear codes in $2^{n/20}$: How $1 + 1 = 0$ improves information set decoding. In *Advances in Cryptology - EUROCRYPT 2012*, LNCS. Springer, 2012.

[BMS11] Paulo S.L.M Barreto, Rafael Misoczki, and Marcos A. Jr. Simplicio. One-time signature scheme from syndrome decoding over generic error-correcting codes. *Journal of Systems and Software*, 84(2):198–204, 2011.

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *CCS '93, Proceedings of the 1st ACM Conference on Computer and Communications Security, Fairfax, Virginia, USA, November 3-5, 1993.*, pages 62–73, 1993.

[BR96] Mihir Bellare and Phillip Rogaway. The exact security of digital signatures-how to sign with rsa and rabin. In *Advances in Cryptology - EUROCRYPT '96*, volume 1070 of *LNCS*, pages 399–416. Springer, 1996.

[CFS01] Nicolas Courtois, Matthieu Finiasz, and Nicolas Sendrier. How to achieve a McEliece-based digital signature scheme. In *Advances in Cryptology - ASIACRYPT 2001*, volume 2248 of *LNCS*, pages 157–174, Gold Coast, Australia, 2001. Springer.

[CJ04] Jean-Sebastien Coron and Antoine Joux. Cryptanalysis of a provably secure cryptographic hash function. IACR Cryptology ePrint Archive, Report 2004/013, 2004. `http://eprint.iacr.org/`.

[Cor02] Jean-Sébastien Coron. Optimal security proofs for PSS and other signature schemes. In *Advances in Cryptology - EUROCRYPT 2002, International Conference on the Theory and Applications of Cryptographic Techniques, Amsterdam, The Netherlands, April 28 - May 2, 2002, Proceedings*, pages 272–287, 2002.

[Cou01] Nicolas Courtois. Efficient zero-knowledge authentication based on a linear algebra problem MinRank. In *Advances in Cryptology - ASIACRYPT 2001*, volume 2248 of *LNCS*, pages 402–421, Gold Coast, Australia, 2001. Springer.

[COV07] Pierre-Louis Cayrel, Ayoub Otmani, and Damien Vergnaud. On Kabatianskii-Krouk-Smeets signatures. In *Arithmetic of Finite Fields - WAIFI 2007*, volume 4547 of *LNCS*, pages 237–251, Madrid, Spain, June 21–22 2007.

[CTS16] Rodolfo Canto-Torres and Nicolas Sendrier. Analysis of information set decoding for a sub-linear error weight. In *Post-Quantum Cryptography 2016*, LNCS, pages 144–161, Fukuoka, Japan, February 2016.

[DAT17] Thomas Debris-Alazard and Jean-Pierre Tillich. Statistical decoding. preprint, January 2017. arXiv:1701.07416.

[Dum91] Ilya Dumer. On minimum distance decoding of linear codes. In *Proc. 5th Joint Soviet-Swedish Int. Workshop Inform. Theory*, pages 50–52, Moscow, 1991.

[FGO$^+$11] Jean-Charles Faugère, Valérie Gauthier, Ayoub Otmani, Ludovic Perret, and Jean-Pierre Tillich. A distinguisher for high rate McEliece cryptosystems. In *Proc. IEEE Inf. Theory Workshop- ITW 2011*, pages 282–286, Paraty, Brasil, October 2011.

[FGO$^+$13] Jean-Charles Faugère, Valérie Gauthier, Ayoub Otmani, Ludovic Perret, and Jean-Pierre Tillich. A distinguisher for high rate McEliece cryptosystems. *IEEE Trans. Inform. Theory*, 59(10):6830–6844, October 2013.

[Fin10] Matthieu Finiasz. Parallel-CFS - strengthening the CFS McEliece-based signature scheme. In *Selected Areas in Cryptography 17th International Workshop, 2010, Waterloo, Ontario, Canada, August 12-13, 2010, revised selected papers*, volume 6544 of *LNCS*, pages 159–170. Springer, 2010.

[FS09] Matthieu Finiasz and Nicolas Sendrier. Security bounds for the design of code-based cryptosystems. In M. Matsui, editor, *Advances in Cryptology - ASIACRYPT 2009*, volume 5912 of *LNCS*, pages 88–105. Springer, 2009.

[GRS16] Philippe Gaborit, Olivier Ruatta, and Julien Schrek. On the complexity of the rank syndrome decoding problem. *IEEE Trans. Information Theory*, 62(2):1006–1019, 2016.

[GRSZ14] Philippe Gaborit, Olivier Ruatta, Julien Schrek, and Gilles Zémor. New results for rank-based cryptography. In *Progress in Cryptology - AFRICACRYPT 2014*, volume 8469 of *LNCS*, pages 1–12, 2014.

[GS12] Philippe Gaborit and Julien Schrek. Efficient code-based one-time signature from automorphism groups with syndrome compatibility. In *Proc. IEEE Int. Symposium Inf. Theory - ISIT 2012*, pages 1982–1986, Cambridge, MA, USA, July 2012.

[GSJB14] Danilo Gligoroski, Simona Samardjiska, Håkon Jacobsen, and Sergey Bezzateev. McEliece in the world of Escher. IACR Cryptology ePrint Archive, Report2014/360, 2014. `http://eprint.iacr.org/`.

[GZ16] Philippe Gaborit and Gilles Zémor. On the hardness of the decoding and the minimum distance problems for rank codes. *IEEE Trans. Information Theory*, 62(12):7245–7252, 2016.

[HT15] Adrien Hauteville and Jean-Pierre Tillich. New algorithms for decoding in the rank metric and an attack on the LRPC cryptosystem, 2015. abs/1504.05431.

[JJ02] Thomas Johansson and Fredrik Jönsson. On the complexity of some cryptographic problems based on the general decoding problem. *IEEE Trans. Inform. Theory*, 48(10):2669–2678, October 2002.

[KKS97] Gregory Kabatianskii, Ernst Krouk, and Ben. J. M. Smeets. A digital signature scheme based on random error-correcting codes. In *IMA Int. Conf.*, volume 1355 of *LNCS*, pages 161–167. Springer, 1997.

[KKS05] Gregory Kabatianskii, Ernst Krouk, and Ben. J. M. Smeets. *Error Correcting Coding and Security for Data Networks: Analysis of the Superchannel Concept.* John Wiley & Sons, 2005.

[Kor09] Satish Babu Korada. *Polar Codes for Channel and Source Coding.* PhD thesis, 'Ecole Polytechnique Fédérale de Lausanne (EPFL), July 2009.

[KT17] Ghazal Kachigar and Jean-Pierre Tillich. Quantum information set decoding algorithms. preprint, arXiv:1703.00263 [cs.CR], February 2017.

[LJ12] Carl Löndahl and Thomas Johansson. A new version of McEliece PKC based on convolutional codes. In *Information and Communications Security, ICICS*, volume 7168 of *LNCS*, pages 461–470. Springer, 2012.

[LT13] Grégory Landais and Jean-Pierre Tillich. An efficient attack of a McEliece cryptosystem variant based on convolutional codes. In P. Gaborit, editor, *Post-Quantum Cryptography'13*, volume 7932 of *LNCS*, pages 102–117. Springer, June 2013.

[MCT16] Irene Márquez-Corbella and Jean-Pierre Tillich. Using Reed-Solomon codes in the $(u|u+v)$ construction and an application to cryptography. preprint, 2016. arXiv:1601:08227.

[MMT11] Alexander May, Alexander Meurer, and Enrico Thomae. Decoding random linear codes in $O(2^{0.054n})$. In Dong Hoon Lee and Xiaoyun Wang, editors, *Advances in Cryptology - ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 107–124. Springer, 2011.

[MO15] Alexander May and Ilya Ozerov. On computing nearest neighbors with applications to decoding of binary linear codes. In E. Oswald and M. Fischlin, editors, *Advances in Cryptology - EUROCRYPT 2015*, volume 9056 of *LNCS*, pages 203–228. Springer, 2015.

[MS09] L. Minder and A. Sinclair. The extended *k*-tree algorithm. In C. Mathieu, editor, *Proceedings of SODA 2009*, pages 586–595. SIAM, 2009.

[Nie86] Harald Niederreiter. Knapsack-type cryptosystems and algebraic coding theory. *Problems of Control and Information Theory*, 15(2):159–166, 1986.

[OT11] Ayoub Otmani and Jean-Pierre Tillich. An efficient attack on all concrete KKS proposals. In *Post-Quantum Cryptography 2011*, volume 7071 of *LNCS*, pages 98–116, 2011.

[Pra62] Eugene Prange. The use of information sets in decoding cyclic codes. *IRE Transactions on Information Theory*, 8(5):5–9, 1962.

[PT16] Aurélie Phesso and Jean-Pierre Tillich. An efficient attack on a code-based signature scheme. In *Post-Quantum Cryptography 2016*, volume 9606 of *LNCS*, pages 86–103, Fukuoka, Japan, February 2016. Springer.

[Sen11] Nicolas Sendrier. Decoding one out of many. In *Post-Quantum Cryptography 2011*, volume 7071 of *LNCS*, pages 51–67, 2011.

[Sho04] Victor Shoup. Sequences of games: a tool for taming complexity in security proofs. *IACR Cryptology ePrint Archive*, 2004:332, 2004.

[SK14] Sujan Raj Shrestha and Young-Sik Kim. New McEliece cryptosystem based on polar codes as a candidate for post-quantum cryptography. In *2014 14th International Symposium on Communications and Information Technologies (ISCIT)*, pages 368–372. IEEE, 2014.

[Ste88] Jacques Stern. A method for finding codewords of small weight. In G. D. Cohen and J. Wolfmann, editors, *Coding Theory and Applications*, volume 388 of *LNCS*, pages 106–113. Springer, 1988.

[Ste93] Jacques Stern. A new identification scheme based on syndrome decoding. In D.R. Stinson, editor, *Advances in Cryptology - CRYPTO'93*, volume 773 of *LNCS*, pages 13–21. Springer, 1993.

[Wag02] David Wagner. A generalized birthday problem. In Moti Yung, editor, *Advances in Cryptology - CRYPTO 2002*, volume 2442 of *LNCS*, pages 288–303. Springer, 2002.

A Proofs for §3

A.1 List Emulation

In the security proof, we need to build lists of indices (salts) in $\mathbb{F}_2^{\lambda_0}$. Those lists have size q_{sign}, the maximum number of signature queries allowed to the adversary, a number which is possibly very large. For each message $\mathbf{m}$ which is either hashed or signed in the game we need to be able to

- create a list $L_{\mathbf{m}}$ of q_{sign} random elements of $\mathbb{F}_2^{\lambda_0}$, using the constructor `new list`,
- pick an element in $L_{\mathbf{m}}$, using the method $L_{\mathbf{m}}$`.next`, this element can be picked only once,
- decide whether or not a given salt $\mathbf{r}$ is in $L_{\mathbf{m}}$, using the method $L_{\mathbf{m}}$`.contains`.

The straightforward manner to achieve this is to draw q_{sign} random numbers when the list is constructed, this has to be done once for each different message $\mathbf{m}$ used in the game. This may result in a quadratic cost $q_{\text{hash}}q_{\text{sign}}$ just to build the lists. Once the lists are constructed, and assuming they are stored in a proper data structure (a heap for instance) picking an element or testing membership has a cost at most $O(\log q_{\text{sign}})$, that is at most linear in the security parameter λ.

```
class list
  elt, index
  list()
    index ← 0
    for $i = 1, \dots, q_{\text{sign}}$
      elt[$i$] ← randint($2^{\lambda_0}$)
```

```
method list.contains($\mathbf{r}$)
  return $\mathbf{r} \in \{\texttt{elt}[i], 1 \leq i \leq q_{\texttt{sign}}\}$
```

```
method list.next()
  index ← index + 1
  return elt[index]
```

Instead we may emulate the lists and never construct them explicitly as above. At any point in the game, we will denote by $S_{\mathbf{m}}$ the list of values of $\mathbf{r}$ such that `Sign`$(\mathbf{m})$ was queried and returned $\mathbf{r}$ and we will denote by $H_{\mathbf{m}}$ the list of values of $\mathbf{r}$ such that `Hash`$(\mathbf{m}, \mathbf{r})$ was queried, partitioned in $H_{\mathbf{m}}^{true}$ and $H_{\mathbf{m}}^{false}$ depending on the return value. The three attributes of the class `list`, namely `in`, `out` and `used`, will contain the elements of $H_{\mathbf{m}}^{true}$, $H_{\mathbf{m}}^{false}$ and $S_{\mathbf{m}}$ respectively. In all those lists elements may appear several times, however in the current setting, the game is aborted earlier (Game 1) whenever a salt is used more than once in a signature of $\mathbf{m}$, it follows that in the above lists no element can appear more than once.

```
class list
  in, out, used
  list()
    in ← ∅
    out ← ∅
    used ← ∅
```

```
method list.contains($\mathbf{r}$)
  if $\mathbf{r} \notin$ in ∪ out ∪ used
    if rand() $\leq \beta$
      in.push($\mathbf{r}$)
    else
      out.push($\mathbf{r}$)
  return $\mathbf{r} \in$ in ∪ used
```

```
method list.next()
  if rand() $\leq \gamma$
    $\mathbf{r}$ ← in.pop()
  else
    $\mathbf{r} \hookleftarrow \mathbb{F}_2^{\lambda_0} \setminus$ (in ∪ out ∪ used)
  used.push($\mathbf{r}$)
  return $\mathbf{r}$
```

All `push`, `pop`, membership testing above can be implemented in time proportional to λ_0. The method `pop` removes randomly an element from the list and returns it. The method `push` adds an element in a list. The procedure `rand`() picks uniformly a real number between 0 and 1. We denote

$$\beta = 1 - \left(1 - \frac{1}{2^{\lambda_0} - |H_{\mathbf{m}}| - |S_{\mathbf{m}}|}\right)^{q_{\text{sign}} - |H_{\mathbf{m}}^{true}| - |S_{\mathbf{m}}|} \quad \text{and } \gamma = \frac{|H_{\mathbf{m}}^{true}|}{q_{\text{sign}} - |S_{\mathbf{m}}|}.$$

For the emulation to be correct we need the following to hold

Proposition 11. *At any point of the game, for any message* $\mathbf{m}$*, and for all* $\mathbf{r} \in \mathbb{F}_2^{\lambda_0}$*, we have*

(i) $\mathbb{P}[L_{\mathbf{m}}.\texttt{contains}(\mathbf{r})] = \mathbb{P}[\mathbf{r} \in L_{\mathbf{m}}]$

(ii) $\mathbb{P}[\mathbf{r} \leftarrow L_{\mathbf{m}}.\texttt{next}()] = \mathbb{P}[\mathbf{r} \hookleftarrow L_{\mathbf{m}} \setminus S_{\mathbf{m}}]$

Proof. First recall that the game will abort if at some point the same $\mathbf{r}$ is returned for signing $\mathbf{m}$. It follows that $H_{\mathbf{m}}^{true}$, $H_{\mathbf{m}}^{false}$ and $S_{\mathbf{m}}$ are disjoint and never contain duplicate elements.

(i) If $\mathbf{r} \in H_{\mathbf{m}}^{true} \cup S_{\mathbf{m}}$ both probabilities equal 1 and if $\mathbf{r} \in H_{\mathbf{m}}^{false}$ both probabilities equal 0.
If $\mathbf{r} \notin S_{\mathbf{m}} \cup H_{\mathbf{m}}$. There are $q_{\text{sign}} - |H_{\mathbf{m}}^{true}| - |S_{\mathbf{m}}|$ slots are available in $L_{\mathbf{m}}$ and $2^{\lambda_0} - |H_{\mathbf{m}}| - |S_{\mathbf{m}}|$ possible values remaining. Thus

$$\mathbb{P}[\mathbf{r} \in L_{\mathbf{m}}] = 1 - \left(1 - \frac{1}{2^{\lambda_0} - |H_{\mathbf{m}}| - |S_{\mathbf{m}}|}\right)^{q_{\text{sign}} - |H_{\mathbf{m}}^{true}| - |S_{\mathbf{m}}|} = \beta = \mathbb{P}[L_{\mathbf{m}}.\texttt{contains}(\mathbf{r})]$$

(ii) If $\mathbf{r} \in H_{\mathbf{m}}^{false} \cup S_{\mathbf{m}}$ both probabilities equal 0.
If $\mathbf{r} \in H_{\mathbf{m}}^{true}$, then $\mathbf{r} \in L_{\mathbf{m}} \setminus S_{\mathbf{m}}$ and

$$\mathbb{P}[\mathbf{r} \hookleftarrow L_{\mathbf{m}} \setminus S_{\mathbf{m}}] = \frac{1}{q_{\text{sign}} - |S_{\mathbf{m}}|}$$

In the emulated call $L_{\mathbf{m}}.\texttt{next}()$, if $\mathbf{r} \in H_{\mathbf{m}}^{true}$ it is returned with probability $\gamma / |H_{\mathbf{m}}^{true}|$, which is the same.
Finally, if $\mathbf{r} \notin H_{\mathbf{m}} \cup S_{\mathbf{m}}$, the value $\mathbf{r}$ is among the $2^{\lambda_0} - |H_{\mathbf{m}}| - |S_{\mathbf{m}}|$ values that have not been seen yet by the algorithm and all those values are equally likely to be drawn in both cases.

A.2 Proof of Lemma 1

The goal of this subsection is to estimate the probability of a collision in a signature query for a message $\mathbf{m}$ when we allow at most q_{sign} queries (the event F in the security proof) and to deduce Lemma 1 of §3.3. We recall that in $\mathscr{S}_{code}$ for each signature query, we pick $\mathbf{r}$ uniformly at random in $\{0,1\}^{\lambda_0}$. Then the probability we are looking for is bounded by the probability to pick the same $\mathbf{r}$ at least twice after q_{sign} draws. The following lemma will be useful.

Lemma 4. *The probability to have at least one collision after drawing uniformly and independently t elements in a set of size n is upper bounded by t^2/n for sufficiently large n and $t^2 < n$.*

Proof. The probability of no collisions after drawing independently t elements among n is:

$$p_{n,t} \overset{\triangle}{=} \prod_{i=0}^{t-1} \left(1 - \frac{i}{n}\right) \geq 1 - \sum_{i=0}^{t-1} \frac{i}{n} = 1 - \frac{t(t-1)}{2n}$$

from which we easily get $1 - p_{n,t} \leq t^2/n$, concluding the proof.

In our case, the probability of the event F is bounded by the previous probability for $t = q_{\text{sign}}$ and $n = 2^{\lambda_0}$, so, with $\lambda_0 = \lambda + 2\log_2 q_{\text{sign}}$, we can conclude that

$$\mathbb{P}(F) \leq \frac{q_{\text{sign}}^2}{2^{\lambda_0}} = \frac{1}{2^{\lambda_0 - 2\log_2(q_{\text{sign}})}} = \frac{1}{2^{\lambda}}$$

which concludes the proof of Lemma 1.

A.3 Proof of Lemma 2

Our goal in this subsection is to prove Lemma 2 of §3.3 which asserts that syndromes by $\mathbf{H}_{\text{pub}}$ of errors of weight w are indistinguishable from random elements in $\mathbb{F}_2^{n-k}$:

Lemma 2.

$$\mathbb{P}(S_1) \leq \mathbb{P}(S_2) + \frac{1}{2} q_{hash} \sqrt{\frac{2^{n-k}}{\binom{n}{w}}} + 2\rho_c\left(\mathscr{D}^{\mathscr{F}}, \mathscr{D}^{\mathscr{R}}\right)\left(t \cdot O(n^2)\right)$$

Let us first introduce two distributions:

- $\mathscr{F}_0$ is defined as the distribution of $\{(\mathbf{H}, \mathbf{H}\mathbf{e}_j^T), 1 \leq j \leq q\}$ where the $\mathbf{e}_j$'s are drawn uniformly at random in S_w and $\mathbf{H}$ is drawn uniformly at random in $\mathscr{F}$.
- The distribution $\mathscr{F}_1$ is defined as the distribution of $\{(\mathbf{H}, \mathbf{s}_j^T), 1 \leq j \leq q\}$ where the $\mathbf{s}_j$'s are drawn uniformly at random in $\mathbb{F}_2^{n-k}$ and $\mathbf{H}$ is drawn uniformly at random in $\mathscr{F}$.
- Similarly, we define the distributions $\mathscr{R}_0$ and $\mathscr{R}_1$ for $\mathbf{H} \hookleftarrow \mathbb{F}_2^{(n-k)\times n}$.

We now observe that (we use here notations of the security proof in §3.3)

$$|\mathbb{P}(S_1) - \mathbb{P}(S_2)| \leq \rho_c\left(\mathscr{F}_0, \mathscr{F}_1\right)(t). \tag{18}$$

Moreover, from the metric-like properties of the computational distance we have:

$$\rho_c\left(\mathscr{F}_0, \mathscr{F}_1\right)(t) \leq \rho_c\left(\mathscr{F}_0, \mathscr{R}_0\right)(t) + \rho_c\left(\mathscr{R}_0, \mathscr{R}_1\right)(t) + \rho_c\left(\mathscr{R}_1, \mathscr{F}_1\right)(t). \tag{19}$$

We upper-bound the first term and the last term of the sum through the following lemma.

Lemma 5.

$$\rho_c\left(\mathscr{F}_0, \mathscr{R}_0\right)(t) \leq \rho_c\left(\mathscr{D}^{\mathscr{F}}, \mathscr{D}^{\mathscr{R}}\right)\left(t + O(qn^2)\right) \tag{20}$$

$$\rho_c\left(\mathscr{F}_1, \mathscr{R}_1\right)(t) \leq \rho_c\left(\mathscr{D}^{\mathscr{F}}, \mathscr{D}^{\mathscr{R}}\right)\left(t + O(qn)\right). \tag{21}$$

Proof. We construct an algorithm $\mathscr{A}'$ with advantage $\rho_c\left(\mathscr{F}_0, \mathscr{R}_0\right)(t)$ distinguishing between $\mathscr{D}^{\mathscr{F}}$ and $\mathscr{D}^{\mathscr{R}}$ from an algorithm $\mathscr{A}$ of complexity $\leq t$ distinguishing with advantage $\rho_c\left(\mathscr{F}_0, \mathscr{R}_0\right)(t)$ between $\mathscr{F}_0$ and $\mathscr{R}_0$ as follows. From the sample $\mathbf{H}$ that we have, we construct q syndromes $\mathbf{H}\mathbf{e}_j^T$'s by drawing uniformly at random q samples $\mathbf{e}_1, \ldots, \mathbf{e}_q$ from S_w. This takes time $O(qn^2)$. Then we apply $\mathscr{A}$ on the sample $\{(\mathbf{H}, \mathbf{s}_j^T), 1 \leq j \leq q\}$. This is our algorithm $\mathscr{A}'$ for distinguishing between $\mathscr{D}^{\mathscr{F}}$ and $\mathscr{D}^{\mathscr{R}}$. The advantage of this algorithm is by definition smaller than $\rho_c\left(\mathscr{D}^{\mathscr{F}}, \mathscr{D}^{\mathscr{R}}\right)\left(t + O(qn^2)\right)$. This shows (20). (21) is proved in a similar fashion. □

We are now going to bound $\rho_c\left(\mathscr{R}_0, \mathscr{R}_1\right)$. We start by using Theorem 1

$$\rho_c\left(\mathscr{R}_0, \mathscr{R}_1\right)(t) \leq \rho\left(\mathscr{R}_0, \mathscr{R}_1\right).$$

Let us bring in for this purpose two auxiliary distributions. Let us first define $\mathscr{R}_0'$ as the distribution of

$$\left(\mathbf{H}, \mathbf{H}\mathbf{e}^T\right) \text{ where } \mathbf{H} \hookleftarrow \mathbb{F}_2^{n-k} \text{ and } \mathbf{e} \hookleftarrow S_w$$

whereas $\mathscr{R}_1'$ is the distribution of

$$(\mathbf{H}, \mathbf{s}) \text{ where } \mathbf{H} \hookleftarrow \mathbb{F}_2^{n-k} \text{ and } \mathbf{s} \hookleftarrow \mathbb{F}_2^{n-k}.$$

These distributions are very similar to $\mathscr{R}_0$ and $\mathscr{R}_1$. The distribution $\mathscr{R}_i$ is in the proof of security a sample of q_{hash} independent syndromes obtianed from the same matrix $\mathbf{H}$. We can use here Proposition 3 and obtain

$$\rho\left(\mathscr{R}_0, \mathscr{R}_1\right) \leq q_{\text{hash}}\, \rho\left(\mathscr{R}_0', \mathscr{R}_1'\right). \tag{22}$$

We are now going to bound $\rho\left(\mathscr{R}_0', \mathscr{R}_1'\right)$ with the leftover hash lemma. For this purpose, let us recall the following definition.

Definition 14 (Universal Hashing). *A family $\mathscr{H}$ of deterministic functions $h : \mathscr{X} \to \{0,1\}^v$ is a called a universal hash family if for any $x_1 \neq x_2 \in \mathscr{X}$ we have $\mathbb{P}_{h \hookleftarrow \mathscr{H}}\left(h(x_1) = h(x_2)\right) \leq \frac{1}{2^v}$.*

In our case, we define the family $\mathscr{H}$ as the set $\left\{\mathbf{H} : \mathbf{H} \in \mathbb{F}_2^{(n-k)\times n}\right\}$ acting on S_v:

$$\begin{aligned} \mathbf{H} :\ S_w &\longrightarrow \mathbb{F}_2^{n-k}. \\ \mathbf{e} &\mapsto \mathbf{H}\mathbf{e}^T \end{aligned}$$

Lemma 7 in §C shows that

$$\mathbb{P}_{\mathbf{H}\hookleftarrow\mathbb{F}_2^{n-k}}\left(\mathbf{H}\mathbf{e}_1^T = \mathbf{H}\mathbf{e}_2^T\right) = \frac{1}{2^{n-k}}.$$

It follows that $\mathscr{H}$ is a universal hash family. We recall now the following lemma which is known as the leftover hash Lemma (see for instance [BDK+11]).

Lemma 6. *Let $\mathscr{H}$ be a universal hash family of functions $h : \mathscr{X} \to \{0,1\}^v$. Let $\mathscr{D}$ be the distribution of $h(X)$ where h is drawn uniformly at random in $\mathscr{H}$ and X is a random variable taking its values in $\mathscr{X}$. Let $m \stackrel{\triangle}{=} -\log_2\left(\max_{x\in\mathscr{X}} \mathbb{P}(X=x)\right)$ be the min entropy of X. Then*

$$\rho(\mathscr{D}, \mathscr{U}_v) \leq \frac{1}{2}\sqrt{\frac{2^v}{2^m}}.$$

Let X be a random variable whose distribution is $\mathscr{U}_w$ and let $\mathscr{D}$ be the distribution of $\mathbf{H}X^T$ where $\mathbf{H} \hookleftarrow \mathbb{F}_2^{(n-k)\times n}$. We observe that

$$\rho\left(\mathscr{R}'_0, \mathscr{R}'_1\right) = \rho\left(\mathscr{D}, \mathscr{U}_{n-k}\right).$$

We can therefore apply the leftover hash lemma. For this, let us compute the min entropy of X. It is readily verified that it is equal to

$$-\log_2\left(\frac{1}{\binom{n}{w}}\right) = \log_2\binom{n}{w}.$$

which leads to the conclusion:

$$\rho\left(\mathscr{R}'_0, \mathscr{R}'_1\right) \leq \frac{1}{2}\sqrt{\frac{2^{n-k}}{\binom{n}{w}}} \tag{23}$$

By plugging results of Lemma 5, (19), (22) and (23) in (18) we get:

$$\mathbb{P}(S_1) \leq \mathbb{P}(S_2) + \frac{1}{2}q_{\text{hash}}\sqrt{\frac{2^{n-k}}{\binom{n}{w}}} + 2\rho_c\left(\mathscr{D}^{\mathscr{F}}, \mathscr{D}^{\mathscr{R}}\right)\left(t + O(qn^2)\right)$$

which concludes the proof.

B Proofs for §4

B.1 Proof of Propositions 5 and 6

Let us recall Proposition 5

Proposition 5. *Let $p(i) \stackrel{\triangle}{=} \mathbb{P}_{\mathbf{y},\theta}(|\varphi(V,\mathbf{y})| = i)$. If two executions of φ are independent, then for all i in $\{0,\dots,w\}$ such that $w-i \equiv 0 \pmod 2$ we have*

$$p_2^{sdd}\left(\frac{w-i}{2}\right) = p_1^{sdd}(i) = \frac{x_i\, p(i)}{p_w^1} \tag{4}$$

where

$$p_w^1 \stackrel{\triangle}{=} \sum_{\substack{0\leq j\leq w\\ j\equiv w \pmod 2}} x_j\, p(j)$$

and $p_1^{sdd}(i) = 0$ for other choices of i.

Proof. Let $\mathbf{e}$ be the output of Algorithm 2. Recall that

$$p_1^{sdd}(j) \stackrel{\triangle}{=} \mathbb{P}\left(w_1(\mathbf{e}) = j\right).$$

As two executions of φ are independent, by a disjunction of independent events the probability to get an error $\mathbf{e}$ such that $w_1(\mathbf{e}) = i$ is given by:

$$\sum_{l=0}^{+\infty} \alpha^l \beta_i = \frac{\beta_i}{1-\alpha} \tag{24}$$

where α denotes the probability that the output of φ at Instruction 2 of Algorithm 2 is rejected and β_i the probability to have an error of weight i which is accepted. These probabilities are readily seen to be equal to:

$$\beta_i = p(i)x_i \quad ; \quad \alpha = 1 - \sum_{\substack{0 \le j \le w \\ j \equiv w \pmod 2}} x_j p(j).$$

Plugging this expression in (24) finishes the proof. □

Let us recall now Proposition 6

Proposition 6. *If the source decoder φ used in Algorithm 2 behaves uniformly for V and uniformly for $\mathrm{Punc}_{\mathbf{e}_V}(U)$ for all error patterns $\mathbf{e}_V$ obtained as $\mathbf{e}_V = \varphi(V, \mathbf{y}_1 + \mathbf{y}_2)$, we have:*

$$\rho\left(\mathscr{D}_w, \mathscr{U}_w\right) = \rho\left(p_1^{sdd}, p_1^u\right)$$

The output of Algorithm 2 is the uniform distribution over S_w if in addition two executions of φ are independent and the no-rejection probability vector $\mathbf{x}$ is chosen for any i in $\{0, \dots, w\}$ as

$$x_i = \frac{1}{M_{rs}} \frac{p_1^u(i)}{p(i)} \text{ if } w \equiv i \pmod 2$$

and 0 otherwise with $M_{rs} \stackrel{\triangle}{=} \sup_{\substack{0 \le i \le w \\ i \equiv w \pmod 2}} \frac{p_1^u(i)}{p(i)}$.

Proof. Let us first introduce some notation. Let $\mathbf{e}$ be a random variable whose distribution is $\mathscr{U}_w$, i.e. the uniform distribution over S_w, and let $\tilde{\mathbf{e}}$ be a random variable whose distribution is $\mathscr{D}_w$. The last random variable can be viewed in a natural way as the ouput of Algorithm 2 and is of the form $\tilde{\mathbf{e}} = (\mathbf{e}_U | \mathbf{e}_U + \mathbf{e}_V)$. We view $\mathbf{e}_U$ and $\mathbf{e}_V$ as random variables. We have

$$\rho\left(\mathscr{D}_w, \mathscr{U}_w\right) = \sum_{\mathbf{e}_1, \mathbf{e}_2 \in \mathbb{F}_2^{n/2} : |(\mathbf{e}_1 | \mathbf{e}_2)| = w} \left| \mathbb{P}(\tilde{\mathbf{e}} = (\mathbf{e}_1 | \mathbf{e}_2)) - \mathbb{P}(\mathbf{e} = (\mathbf{e}_1 | \mathbf{e}_2)) \right|. \tag{25}$$

We notice now that

$$\begin{aligned} \mathbb{P}(\tilde{\mathbf{e}} = (\mathbf{e}_1 | \mathbf{e}_2)) &= \mathbb{P}(\mathbf{e}_U = \mathbf{e}_1 | \mathbf{e}_V = \mathbf{e}_1 + \mathbf{e}_2) \mathbb{P}(\mathbf{e}_V = \mathbf{e}_1 + \mathbf{e}_2) \\ &= \mathbb{P}(\mathbf{e}_U = \mathbf{e}_1 | \mathbf{e}_V = \mathbf{e}_1 + \mathbf{e}_2) \mathbb{P}_{(\mathbf{y}_1, \mathbf{y}_2), \theta}(\varphi(V, \mathbf{y}_1 + \mathbf{y}_2) = \mathbf{e}_1 + \mathbf{e}_2) \end{aligned} \tag{26}$$

From the assumption on the uniform behavior of φ we deduce that $\mathbb{P}_{(\mathbf{y}_1, \mathbf{y}_2), \theta}(\varphi(V, \mathbf{y}_1 + \mathbf{y}_2) = \mathbf{e}_1 + \mathbf{e}_2)$ only depends on the Hamming weight $|\mathbf{e}_1 + \mathbf{e}_2|$ of $\mathbf{e}_1 + \mathbf{e}_2$. We recall now that in Algorithm 2 we have

$$\begin{aligned} \mathbf{e}_U &= \mathbf{y}_1 + \mathbf{u} \text{ where} \\ \mathbf{u} &= \underset{\mathbf{e}_V}{\overset{U}{\mathrm{Comp}}}(\underset{\mathbf{e}_V}{\mathrm{Punc}}(\mathbf{y}_1) + \mathbf{e}'_U) \text{ with} \\ \mathbf{e}'_U &= \varphi_{(w - |\mathbf{e}_V|)/2}(\underset{\mathbf{e}_V}{\mathrm{Punc}}(U), \underset{\mathbf{e}_V}{\mathrm{Punc}}(\mathbf{y}_1)). \end{aligned}$$

Let

$$\begin{aligned} n' &= n/2 - |\mathbf{e}| \\ w' &\overset{\triangle}{=} \frac{w - |\mathbf{e}|}{2} \\ U' &\overset{\triangle}{=} \underset{\mathbf{e}_V}{\mathrm{Punc}}(U). \end{aligned}$$

It will also be convenient to split $\mathbf{y}_1$, $\mathbf{e}_U$ and $\mathbf{e}_1$ into two parts: the first one, denoted respectively by $\mathbf{y}_1'$, $\mathbf{e}_U'$, and $\mathbf{e}_1'$ is the restriction of these vectors to the complement of the support of $\mathbf{e}_V$, whereas the second one, denoted respectively by $\mathbf{y}_1''$, $\mathbf{e}_U''$ and $\mathbf{e}_1''$ is the restriction of these vectors to the support of $\mathbf{e}_V$. With this notation, we now notice that

$$\begin{aligned} \mathbb{P}(\mathbf{e}_U = \mathbf{e}_1 | \mathbf{e}_V = \mathbf{e}_1 + \mathbf{e}_2) &= \mathbb{P}_{\mathbf{y}_1,\mathbf{y}_2,\theta}(\mathbf{e}_U' = \mathbf{e}_1', \mathbf{e}_U'' = \mathbf{e}_1'' | \mathbf{e}_V = \mathbf{e}_1 + \mathbf{e}_2) \\ &= \mathbb{P}_{\mathbf{y}_1',\theta}(\mathbf{e}_U' = \mathbf{e}_1')\mathbb{P}_{\mathbf{y}_1''}(\mathbf{e}_U'' = \mathbf{e}_1'') \\ &= \mathbb{P}_{\mathbf{y}_1',\theta}(\varphi_{w'}(U', \mathbf{y}_1') = \mathbf{e}_1')\mathbb{P}_{\mathbf{y}_1''}(\mathbf{e}_U'' = \mathbf{e}_1'') \\ &= \frac{1}{\binom{n'}{w'}} \frac{1}{2^{n/2-n'}}. \end{aligned} \tag{27}$$

The last equality follows from the fact that φ behaves uniformly on U' and therefore the output of $\varphi_{w'}(U', \mathbf{y}_1')$ is the uniform distribution over the set of words of weight w' in $\mathbb{F}_2^{n'}$. (27) implies that $\mathbb{P}(\mathbf{e}_U = \mathbf{e}_1 | \mathbf{e}_V = \mathbf{e}_1 + \mathbf{e}_2)$ only depends on the weight of w' which itself only depends on the weight of $\mathbf{e}_1 + \mathbf{e}_2$. Since $\mathbb{P}_{(\mathbf{y}_1,\mathbf{y}_2),\theta}(\varphi(V, \mathbf{y}_1 + \mathbf{y}_2) = \mathbf{e}_1 + \mathbf{e}_2)$ has the same property, we deduce from (26), that $\mathbb{P}(\tilde{\mathbf{e}} = (\mathbf{e}_1|\mathbf{e}_2))$ only depends on the weight of $\mathbf{e}_1 + \mathbf{e}_2$. Obviously $\mathbb{P}(\mathbf{e} = (\mathbf{e}_1|\mathbf{e}_2))$ also has this property. We may therefore write

$$\begin{aligned} \mathbb{P}(\tilde{\mathbf{e}} = (\mathbf{e}_1|\mathbf{e}_2)) &= f(|\mathbf{e}_1 + \mathbf{e}_2)|) \\ \mathbb{P}(\mathbf{e} = (\mathbf{e}_1|\mathbf{e}_2)) &= g(|\mathbf{e}_1 + \mathbf{e}_2)|) \end{aligned}$$

for some functions f and g. Plugging these expressions in (25) yields by bringing in the quantity m_i which is the number of $\mathbf{e}$ in S_w such that $w_1(\mathbf{e}) = i$:

$$\begin{aligned} \rho\left(\mathscr{D}_w, \mathscr{U}_w\right) &= \sum_{\substack{0 \le i \le w \\ i \equiv w \pmod 2}} \sum_{\mathbf{m} \in S_w | w_1(\mathbf{m}) = i} |\mathbb{P}(\tilde{\mathbf{e}} = \mathbf{m}) - \mathbb{P}(\mathbf{e} = \mathbf{m})| \\ &= \sum_{\substack{0 \le i \le w \\ i \equiv w \pmod 2}} m_i \left|f(i) - g(i)\right| \\ &= \sum_{\substack{0 \le i \le w \\ i \equiv w \pmod 2}} |m_i(f(i) - g(i))| \\ &= \sum_{\substack{0 \le i \le w \\ i \equiv w \pmod 2}} |\mathbb{P}(w_1(\tilde{\mathbf{e}}) = i) - \mathbb{P}(w_1(\mathbf{e}) = i)| \\ &= \rho(p_1^{sdd} - p_1^u). \end{aligned}$$

The last part of the proposition follows from the fact that the $p_1^{sdd}(i)$'s are functions of the non-rejection probability vector $\mathbf{x} = (x_i)$. Thanks to what we just proved, we can compute the x_i's to have $\rho\left(\mathscr{D}_w^1, \mathscr{U}_w^1\right) = 0$. This will imply that the output of Algorithm 2 is the uniform distribution. Indeed, we first notice that for all i:

$$0 \le x_i = \frac{1}{M_{rs}} \frac{p_1^u(i)}{p(i)} = \left(\inf_{\substack{0 \le j \le w \\ w \equiv j \pmod 2}} \frac{p(j)}{p_1^u(j)} \right) \frac{p_1^u(i)}{p(i)} \le \frac{p(i)}{p_u^1(i)} \frac{p_1^u(i)}{p(i)} = 1$$

which allows to assert that $\mathbf{x}$ is a probability vector. We now use the following equalities for all i:

$$\begin{aligned} p_1^{sdd}(i) &= \frac{x_i\, p(i)}{p_w^1} \\ &= \frac{p_1^u(i)}{M_{rs} \sum\limits_{\substack{0 \le j \le w \\ w \equiv j \pmod 2}} \frac{1}{M_{rs}} p_1^u(j)} \\ &= p_1^u(i). \end{aligned}$$

where the last line relies on the equality $\sum\limits_{\substack{0 \le j \le w \\ w \equiv j \pmod 2}} p_1^u(j) = 1$. □

B.2 Proof of Proposition 7

Here the internal coins are over the choices of the $n-k$ positions (columns of the parity-check matrix $\mathbf{H}$ of $\mathscr{C}$) I we choose to invert in the Prange algorithm the square submatrix of $\mathbf{H}$. We have here

$$\begin{aligned} p(w) &= \sum_{\mathbf{e}:|\mathbf{e}|=w} \mathbb{P}_{\mathbf{y},I}\left(\mathbf{e} = \varphi^{\text{Prange}}(\mathscr{C},\mathbf{y})\right) \\ &= \sum_{I \subset \{1,\dots,n\}:|I|=n-k} \mathbb{P}(I) \sum_{\mathbf{e}:|\mathbf{e}|=w} \mathbb{P}_{\mathbf{y}}(\mathbf{e} = \varphi^{\text{Prange}}(\mathscr{C},\mathbf{y})|I) \\ &= \sum_{I \subset \{1,\dots,n\}:|I|=n-k} \mathbb{P}(I) \sum_{\mathbf{e}:|\mathbf{e}|=w,\mathrm{Supp}(\mathbf{e}) \subset I} \frac{1}{2^{n-k}} \\ &= \frac{\binom{n-k}{w}}{2^{n-k}}. \end{aligned}$$

C Proof of Proposition 8 in §6

Let us recall Proposition 8

Proposition 8. *Assume that we choose a $(U|U+V)$ code by picking the parity-check matrices of U and V uniformly at random among the binary matrices of size $(n/2-k_U) \times n/2$ and $(n/2-k_V) \times n/2$ respectively. Let $a_{(U|U+V)}(w)$, $a_{(U|U)}(w)$ and $a_{(0|V)}(w)$ be the expected number of codewords of weight w that are respectively in the $(U|U+V)$ code, of the form $(\mathbf{u}|\mathbf{u})$ where $\mathbf{u}$ belongs to U and of the form $(\mathbf{0}|\mathbf{v})$ where $\mathbf{v}$ belongs to V. These numbers are given for even w in $\{0,\dots,n\}$ by*

$$a_{(U|U+V)}(w) = \frac{\binom{n/2}{w/2}}{2^{n/2-k_U}} + \frac{\binom{n/2}{w}}{2^{n/2-k_V}} + \frac{1}{2^{n-k_U-k_V}} \left(\binom{n}{w} - \binom{n/2}{w} - \binom{n/2}{w/2} \right)$$

$$a_{(U|U)}(w) = \frac{\binom{n/2}{w/2}}{2^{n/2-k_U}} \quad ; \quad a_{(0|V)}(w) = \frac{\binom{n/2}{w}}{2^{n/2-k_V}}$$

and for odd w in $\{0,\dots,n\}$ by

$$a_{(U|U+V)}(w) = \frac{\binom{n/2}{w}}{2^{n/2-k_V}} + \frac{1}{2^{n-k_U-k_V}} \left(\binom{n}{w} - \binom{n/2}{w} \right)$$

$$a_{(U|U)}(w) = 0 \quad ; \quad a_{(0|V)}(w) = \frac{\binom{n/2}{w}}{2^{n/2-k_V}}$$

On the other hand, when we choose a code of length n with a random parity-check matrix of size $(n-k_U-k_V)\times n$ chosen uniformly at random, then the expected number $a(w)$ of codewords of weight $w>0$ is given by

$$a(w)=\frac{\binom{n}{w}}{2^{n-k_U-k_V}}.$$

We will need the following lemma.

Lemma 7. *Let* $\mathbf{y}$ *be a non-zero vector of* $\mathbb{F}_2^n$ *and* $\mathbf{s}$ *an arbitrary element in* $\mathbb{F}_2^r$*. We choose a matrix* $\mathbf{H}$ *of size* $r\times n$ *uniformly at random among the set of* $r\times n$ *binary matrices. In this case*

$$\mathbb{P}\left(\mathbf{H}\mathbf{y}^T=\mathbf{s}^T\right)=\frac{1}{2^r}$$

Proof. The coefficient of $\mathbf{H}$ at row i and column j is denoted by h_{ij}, whereas the coefficients of $\mathbf{y}$ and $\mathbf{s}$ are denoted by y_i and s_i respectively. The probability we are looking for is the probability to have

$$\sum_j h_{ij}y_j=s_i \tag{28}$$

for all i in $\{1,\dots,r\}$. Since $\mathbf{y}$ is non zero, it has at least one non-zero coordinate. Without loss of generality, we may assume that $y_1=1$. We may rewrite (28) as $h_{i1}=\sum_{j>1}h_{ij}y_j$. This event happens with probability $\frac{1}{2}$ for a given i and with probability $\frac{1}{2^r}$ on all r events simultaneously due to the independence of the h_{ij}'s.

The last part of Proposition 8 is a direct application of this lemma. We namely have

Proposition 12. *Let $a(w)$ be the expected number of codewords in a binary linear code $\mathscr{C}$ of length n whose parity-check matrix is chosen* $\mathbf{H}$ *uniformly at random among all binary matrices of size $r\times n$. We have*

$$a(w)=\frac{\binom{n}{w}}{2^{n-r}}.$$

Proof. Let $Z\overset{\triangle}{=}\sum_{\mathbf{x}\in\mathbb{F}_2^n:|\mathbf{x}|=w}Z_{\mathbf{x}}$ where $Z_{\mathbf{x}}$ is the indicator function of the event "$\mathbf{x}$ is in $\mathscr{C}$". We have

$$\begin{aligned}
a(w)&=\mathbb{E}(Z)\\
&=\sum_{\mathbf{x}\in\mathbb{F}_2^n:|\mathbf{x}|=w}\mathbb{E}(Z_{\mathbf{x}})\\
&=\sum_{\mathbf{x}\in\mathbb{F}_2^n:|\mathbf{x}|=w}\mathbb{P}(\mathbf{x}\in\mathscr{C})\\
&=\sum_{\mathbf{x}\in\mathbb{F}_2^n:|\mathbf{x}|=w}\mathbb{P}(\mathbf{H}\mathbf{x}^T=0)\\
&=\sum_{\mathbf{x}\in\mathbb{F}_2^n:|\mathbf{x}|=w}\frac{1}{2^r}\\
&=\frac{\binom{n}{w}}{2^{n-r}}.
\end{aligned}$$

This proves the part of Proposition 8 dealing with the expected weight distribution of a random linear code. We are ready now to prove Proposition 8 concerning the expected weight distribution of a random $(U|U+V)$ code.

Weight distributions of $(U|U)\overset{\triangle}{=}\{(\mathbf{u}|\mathbf{u}):\mathbf{u}\in U\}$ **and** $(0|V)\overset{\triangle}{=}\{(0|\mathbf{v}):\mathbf{v}\in V\}$. This follows directly from Proposition 12 since $a_{(U,U)}(w)=0$ for odd and $a_{(U,U)}(w)$ is equal to the expected number of codewords of weight $w/2$ in a random linear code of length $n/2$ with a parity-check matrix of size $(n/2-k_U)\times n/2$ when w is even. On the other hand $a_{(0|V)}$ is equal to the expected number of weight w in a random linear code of length $n/2$ and with a parity-check matrix of size $(n/2-k_V)\times n/2$. In other words

$$\begin{aligned}
a_{(U|U)}(w) &= 0 \text{ if } w \text{ is odd}\\
a_{(U|U)}(w) &= \frac{\binom{n/2}{w/2}}{2^{n/2-k_U}} \text{ if } w \text{ is even}\\
a_{(0|V)}(w) &= \frac{\binom{n/2}{w}}{2^{n/2-k_V}}
\end{aligned}$$

Weight distributions of $(U|U+V)$. $(U|U+V)$ is chosen randomly by picking up a parity-check matrix $\mathbf{H}_U$ of U uniformly at random among the set of $(n/2-k_U)\times n/2$ binary matrices and a parity-check matrix $\mathbf{H}_V$ of V uniformly at random among the set of $(n/2-k_V)\times n/2$ binary matrices. Let $Z \overset{\triangle}{=} \sum_{\mathbf{x}\in\mathbb{F}_2^n:|\mathbf{x}|=w} Z_{\mathbf{x}}$ where $Z_{\mathbf{x}}$ is the indicator function of the event "$\mathbf{x}$ is in $(U|U+V)$".

We have

$$\begin{aligned}
a_{(U|U+V)}(w) &= \mathbb{E}(Z)\\
&= \sum_{\mathbf{x}\in\mathbb{F}_2^n:|\mathbf{x}|=w} \mathbb{E}(Z_{\mathbf{x}})\\
&= \sum_{\mathbf{x}\in\mathbb{F}_2^n:|\mathbf{x}|=w} \mathbb{P}(Z_{\mathbf{x}}=1)\\
&= \sum_{\mathbf{x}\in\mathbb{F}_2^n:|\mathbf{x}|=w} \mathbb{P}(\mathbf{x}\in(U|U+V)) \qquad (29)
\end{aligned}$$

By writing $\mathbf{x}=(\mathbf{x}_1|\mathbf{x}_2)$ where $\mathbf{x}_i$ is in $\mathbb{F}_2^{n/2}$ we know that $\mathbf{x}$ is in $(U|U+V)$ if and only if at the same time $\mathbf{x}_1$ is in U and $\mathbf{x}_2+\mathbf{x}_1$ is in V, that is

$$\mathbf{H}_U\mathbf{x}_1^T = 0,\ \ \mathbf{H}_V\mathbf{x}_1^T = \mathbf{H}_V\mathbf{x}_2^T.$$

There are three cases to consider

Case 1: $\mathbf{x}_1=0$ and $\mathbf{x}_2\neq 0$. In this case

$$\mathbb{P}(\mathbf{x}\in(U|U+V)) = \mathbb{P}(\mathbf{H}_V\mathbf{x}_2^T=0) = \frac{1}{2^{n/2-k_V}} \qquad (30)$$

Case 2: $\mathbf{x}_1=\mathbf{x}_2$. In this case

$$\mathbb{P}(\mathbf{x}\in(U|U+V)) = \mathbb{P}(\mathbf{H}_U\mathbf{x}_1^T=0) = \frac{1}{2^{n/2-k_U}} \qquad (31)$$

Case 3: $\mathbf{x}_1\neq\mathbf{x}_2$ and $\mathbf{x}_1\neq 0$. In this case

$$\mathbb{P}(\mathbf{x}\in(U|U+V)) = \mathbb{P}(\mathbf{H}_U\mathbf{x}_1^T=0 \text{ and } \mathbf{H}_V(\mathbf{x}_1^T+\mathbf{x}_2^T)=0) = \frac{1}{2^{n/2-k_U}}\frac{1}{2^{n/2-k_V}} \qquad (32)$$

Note that we used in each case Lemma 7.

By substituting $\mathbb{P}(\mathbf{x}\in(U|U+V))$ in (29) we obtain for even $0<w\leq n$

$$a_{(U|U+V)}(w) = \frac{\binom{n/2}{w/2}}{2^{n/2-k_U}} + \frac{\binom{n/2}{w}}{2^{n/2-k_V}} + \frac{1}{2^{n-k_U-k_V}}\left(\binom{n}{w} - \binom{n/2}{w} - \binom{n/2}{w/2}\right)$$

and for odd $w\leq n$

$$a(w) = \frac{\binom{n/2}{w}}{2^{n/2-k_V}} + \frac{1}{2^{n-k_U-k_V}}\left(\binom{n}{w} - \binom{n/2}{w}\right)$$