Wave: A New Code-Based Signature Scheme ⋆

Thomas Debris-Alazard[1,2], Nicolas Sendrier[2], and Jean-Pierre Tillich[2]

[1] Sorbonne Universités, UPMC Univ Paris 06
[2] Inria, Paris
{thomas.debris,nicolas.sendrier,jean-pierre.tillich}@inria.fr

Abstract. We present here Wave the first "hash-and-sign" code-based signature scheme which strictly follows the GPV strategy [GPV08]. It uses the family of ternary generalized $(U, U + V)$ codes. We prove that Wave achieves *existential unforgeability under adaptive chosen message attacks* (EUF-CMA) in the random oracle model (ROM) with a tight reduction to two assumptions from coding theory: one is a distinguishing problem that is related to the trapdoor we insert in our scheme, the other one is DOOM, a multiple target version of syndrome decoding. The algorithm produces uniformly distributed signatures through a suitable rejection sampling. Our scheme enjoys efficient signature and verification algorithms. For 128 bits of classical security, signature are 8 thousand bits long and the public key size is slightly smaller than one megabyte. Furthermore, with our current choice of parameters, the rejection rate is limited to one rejection every 3 or 4 signatures.

1 Introduction

Code-Based Signature Schemes. It is a long standing open problem to build an efficient and secure digital signature scheme based on the hardness of decoding a linear code which could compete with widespread schemes like DSA or RSA. Those signature schemes are well known to be broken by quantum computers and code-based schemes could indeed provide a valid quantum resistant replacement. A first answer to this question was given by the CFS scheme proposed in [CFS01]. It consisted in signing with the Niederreiter public-key decryption primitive [Nie86]. This requires a linear code for which there exists an efficient decoding algorithm, able to find the closest codeword for a non-negligible proportion of all words. This means that if $\mathbf{H}$ is an $r \times n$ parity-check matrix of the code, there exists for a non-negligible proportion of all $\mathbf{s}$ in $\mathbb{F}_2^r$ an efficient procedure to decode, that is find a word $\mathbf{e}$ in $\mathbb{F}_2^n$ of smallest Hamming weight such that

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

In such a case we say that $\mathbf{s}$, which is generally called a syndrome in the literature, can be decoded. [CFS01] achieved this task by using high rate Goppa codes. This signature scheme followed a relaxed form of the "hash-and-sign" paradigm. To sign a message $\mathbf{m}$, a hash function h is used to produce a sequence $\mathbf{s}_0, \ldots, \mathbf{s}_\ell$ of elements of $\mathbb{F}_2^r$. For instance $\mathbf{s}_0 = h(\mathbf{m})$ and $\mathbf{s}_i = h(\mathbf{s}_0, i)$ 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 $\mathbf{e}\mathbf{H}^\intercal = \mathbf{s}_i$. This signature scheme has however two drawbacks: (i) for high rates Goppa codes the indistinguishability assumption used in its security proof has been invalidated in [FGO$^+$11], (ii) it scales poorly with respect to security. Indeed, a crude extrapolation of parallel CFS [Fin10] and its implementations [LS12, BCS13] yields for 128 bits of classical security a public key size of several gigabytes and a signature time of several seconds. Those figures even grow to terabytes and hours for quantum-safe security levels, making the scheme unpractical.

Other Code-Based Signature Schemes. Instead of trying to solve a conventional decoding problem, meaning that we want to find an error of minimum weight satisfying (1), it is enough to

⋆ This work was supported by the ANR CBCRYPT project, grant ANR-17-CE39-0007 of the French Agence Nationale de la Recherche.

require in this cryptographic context to solve (1) for an error $\mathbf{e}$ whose weight w is not necessarily minimal but just sufficiently low, so that problem (1) stays hard for someone whose does not know the secret structure of the code that is used. This approach has been followed in [BBC$^+$13] with LDGM codes, in [GSJB14] with (essentially) convolutional codes and in the NIST proposal pqsigRM [LKLN17] with modified Reed-Muller codes. The LDGM scheme was broken in [PT16], [GSJB14] has been broken in [MP16] (and 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]).

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 was also the proposal RaCoSS to the NIST that was based on a public matrix whose columns are formed by syndromes of low weight errors. It was broken in [HBPL18]. Another possibility is to use the Fiat-Shamir heuristic to turn a zero-knowledge authentication scheme into a signature scheme. When based on the Stern code-based authentication scheme [Ste93b] this leads however to a signature scheme with really large signature sizes (of the order of hundred(s) of kilobits). 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] with the RankSign scheme. This scheme enjoys remarkably small key sizes, it is of order tens of thousands bits for 128 bits of security. Unfortunately it got broken in [DT18]. In summary, it is still a very challenging and open question to come up with an efficient and secure signature scheme based on error-correcting codes.

Our Contribution: a "Hash-and-Sign" Signature Scheme Based on the GPV Approach.

Our scheme is based on the hash-and-sign approach and the GPV strategy [GPV08] to devise such signature schemes. Recall that the notions put forward in that paper allowed to build the first identity based encryption scheme based on hard problems on lattices. This strategy has also been adopted in Falcon [FHK$^+$], a lattice based signature submission to the NIST call for post-quantum cryptographic primitives. It is based on the notion of preimage sampleable function. Roughly speaking, this is a family of trapdoor one-way functions $(f_a)_a$ such that with overwhelming probability over the choice of the function f_a (i) the distribution of the images $f_a(x)$ is very close to the uniform distribution over the set of possible outputs (ii) the distribution of the output of the algorithm inverting f_a using the trapdoor is very close to the uniform distribution over the inputs to f_a. In [GPV08] such functions are based on a's that are matrices over $\mathbb{Z}_q^{n\times m}$, whereas the input to f_a is a subset of elements $\mathbf{e} \in \mathbb{Z}^m$ that are of euclidean norm bounded by some quantity W and

$$f_{\mathbf{A}}(\mathbf{e}) = \mathbf{e}\mathbf{A}^\intercal \mod q.$$

Our preimage sampleable function is of the same kind, i.e.

$$f_{\mathbf{H}}(\mathbf{e}) = \mathbf{e}\mathbf{H}^\intercal.$$

with the only difference that we perform matrix multiplication in the finite field $\mathbb{F}_q$ and the inputs $\mathbf{e}$ will be restricted to have Hamming weight exactly w, where w is chosen such that it is hard to solve (1) for $\mathbf{e}$'s of such a Hamming weight.

In [GPV08] a signature scheme based on preimage sampleable functions is given that is shown to be strongly existentially unforgeable under a chosen-message attack if in addition the preimage sampleable functions are also collision resistant. With our choice of w and $\mathbb{F}_q$, our preimage sampleable functions are not collision resistant. However, as observed in [GPV08], collision resistance allows a tight security reduction but is not necessary : a security proof could also be given when the function is "only" preimage sampleable. We will also get a tight security reduction in our case

by choosing carefully the difficult problems we use in the security : one is a distinguishing problem that is related to the trapdoor we insert in our scheme, the other one is a "multiple instances-only one solution required" version of the decoding problem (1). This is the so called "Decoding One Out of Many" problem (DOOM in short) [Sen11].

Our Trapdoor: Generalized $(U, U+V)$ Codes. In [GPV08] the trapdoor consists in a short basis of the lattice considered in the construction. Our trapdoor will be of a different nature, it consists in choosing parity-check matrices of generalized $(U, U+V)$ codes. Not every parity-check matrix is a parity-check matrix of a generalized $(U, U+V)$ code, however there are really plenty of such codes. The U and V codes can namely be chosen at random in this construction and the number of such codes of dimension k and length n is of order $q^{\Theta\left(n^2\right)}$ when $k = \Theta(n)$. A generalized $(U, U+V)$ code of length n over $\mathbb{F}_q$ has 6 ingredients

- Two codes U and V of length $n/2$
- Four $n/2 \times n/2$ diagonal matrices $\mathbf{D}_1, \ldots, \mathbf{D}_4$ over $\mathbb{F}_q$ such that $\mathbf{D} \stackrel{\triangle}{=} \begin{pmatrix} \mathbf{D}_1 & \mathbf{D}_2 \\ \mathbf{D}_3 & \mathbf{D}_4 \end{pmatrix}$ is invertible.

The generalized $(U, U+V)$ code, which we denote by $(U\mathbf{D}_1 + V\mathbf{D}_2, U\mathbf{D}_3 + V\mathbf{D}_4)$ is defined by

$$(U\mathbf{D}_1 + V\mathbf{D}_2, U\mathbf{D}_3 + V\mathbf{D}_4) \stackrel{\triangle}{=} \{(\mathbf{u}\mathbf{D}_1 + \mathbf{v}\mathbf{D}_2, \mathbf{u}\mathbf{D}_3 + \mathbf{v}\mathbf{D}_4) : \mathbf{u} \in U, \mathbf{v} \in V\}.$$

Standard $(U, U+V)$ codes correspond to $\mathbf{D}_1 = \mathbf{D}_3 = \mathbf{D}_4 = \mathbf{1}_{n/2}$ and $\mathbf{D}_2 = \mathbf{0}_{n/2}$, where $\mathbf{1}_{n/2}$ stands for the identity matrix of size $n/2$ and $\mathbf{0}_{n/2}$ is the $n/2 \times n/2$ zero matrix.

It is not the first time that $(U, U+V)$ codes or generalized $(U, U+V)$ codes are suggested for a cryptographic use. $(U, U+V)$ codes were already considered for constructing a McEliece cryptosystem in [KKS05, p.225-228] and generalized $(U, U+V)$ codes in [PMIB17]. However both papers did not consider the improvement in the error correction performance that comes with the $(U, U+V)$-construction (generalized or not) if a decoder that uses soft information is used. This was first observed in the very same cryptographic context in [MCT16]. Having codes with a better error correction in this context results in being able to reduce the key sizes of the scheme. The (generalized) $(U, U+V)$-construction also potentially allows to use in this context codes for U and V that would be insecure in this context if used alone, such as for instance generalized Reed-Solomon codes. This allows for instance to thwart the key attacks [SS96, CGG$^+$14] on the McEliece or Niederreiter scheme based on generalized Reed-Solomon code [Nie86].

We push this idea further here, by allowing U and V to be *completely random* and by decoding them with a very simple decoder, namely a variation of the Prange decoder [Pra62] that is able to produce for *any* parity-check matrix $\mathbf{H}$ at will a solution of (1) when w is in the range $[\![\frac{q-1}{q}r, n - \frac{r}{q}]\!]$. Note that this algorithm works in polynomial time and that outside this range of weights the complexity of the best known algorithms is exponential in n for weights w of the form $w = \omega n$ where ω is a constant that lies outside the interval $[\frac{q-1}{q}\rho, 1 - \frac{\rho}{q}]$ where $\rho \stackrel{\triangle}{=} \frac{r}{n}$. In the case of a parity-check matrix of a generalized $(U, U+V)$ codes, a small tweak in the decoder is able to take advantage of the generalized $(U, U+V)$ structure to obtain smaller or larger weights for w outside this regime. This is in essence the trapdoor of our signature scheme. A further tweak in the decoder consisting in performing only a small amount of rejection sampling (with our choice of parameters one rejection every 3 or 4 signatures) allows to obtain solutions that are uniformly distributed over the words of weight w. Furthermore we also show that syndromes $\mathbf{e}\mathbf{H}^\intercal$ associated to this kind of codes are statistically indistinguishable from random syndromes when errors $\mathbf{e}$ are drawn uniformly at random among the words of weight w. These are the two key properties for obtaining a function $f_{\mathbf{H}}$ that is preimage sampleable in our signature scheme. Finally, a variation of the proof decoding technique of [GPV08] allows to give a tight security proof of our signature scheme that relies only on the hardness of two problems, namely

Decoding Problem: Solving at least one instance of the decoding problem (1) out of multiple instances for a certain relative weight w/n that is outside the range $[\omega^-_{\text{easy}}, \omega^+_{\text{easy}}]$, where $\omega^-_{\text{easy}} = \frac{q-1}{q}\frac{r}{n}$ and $\omega^+_{\text{easy}} = \frac{n-r}{n} + \omega^-_{\text{easy}}$

Distinguishing Problem: Deciding whether a linear code is a permuted generalized $(U, U+V)$ code or not.

Interestingly, some recent work [CD17] has shown that these two properties (namely statistical indistinguishability of the signatures and the syndromes associated to the code family chosen in the scheme) are also enough to obtain a tight security proof in the Quantum Random Oracle Model (QROM) for generic code-based signatures under the assumption that the Decoding Problem remains hard against a quantum computer and that the code family which is used is computationally indistinguishable from generic linear codes. In other words, this can be used to give a tight security proof of our generalized $(U, U+V)$ codes in the QROM.

The Hardness of the Decoding Problem. All code-based cryptography relies upon that problem. The problem of solving (1) for a q-ary $r \times n$ matrix $\mathbf{H}$ is well known to be hard when we seek a word $\mathbf{e}$ of relative weight $w/n < \omega^-_{\text{easy}} = \frac{q-1}{q}\frac{r}{n}$. Beyond this point the problem is easily solved with linear algebra until we reach $\omega^+_{\text{easy}} = \frac{n-r}{n} + \omega^-_{\text{easy}} = 1 - \frac{1}{q}\frac{r}{n}$, and the problem becomes hard again, a fact which is not as widely spread and which we will use in this work.

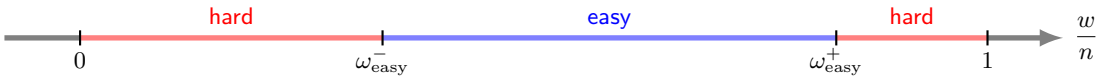


Fig. 1. Asymptotic Hardness of Decoding

Furthermore, here we are in a case where the decoding problem has multiple solutions 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 adapt the known algorithms for decoding a linear code in order to solve this modified problem. This modification was later analysed 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.

The Hardness of the Distinguishing Problem. This problem might seem at first sight to be ad-hoc. However, even in the very restricted case where the generalized $(U, U+V)$-code is just a $(U, U+V)$-code and when the permutation is restricted to leave globally stable the right and left part, detecting whether the resulting code is a permuted $(U, U+V)$-code is NP-complete problem (see [DST17b, §7.1, Thm. 4]). Therefore the Distinguishing Problem is also NP-complete for generalized $(U, U+V)$-code. This theorem is proved in the case of binary $(U, U+V)$-codes in [DST17b, §7.1, Thm 3]). The proof given there carries over directly to an arbitrary finite field $\mathbb{F}_q$. However as observed in [DST17b, p. 3], these NP-completeness reductions hold in the particular case where the dimensions k_U and k_V of the code U and V satisfy $k_U < k_V$. If we stick to the binary case, i.e. $q = 2$, then in order that our $(U, U+V)$ decoder works outside the integer interval $[\![\frac{r}{2}, n - \frac{r}{2}]\!]$ it is necessary that $k_U > k_V$. Unfortunately in this case there is an efficient probabilistic algorithm solving the distinguishing problem that is based on the fact that in this case the hull of the permuted $(U, U+V)$-code is typically of large dimension, namely $k_U - k_V$ (see [DST17a, §1 p.1-2]). This problem can not be settled in the binary case by considering generalized $(U, U+V)$ codes instead of just plain $(U, U+V)$-codes, since it is only for the restricted class of $(U, U+V)$ codes that the $(U, U+V)$ decoder considered in [DST17a] is able to work properly outside the critical interval $[\![\frac{r}{2}, n - \frac{r}{2}]\!]$. This is really related to the polarization phenomenon that lead to the famous construction of polar codes [Arı09]: in the binary case, there is only one 2×2 kernel that polarizes, namely the kernel that corresponds to $(U, U+V)$-codes.

This situation changes drastically when we move to larger finite fields. There are already several different 2×2-kernels that polarize over $\mathbb{F}_3$. In our cryptographic setting, this translates into the fact that it is not only for $(U, U+V)$-codes that we can solve the problem (1) efficiently. This holds for a very large of choices of the $\mathbf{D}_i$'s. The fact that there is a very large choice of matrices of $\mathbf{D}_i$ is clearly a very powerful phenomenon that makes the distinguishing problem much harder. In terms of simplicity of the decoding procedure used in the signing process, it seems that defining our codes over the finite field $\mathbb{F}_3$ is particularly attractive. In such a case, there is a big gain for the generalized $(U, U+V)$ codes that we have chosen and their decoding algorithm to choose the signature weight w to be very large, i.e. significantly above the upper-limit $n - \frac{r}{3}$ below which the decoding problem becomes polynomial as long as w is also above $\frac{2r}{3}$. We will discuss this situation in depth in §7. In this case, it seems that the best approach for solving the distinguishing problem is based on the following observation. This code has namely 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 the Distinguishing Problem 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 Decoding Problem. In some sense, it seems that we might rely on the very same problem, namely solving the Decoding Problem, even if our proof technique does not show this.

All in all, we propose to instantiate our signature scheme in the finite field $\mathbb{F}_3$ which gives the first practical signature scheme based on ternary 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 $O(\lambda^2)$. It should be noted that contrarily to the current thread of research in code-based or lattice-based cryptography which consists in relying on structured codes or lattices based on ring structures in order to decrease the key-sizes we did not follow this approach here. This allows for instance to rely on the NP-complete Decoding Problem which is generally believed to be hard on average rather that on decoding in quasi-cyclic codes for instance whose status is still unclear with a constant number of circulant blocks. Despite the fact that we did not use the standard approach for reducing the key sizes relying on quasi-cyclic codes for instance, we obtain acceptable key sizes (less than one megabyte for 128 bits of security) which compare very favourably to unstructured lattice-based signature schemes such as TESLA for instance [ABB+17]. This is due in part to the tightness of our security reduction.

Organization of the Paper. The paper is organized as follows, we present the outline of our scheme in §3 as well as the properties which are asked to reach the GPV strategy, namely the definition of one-way preimage sampleable functions. In §4 we give the trapdoor that we consider and in §5 we firstly show that it achieves the domain sampling with uniform output of preimage sampleable functions and then we explain how to produce uniformly distributed signatures with some rejection sampling. In §6 we prove it is secure under *existential unforgeability under an adaptive chosen message attack* (EUF-CMA) in the random oracle model (ROM), in relation with this proof we respectively examine in §7 and §8 the best messages 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.

2 Notation

We provide here some notation that will be used throughout the paper.

General Notation. The notation $x \stackrel{\triangle}{=} y$ means that x is defined to be equal to y. We denote by $\mathbb{F}_q$ the finite field with q elements and by $S_{w,n}$ the subset of $\mathbb{F}_q^n$ of words of weight w (q will be

clear from the context). n will also be generally clear from the context. In such a case we just write S_w.

Vector Notation. 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 by $\mathbf{x}_I$ the vector whose coordinates are those of $\mathbf{x} = (x_i)_{1 \leq i \leq n}$ which are indexed by I, i.e.

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

Sometimes we denote for a vector $\mathbf{x}$ by $\mathbf{x}(i)$ its i-th entry, or for a matrix $\mathbf{A}$, by $\mathbf{A}(i,j)$ its entry in row i and column j.

We define the support of $\mathbf{x} = (x_i)_{1 \leq i \leq n}$ 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

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

By extension the support $\mathrm{Supp}(\mathbf{M})$ of a matrix $\mathbf{M}$ is the union of its rows supports.

For a vector $\mathbf{x}$ in $\mathbb{F}_q^n$ and $a \in \mathbb{F}_q$ we denote by $|\mathbf{x}|_a$ its number of entries in it that are equal to a:

$$|\mathbf{x}|_a \stackrel{\triangle}{=} |\{i : x_i = a\}|.$$

Probabilistic Notation. 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 $\mathcal{D}$ we write $\xi \sim \mathcal{D}$ to indicate that the random variable ξ is chosen according to $\mathcal{D}$. The uniform distribution on a certain discrete set is denoted by $\mathcal{U}$. The set will be specified in the text. We denote the uniform distribution on S_w by $\mathcal{U}_w$. When we have probability distributions $\mathcal{D}_1$, $\mathcal{D}_2$, …, $\mathcal{D}_n$ over discrete sets $\mathcal{E}_1$, $\mathcal{E}_2$, …, $\mathcal{E}_n$, we denote by $\mathcal{D}_1 \otimes \mathcal{D}_2 \otimes \cdots \otimes \mathcal{D}_n$ the product probability distribution, i.e $\mathcal{D}_1 \otimes \cdots \otimes \mathcal{D}_n(x_1, \ldots, x_n) \stackrel{\triangle}{=} \mathcal{D}_1(x_1) \ldots \mathcal{D}_n(x_n)$ for $(x_1, \ldots, x_n) \in \mathcal{E}_1 \times \cdots \times \mathcal{E}_n$. The n-th power product of a distribution $\mathcal{D}$ is denoted by $\mathcal{D}^{\otimes n}$, i.e. $\mathcal{D}^{\otimes n} \stackrel{\triangle}{=} \underbrace{\mathcal{D} \otimes \cdots \otimes \mathcal{D}}_{n \text{ times}}$.

The statistical distance between two discrete probability distributions over a same space $\mathcal{E}$ is defined as:

$$\rho(\mathcal{D}_0, \mathcal{D}_1) \stackrel{\triangle}{=} \frac{1}{2} \sum_{x \in \mathcal{E}} |\mathcal{D}_0(x) - \mathcal{D}_1(x)|.$$

For two random variables X and Y ranging over the same space, we will also denote by $\rho(X, Y)$ the statistical distance between the distribution $\mathcal{D}_X$ of X and the distribution $\mathcal{D}_Y$ of Y, that is

$$\rho(X, Y) \stackrel{\triangle}{=} \rho(\mathcal{D}_X, \mathcal{D}_Y).$$

Recall that a function $f(n)$ is said to be negligible if for all polynomials $p(n)$, $|f(n)| < p(n)^{-1}$ for all sufficiently large n that we will denote by $f \in \mathrm{negl}(n)$.

Sometimes when we wish to emphasize on which probability space the probabilities or the expectations are taken, we denote by a subscript the random variable specifying the associated probability space over which the probabilities or expectations are taken. For instance the probability $\mathbb{P}_X(\mathcal{E})$ of the event $\mathcal{E}$ is taken over Ω the probability space over which the random variable X is defined, *i.e.* if X is for instance a real random variable, X is a function from a probability space Ω to $\mathbb{R}$, and the aforementioned probability is taken according to the probability chosen for Ω.

Coding Theory. For any matrix $\mathbf{M}$ we denote $\langle \mathbf{M} \rangle$ the vector space spanned by its rows. A q-ary linear code $\mathcal{C}$ of length n and dimension k is a subspace of $\mathbb{F}_q^n$ of dimension k and is often defined by a *parity-check matrix* $\mathbf{H}$ over $\mathbb{F}_q$ of size $r \times n$ as

$$\mathcal{C} = \langle \mathbf{H} \rangle^{\perp} = \left\{ \mathbf{x} \in \mathbb{F}_q^n : \mathbf{x}\mathbf{H}^{\intercal} = \mathbf{0} \right\}.$$

When $\mathbf{H}$ is of full rank (which is usually the case) we have $r = n - k$. A *generator matrix* of $\mathcal{C}$ is a $k \times n$ full rank matrix $\mathbf{G}$ over $\mathbb{F}_q$ such that $\langle \mathbf{G} \rangle = \mathcal{C}$. The code rate, usually denoted by R, is defined as the ratio k/n.

An *information set* of a code $\mathcal{C}$ of length n is a set of k coordinate indices $\mathcal{I} \subset \{1, \ldots, n\}$ which indexes k independent columns on any generator matrix. Its complement indexes $n - k$ independent columns on any parity check matrix. For any $\mathbf{s} \in \mathbb{F}_q^{n-k}$, $\mathbf{H} \in \mathbb{F}_q^{(n-k)\times n}$, and any information set $\mathcal{I}$ of $\mathcal{C} = \langle \mathbf{H} \rangle^{\perp}$, for all $\mathbf{x} \in \mathbb{F}_q^n$ there exists a unique $\mathbf{e} \in \mathbb{F}_q^n$ such that $\mathbf{e}\mathbf{H}^{\intercal} = \mathbf{s}$ and $\mathbf{x}_{\mathcal{I}} = \mathbf{e}_{\mathcal{I}}$.

We will also consider here the notion of *punctured code*. For a subset $I \subset \{1, \ldots, n\}$ and a code $\mathcal{C}$ of length n, we denote by $\mathrm{Punc}_I(\mathcal{C})$, the code $\mathcal{C}$ punctured in I. This is defined as the set $\{\mathbf{c}_{\bar{I}} = (c_j)_{j \in \{1,\ldots,n\} \setminus I} : \mathbf{c} \in \mathcal{C}\}$, in other words the set of vectors obtained by deleting in the codewords of $\mathcal{C}$ the positions that belong to I.

3 The Wave-Signature Scheme

3.1 Outline of the Scheme

We define a probabilistic full domain hash (FDH) signature scheme, as in [BR96, Cor02]. We replace RSA with a trapdoor function based upon the hardness of the Decoding Problem. Let $\mathcal{C}$ be a linear of dimension k and length n over $\mathbb{F}_q$ defined by a parity-check matrix $\mathbf{H}$ of size $(n-k) \times n$. The one way function $f_{\mathbf{H}}$ we consider is given by

$$\begin{aligned} f_{\mathbf{H}} : S_w &\longrightarrow \mathbb{F}_q^{n-k} \\ \mathbf{e} &\longmapsto \mathbf{e}\mathbf{H}^{\intercal} \end{aligned}$$

Inverting this function on an input $\mathbf{s}$ amounts to solve the Decoding Problem. We are ready now to give the general scheme we consider. Let us assume that we have a family of codes which is defined by a set $\mathcal{F}$ of parity-check matrices of size $(n-k) \times n$ over $\mathbb{F}_q$ such that for all $\mathbf{H}_{\mathrm{sk}} \in \mathcal{F}$ we have an algorithm $D_{\mathbf{H}_{\mathrm{sk}}}$ which on input $\mathbf{s}$ computes $\mathbf{e} \in f_{\mathbf{H}_{\mathrm{sk}}}^{-1}(\mathbf{s})$ (it will be the family of generalized admissible $(U, U+V)$ codes which are defined in §4.2). Then we pick uniformly at random $\mathbf{H}_{\mathrm{sk}} \in \mathcal{F}$, an $n \times n$ permutation matrix $\mathbf{P}$, a non-singular matrix $\mathbf{S} \in \mathbb{F}_q^{(n-k)\times(n-k)}$ which define the secret and public key as:

$$\mathrm{sk} \leftarrow (\mathbf{H}_{\mathrm{sk}}, \mathbf{P}, \mathbf{S}) \ ; \ \mathrm{pk} \leftarrow \mathbf{H}_{\mathrm{pk}} \text{ where } \mathbf{H}_{\mathrm{pk}} \stackrel{\triangle}{=} \mathbf{S}\mathbf{H}_{\mathrm{sk}}\mathbf{P}$$

Remark 1. Let $\mathcal{C}_{\mathrm{sk}}$ be the code defined by $\mathbf{H}_{\mathrm{sk}}$, then $\mathbf{H}_{\mathrm{pk}}$ defines the following code:

$$\mathcal{C}_{\mathrm{pk}} = \{\mathbf{c}\mathbf{P} : \mathbf{c} \in \mathcal{C}_{\mathrm{sk}}\}.$$

We also select a cryptographic hash function $\mathtt{Hash} : \{0,1\}^* \to \mathbb{F}_q^{n-k}$ and a parameter λ_0 for the random salt $\mathbf{r}$. The algorithms $\mathtt{Sgn}^{\mathrm{sk}}$ and $\mathtt{Vrfy}^{\mathrm{pk}}$ are defined as follows

$\mathtt{Sgn}^{\mathrm{sk}}(\mathbf{m})$:	$\mathtt{Vrfy}^{\mathrm{pk}}(\mathbf{m}, (\mathbf{e}', \mathbf{r}))$:
$\mathbf{r} \hookleftarrow \{0,1\}^{\lambda_0}$	$\mathbf{s} \leftarrow \mathtt{Hash}(\mathbf{m}, \mathbf{r})$
$\mathbf{s} \leftarrow \mathtt{Hash}(\mathbf{m}, \mathbf{r})$	$\mathtt{if}\ \mathbf{e}'\mathbf{H}_{\mathrm{pk}}^{\intercal} = \mathbf{s}\ \mathtt{and}\ \lvert\mathbf{e}'\rvert = w\ \mathtt{return}\ 1$
$\mathbf{e} \leftarrow D_{\mathbf{H}_{\mathrm{sk}}}(\mathbf{s}(\mathbf{S}^{-1})^{\intercal})$	$\mathtt{else\ return}\ 0$
$\mathtt{return}(\mathbf{e}\mathbf{P}, \mathbf{r})$	

Remark 2. We add a salt in the scheme in order to have a tight security proof.

Proof (Correction of the verification step). The pair $(\mathbf{eP}, \mathbf{r})$ passes the verification step because by definition of $D_{\mathbf{H}_{\text{sk}}}(\mathbf{s}(\mathbf{S}^{-1})^\intercal)$ we have $\mathbf{eH}_{\text{sk}}^\intercal = \mathbf{s}(\mathbf{S}^{-1})^\intercal$. Therefore $(\mathbf{eP})\mathbf{H}_{\text{pk}}^\intercal = \mathbf{e}(\mathbf{H}_{\text{pk}}\mathbf{P}^\intercal)^\intercal = \mathbf{e}(\mathbf{H}_{\text{pk}}\mathbf{P}^{-1})^\intercal = \mathbf{e}(\mathbf{SH}_{\text{sk}})^\intercal = \mathbf{eH}_{\text{sk}}^\intercal\mathbf{S}^\intercal = \mathbf{s}(\mathbf{S}^{-1})^\intercal\mathbf{S}^\intercal = \mathbf{s}$. We also have $|\mathbf{eP}| = |\mathbf{e}| = w$.

To summarize, a valid signature of a message $\mathbf{m}$ consists of a pair $(\mathbf{e}, \mathbf{r})$ such that $\mathbf{eH}_{\text{pk}}^\intercal = \texttt{Hash}(\mathbf{m}, \mathbf{r})$ with $\mathbf{e}$ of Hamming weight w.

3.2 One-way Preimage Sampleable Code-based Functions

Classically, FDH signature schemes such as DSA are based on a trapdoor one-way function $f : \mathcal{D} \to \mathcal{A}$ which is a permutation. In this case the trapdoor permits to compute for any $\mathbf{a} \in \mathcal{A}$ the unique $\mathbf{d} \in \mathcal{D}$ (the signature of $\mathbf{a}$) which verifies $f(\mathbf{d}) = \mathbf{a}$. Therefore, in the random oracle model when $\mathbf{a}$ follows the uniform distribution, signatures $\mathbf{d}$ which are produced are uniform too. In this way, the nice property to be a permutation for the trapdoor one-way function offers a first level of security for the signature scheme. Nevertheless, for the trapdoor one way function $f_{\mathbf{H}}$ which is used in our code-based scheme, this condition to be a permutation is too strong to be met in an interesting way. This situation does not only arise for code-based trapdoor FDH signatures, it appears in lattice-based cryptography too. In this context, authors of [GPV08] gave additional properties to be verified by the one way function. One of the crucial property that is asked for is that the algorithm that inverts $f_{\mathbf{H}}$ based on the trapdoor is close to the uniform distribution on the inputs. In the following we will speak of GPV strategy. Authors of [GPV08] summarized this in the definition of preimage sampleable function (see [GPV08, Definition 5.3.1]). To simplify the definition, we will directly express this notion for a restricted (and simplified) class of code-based trapdoor function.

Definition 1 (One-way preimage sampleable code-based functions). *It is a pair of probabilistic polynomial-time algorithm* $(\texttt{Trapdoor}, \texttt{InvertAlg})$ *together with a triple of functions* $(n(\lambda), k(\lambda), w(\lambda))$ *growing polynomially with the security parameter* λ *and giving the length and dimension of the codes and the signature weight we consider, such that*

- `Trapdoor` *when given* λ*, outputs* $(\mathbf{H}, T)$ *where* $\mathbf{H}$ *is an* $(n-k) \times n$ *matrix over* $\mathbb{F}_q$ *and* T *the trapdoor corresponding to* $\mathbf{H}$*. Here and elsewhere we drop the dependence in* λ *of the functions* n, k *and* w*.*
- `InvertAlg` *is a probabilistic algorithm which takes as input* T *and an element* $\mathbf{s} \in \mathbb{F}_q^{n-k}$ *and outputs an* $\mathbf{e} \in S_{w,n}$ *such that* $\mathbf{eH}^\intercal = \mathbf{s}$*.*

The following properties have to hold for all but a negligible fraction of $\mathbf{H}$ *output by* `Trapdoor`*.*

1. Domain Sampling with uniform output:

$$\rho(\mathbf{eH}^\intercal, \mathbf{s}) \in \text{negl}(\lambda)$$

   where $\mathbf{e}$ *and* $\mathbf{s}$ *are two random variables, with* $\mathbf{e}$ *being uniformly distributed over* $S_{w,n}$ *and* $\mathbf{s}$ *being uniformly distributed over* $\mathbb{F}_q^{n-k}$*.*
2. Preimage Sampling with trapdoor: *for every* $\mathbf{s} \in \mathbb{F}_q^{n-k}$*, we have*

$$\rho\left(\texttt{InvertAlg}(\mathbf{s}, T), \mathbf{e}\right) \in \text{negl}(\lambda),$$

   where $\mathbf{e}$ *is uniformly distributed in* $S_{w,n}$*.*
3. One wayness without trapdoor: *for any probabilistic poly-time algorithm* $\mathcal{A}$ *outputting an element* $\mathbf{e} \in S_{w,n}$ *when given* $\mathbf{H} \in \mathbb{F}_q^{(n-k)\times n}$ *and* $\mathbf{s} \in \mathbb{F}_q^{n-k}$*, the probability that* $\mathbf{eH}^\intercal = \mathbf{s}$ *is negligible, where the probability is taken over the choice of* $\mathbf{H}$*, the target value* $\mathbf{s}$ *chosen uniformly at random, and* $\mathcal{A}$*'s random coins.*

It will turn out that by choosing the parameters appropriately and by choosing $\mathbf{H}$ such that it is a parity-check matrix of a permuted generalized $(U, U+V)$ code, we will be able to solve $\mathbf{eH}^\intercal = \mathbf{s}$ by using the underlying generalized $(U, U+V)$ structure in a regime of parameters where solving this problem for generic linear codes is thought to be hard. Moreover, by a suitable rejection technique we will show how the decoding algorithm using the trapdoor can be made oblivious of the underlying trapdoor. This is the preimage sampling condition of the previous definition.This mimics in a sense what has been achieved in the lattice setting of [GPV08] where the inversion algorithm is oblivious to the particular geometry of the trapdoor basis. Similarly to what has been achieved in [GPV08], we will also show that our construction based on permuted generalized $(U, U+V)$ codes also verifies the domain sampling condition.

Under the assumption that the Distinguishing and Decoding Problems are hard, we could have shown that the coding theoretic function $f_{\mathbf{H}}$ that we consider here is one way. This would show that our code-based construction is a preimage sampleable function. However, the proof technique for showing the security of the signature scheme based on a preimage sampleable function given in [GPV08] relies on a stronger version of a preimage sampleable function, it should namely also be collision resistant. Our code-based construction will not meet this condition for the particular choice we will make for our scheme. This comes from the fact that we will focus on a ternary alphabet $q = 3$ and very large values of w. We will proceed in a slightly different way in our case. We namely give a security reduction relying on the assumptions that the Distinguishing and Decoding Problems are hard and on the preimage sampling property on one hand and the domain sampling property on the other hand. The preimage sampling condition of the previous definition is here to ensure that signatures which are produced do not leak any information whereas the domain sampling condition may seem more surprising. As we will see, it naturally appears in the security reduction as it enables to inject a hard instance of the Decoding Problem to which we reduce.

4 Inverting the Syndrome Function

This section is devoted to the inversion of $f_{\mathbf{H}}$. It amounts to solve the following problem

Problem 1 (Syndrome Decoding with fixed weight). Given $\mathbf{H} \in \mathbb{F}_q^{(n-k)\times n}$, $\mathbf{s} \in \mathbb{F}_q^{n-k}$, and an integer w, find $\mathbf{e} \in \mathbb{F}_q^n$ such that $\mathbf{eH}^\intercal = \mathbf{s}$ and $|\mathbf{e}| = w$.

Here

- we recall for which interval $[\![w^-, w^+]\!]$ of values for w we may hope to invert $f_{\mathbf{H}}$ on any possible output;
- we recall in which interval of values $[\![w^-_{\text{easy}}, w^+_{\text{easy}}]\!] \subset [\![w^-, w^+]\!]$ it is easy to invert $f_{\mathbf{H}}$ for any parity-check matrix $\mathbf{H}$ without using any trapdoor: this is the well-known Prange decoder;
- we then explain how in the particular case of a generalized $(U, U+V)$ code we can invert $f_{\mathbf{H}}$ by tweaking the Prange decoder for a significantly larger range $[\![w^-_{\text{UV}}, w^+_{\text{UV}}]\!]$ of w than $[\![w^-_{\text{easy}}, w^+_{\text{easy}}]\!]$. This is the key that shows how to exploit the underlying $(U, U+V)$ structure as a trapdoor for inverting $f_{\mathbf{H}}$.

Any solver of Problem 1 will be called decoding algorithm, whatever is the weight w. For small weights there is at most one solution and it relates to error-correction. For larger weight we may have exponentially many solutions and the problem relates to source-distortion theory in which it is also referred to as decoding.

4.1 Generic Solutions

Surjective Domain of the Syndrome Function The issue is here for which value of w we may expect that $f_{\mathbf{H}}$ is surjective. This clearly implies that $|S_w| \geq q^{n-k}$. In other words we have the following simple fact.

Fact 1 *If $f_{\mathbf{H}}$ is surjective, then necessarily $w \in [\![w^-, w^+]\!]$ with*

$$w^- \triangleq \min\left\{w \in [\![0,n]\!], \quad \binom{n}{w}(q-1)^w \geq q^{n-k}\right\}$$
$$w^+ \triangleq \max\left\{w \in [\![0,n]\!], \quad \binom{n}{w}(q-1)^w \geq q^{n-k}\right\}.$$

For a fixed rate $R = k/n$, it is readily verified that the asymptotic (in n) behaviour of w^- and w^+ is given by

$$\frac{w^-}{n} = g_q^-(1-R) + o(1) \tag{2}$$
$$\frac{w^+}{n} = g_q^+(1-R) + o(1) \ \text{ if } R \leq \log_q\left(\frac{q}{q-1}\right) \tag{3}$$
$$\frac{w^+}{n} = 1 \ \text{ otherwise.} \tag{4}$$

where $h_q(x) \triangleq -(1-x)\log_q(1-x) - x\log_q\left(\frac{x}{q-1}\right)$ and g_q^- its inverse ranging over $[0,(q-1)/q]$, whereas g_q^+ is its inverse ranging over $[(q-1)/q, 1]$, but whose domain is restricted to $[\log_q(q-1), 1]$. This motivates the definition of ω^- and ω^+ as

$$\omega^- \triangleq g_q^-(1-R) \tag{5}$$
$$\omega^+ \triangleq g_q^+(1-R) \ \text{ if } R \leq \log_q\left(\frac{q}{q-1}\right) \tag{6}$$
$$\omega^+ = 1 \ \text{ otherwise.} \tag{7}$$

The Gilbert-Varshamov distance corresponds to the smallest radius of a ball in $\mathbb{F}_q^n$ centred around 0 whose volume is above q^{n-k}. Since the volume of a ball is well approximated by the area of the sphere, w^- is actually very close (if not equal) to the Gilbert-Varshamov distance and ω^- is precisely the asymptotic relative Gilbert-Varshamov distance. A straightforward computation of the expected number of errors $\mathbf{e}$ of weight w such that $\mathbf{eH}^\intercal = \mathbf{s}$ when $\mathbf{H}$ is random shows that we expect an exponential number of solutions when w/n lies in (ω^-, ω^+):

Proposition 1. *Let n, k, w be integers with $k \leq n$ and $\mathbf{s} \in \mathbb{F}_q^{n-k}$. The expected number of solutions of $\mathbf{eH}^\intercal = \mathbf{s}$ in $\mathbf{e}$ of weight w when $\mathbf{H}$ is chosen uniformly at random in $\mathbb{F}_q^{(n-k)\times n}$ is given by:*

$$\frac{\binom{n}{w}(q-1)^w}{q^{n-k}}.$$

When n tends to infinity but w and k are such that $w = \omega n$ and $k = Rn$ with ω fixed in (ω^-, ω^+) and R fixed in $(0,1)$, then this expected number of solutions behaves like $e^{\alpha n(1+o(1))}$ for a certain $\alpha > 0$.

However, even the exponential number of solutions to our problem, coding theory has never come up with an efficient algorithm for finding a solution to this problem in the whole range (ω^-, ω^+). An efficient solution is only known for a subrange by using the Prange decoder.

Easy Domain of the Syndrome Function The subrange of (ω^-, ω^+) for which we know how to solve efficiently Problem 1 is given by the condition $w/n \in [\omega^-_{\text{easy}}, \omega^+_{\text{easy}}]$ where

$$\omega^-_{\text{easy}} \triangleq \frac{q-1}{q}(1-R) \tag{8}$$
$$\omega^+_{\text{easy}} \triangleq \frac{q-1}{q}(1-R) + R, \tag{9}$$

where $R \triangleq \frac{k}{n}$. This is achieved by a sightly generalized version of the Prange decoder [Pra62]. To explain how it works, consider a linear code $\mathcal{C}$ over $\mathbb{F}_q$ of length n and dimension k defined by a parity-check matrix $\mathbf{H}$. We want to find for a given $\mathbf{s}$ and error $\mathbf{e}$ of weight w such that $\mathbf{eH}^\intercal = \mathbf{s}$. Roughly speaking, the subspace structure of $\mathcal{C}$ offers k bits of $\mathbf{e}$ that can be arbitrarily chosen and the other $n-k$ bits are uniquely determined. Indeed, $\mathbf{H}$ 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 $\mathbf{e} = (\mathbf{e}'', \mathbf{e}')$ where $\mathbf{e}' \in \mathbb{F}_q^k$ and $\mathbf{e}'' \in \mathbb{F}_q^{n-k}$. We should therefore have $\mathbf{s} = \mathbf{eH}^\intercal = \mathbf{e}''\mathbf{A}^\intercal + \mathbf{e}'\mathbf{B}^\intercal$, that is $\mathbf{e}'' = (\mathbf{s} - \mathbf{e}'\mathbf{B}^\intercal)(\mathbf{A}^{-1})^\intercal$. In this way we can arbitrarily choose the error $\mathbf{e}'$ of length k. Therefore, if we look for an error of low weight, we can set these k positions to 0. For the remaining part we expect to get about $\frac{q-1}{q}(n-k)$ positions that are non zero. On the other hand, to get an error of largest possible weight, the best strategy seems to set the k positions to non-zero values. We also get in this case in the remaining part about $\frac{q-1}{q}(n-k)$ positions that are non zero. The weights that are easily attainable by this strategy are therefore $\frac{q-1}{q}(n-k) = n\omega^-_{\text{easy}}$ and $k + \frac{q-1}{q}(n-k) = n\omega^+_{\text{easy}}$. We can get all intermediate weights in this interval by choosing the appropriate number of zeros in the k positions of $\mathbf{e}'$. For reasons that will appear when we consider a decoder for generalized $(U, U+V)$-codes it will be convenient to choose the weight of $\mathbf{e}'$ to be a random variable. In other words, the generalized Prange decoder looks as given in Algorithm 1.

Algorithm 1 PrangeOne$(\mathbf{H}, \mathbf{s})$ — One iteration of the Prange decoder

Parameters: q, n, k, $\mathcal{D}$ a distribution over $[\![0, k]\!]$

Require: $\mathbf{H} \in \mathbb{F}_q^{(n-k)\times n}$, $\mathbf{s} \in \mathbb{F}_q^{n-k}$
Ensure: $\mathbf{eH}^\intercal = \mathbf{s}$
1: $t \hookleftarrow \mathcal{D}$
2: $\mathcal{I} \leftarrow$ InfoSet$(\mathbf{H})$
3: $\mathbf{x} \hookleftarrow \{\mathbf{x} \in \mathbb{F}_q^n \mid |\mathbf{x}_\mathcal{I}| = t\}$
4: $\mathbf{e} \leftarrow$ PrangeStep$(\mathbf{H}, \mathbf{s}, \mathcal{I}, \mathbf{x})$
5: **return** $\mathbf{e}$

function InfoSet$(\mathbf{H})$ — information set

Require: $\mathbf{H} \in \mathbb{F}_q^{(n-k)\times n}$
Ensure: the returned value $\mathcal{I}$ is an information set of $\langle\mathbf{H}\rangle^\perp$

An information set of a k-dimensional code is a set of k coordinate indices such that, on any $k\times n$ generator matrix, it indexes a non singular $k\times k$ submatrix. Its complement indexes a non singular $(n-k)\times(n-k)$ submatrix on any $(n-k)\times n$ parity check matrix.

function PrangeStep$(\mathbf{H}, \mathbf{s}, \mathcal{I}, \mathbf{x})$ — Prange vector completion

Require: $\mathbf{H} \in \mathbb{F}_q^{(n-k)\times n}$, $\mathbf{s} \in \mathbb{F}_q^{n-k}$, $\mathcal{I}$ an information set of $\langle\mathbf{H}\rangle^\perp$, $\mathbf{x} \in \mathbb{F}_q^n$
Ensure: $\mathbf{eH}^\intercal = \mathbf{s}$ and $\mathbf{e}_\mathcal{I} = \mathbf{x}_\mathcal{I}$

$\mathbf{P} \leftarrow$ any $n\times n$ permutation matrix sending $\mathcal{I}$ on the last k coordinates
$(\mathbf{A} \mid \mathbf{B}) \leftarrow \mathbf{HP}$ ▷ $\mathbf{A} \in \mathbb{F}_q^{(n-k)\times(n-k)}$
$(\mathbf{0} \mid \mathbf{e}') \leftarrow \mathbf{x}$ ▷ $\mathbf{e}' \in \mathbb{F}_q^k$
$\mathbf{e} \leftarrow \Big((\mathbf{s} - \mathbf{e}'\mathbf{B}^\intercal)\,(\mathbf{A}^{-1})^\intercal, \mathbf{e}'\Big)\,\mathbf{P}^\intercal$
return $\mathbf{e}$

This algorithm represents one step of the Prange decoder and it is called as many times are needed to produce an error $\mathbf{e}$ of weight w. The probability distribution of the weights of the $\mathbf{e}$'s output by this function is readily seen to be given by

Proposition 2. *When $\mathbf{H}$ is chosen uniformly at random in $\mathbb{F}_q^{(n-k)\times n}$ and $\mathbf{s}$ uniformly at random in $\mathbb{F}_q^{n-k}$, we can write the weight of the $\mathbf{e}$'s output by* PrangeOne$(\mathbf{H}, \mathbf{s})$ *as*

$$|\mathbf{e}| = S + T$$

where S and T are independent random variables, $S \in [\![0, n-k]\!]$, $T \in [\![0, k]\!]$, S is the Hamming weight of a vector that is uniformly distributed over $\mathbb{F}_q^{n-k}$ and $\mathbb{P}(T = t) = \mathcal{D}(t)$. The distribution of $|\mathbf{e}|$ is given by

$$\mathbb{P}\left(|\mathbf{e}| = w\right) = \sum_{t=0}^{w} \frac{\binom{n-k}{w-t}(q-1)^{w-t}}{q^{n-k}} \mathcal{D}(t) \tag{10}$$

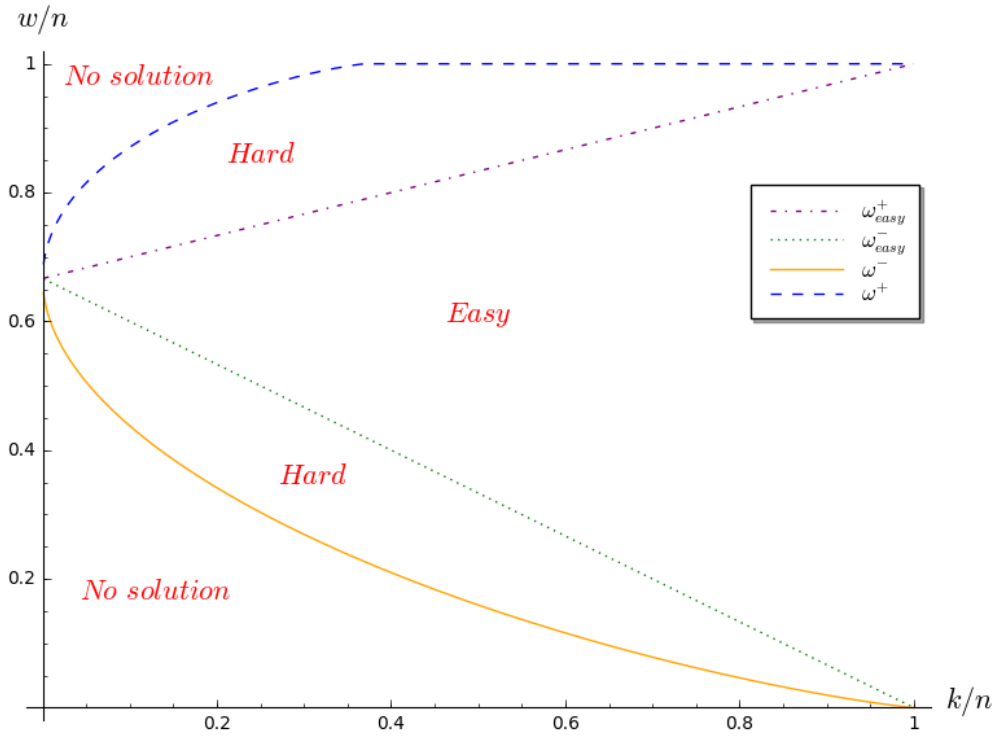
$$\mathbb{E}(|\mathbf{e}|) = \overline{\mathcal{D}} + \frac{q-1}{q}(n-k) = \overline{\mathcal{D}} + n\omega_{easy}^{-}, \tag{11}$$

where $\overline{\mathcal{D}} = \sum_{t=0}^{k} t\mathcal{D}(t)$.

From this proposition, we deduce immediately that any weight w in $[\![\omega_{\text{easy}}^{-} n, \omega_{\text{easy}}^{+} n]\!]$ can be reached by this Prange decoder with a probabilistic polynomial time algorithm that uses a distribution $\mathcal{D}$ such that $\overline{\mathcal{D}} = w - \omega_{\text{easy}}^{-} n$. It will be helpful in what follows to be able to choose a probability distribution $\mathcal{D}$ as this gives a rather large degree of freedom in the distribution of $|\mathbf{e}|$ that will come come very handy to simulate an output distribution that is uniform over the words of weight w in the generalized $(U, U+V)$ code decoder that we will consider in what follows.

To summarize this discussion we have shown that when we want to build a code-based signature scheme, w has to verify $w^- \leq w \leq w^+$ to ensure that $f_{\mathbf{H}}$ is surjective but with an expected exponential number of solutions for a given syndrome (see Proposition 1). However, in a cryptographic setting w/n cannot lie in $[\omega_{\text{easy}}^{-}, \omega_{\text{easy}}^{+}] \subseteq [\omega^-, \omega^+]$ otherwise anybody that uses the generalized Prange algorithm would be able to invert $f_{\mathbf{H}}$. All of this is summarized in Figure 2 where we draw the above different areas asymptotically in n of w/n when k/n is fixed.

Fig. 2. Areas of relative signature distances.



Enlarging the Easy Domain $[\![\boldsymbol{w}_{\text{easy}}^{-}, \boldsymbol{w}_{\text{easy}}^{+}]\!]$ Inverting the syndrome function $f_{\mathbf{H}}$ is the basic problem upon which all code-based cryptography relies. This problem has been studied for a long

time for weights $w \leq w^-_{\text{easy}}$ and despite many efforts the best algorithms [Ste88, Dum91, Bar97, MMT11, BJMM12, MO15, DT17] for solving this problem are all exponential. In other words, after a thorough fifty years of research, none of those algorithms came up with a polynomial complexity for weights $w < w^-_{\text{easy}}$. Furthermore, by adapting all the previous algorithms beyond this point we observe for them the same behaviour: they are all polynomial in the range of weights $[\![w^-_{\text{easy}}, w^+_{\text{easy}}]\!]$ and become exponential once again when $w > w^+_{\text{easy}}$. Therefore, it seems to be an hard problem to enlarge the range where inverting $f_{\mathbf{H}}$ is easy. In the following subsection we present a trapdoor on the matrices $\mathbf{H}$ which enables to invert in polynomial time $f_{\mathbf{H}}$ by tweaking the Prange decoder on a larger range than $[\![w^-_{\text{easy}}, w^+_{\text{easy}}]\!]$.

4.2 Solution with Trapdoor

Let us introduce the family of codes (this is the trapdoor) that we consider to invert $f_{\mathbf{H}}$. As we will see in what follows, this family comes with a simple algorithm which enables to invert $f_{\mathbf{H}}$ with errors of weight which belongs to $[\![w^-_{\text{UV}}, w^+_{\text{UV}}]\!] \subset [\![w^-, w^+]\!]$ but with $[\![w^-_{\text{easy}}, w^+_{\text{easy}}]\!] \subsetneq [\![w^-_{\text{UV}}, w^+_{\text{UV}}]\!]$. We summarize this situation in Figure 3.

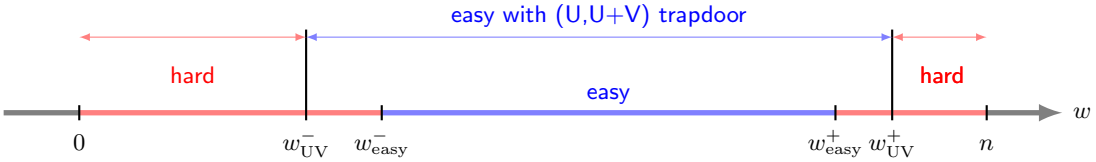


Fig. 3. Hardness of (U,U+V) Decoding

Definition 2 (Generalized admissible $(U, U+V)$-codes). *Let n be an integer and four diagonal matrices $\mathbf{D}_1, \cdots, \mathbf{D}_4$ over $\mathbb{F}_q$ such that:*

$$\mathbf{D} = \begin{pmatrix} \mathbf{D}_1 & \mathbf{D}_2 \\ \mathbf{D}_3 & \mathbf{D}_4 \end{pmatrix} \text{ is invertible and } \forall i \in [\![1, n/2]\!], \quad \mathbf{D}_1(i,i)\mathbf{D}_3(i,i) \neq 0 \tag{12}$$

Let U, V be linear q-ary codes of length $n/2$ and dimension k_U, k_V. We define the subset of $\mathbb{F}_q^n$:

$$(U\mathbf{D}_1 + V\mathbf{D}2, U\mathbf{D}_3 + V\mathbf{D}_4) \stackrel{\triangle}{=} \{(\mathbf{u}\mathbf{D}_1 + \mathbf{v}\mathbf{D}_2, \mathbf{u}\mathbf{D}_3 + \mathbf{v}\mathbf{D}_4) \text{ such that } \mathbf{u} \in U \text{ and } \mathbf{v} \in V\}$$

which is a linear code of length n and dimension $k = k_U + k_V$. A parity-check matrix of such a code is given by

$$\begin{pmatrix} \mathbf{H}_U\mathbf{D}_4\mathbf{M} & -\mathbf{H}_U\mathbf{D}_2\mathbf{M} \\ \mathbf{H}_V\mathbf{D}_3\mathbf{M} & -\mathbf{H}_V\mathbf{D}_1\mathbf{M} \end{pmatrix}$$

where

$$\mathbf{M} \stackrel{\triangle}{=} (\mathbf{D}_1\mathbf{D}_4 - \mathbf{D}_3\mathbf{D}_2)^{-1},$$

$\mathbf{H}_U \in \mathbb{F}_q^{(n/2-k_U)\times n/2}$ *(resp.* $\mathbf{H}_V \in \mathbb{F}_q^{(n/2-k_V)\times n/2}$*) is a parity-check matrix of U (resp. V).*

For a sake of simplicity in the description of the algorithm to invert $f_{\mathbf{H}}$ when $\mathbf{H}$ is a parity-check matrix of a generalized admissible $(U, U+V)$ code, we will restrict our study to the case of $\mathbf{D}_1 = \mathbf{D}_3 = \mathbf{D}_4 = \mathbf{1}_{n/2}$ and $\mathbf{D}_2 = \mathbf{0}_{n/2}$ which corresponds to standard $(U, U+V)$-codes. However, all our discussion (especially this subsection and the following) can be generalized.

It turns out now that in the case of a $(U, U+V)$ code, a simple tweak of the Prange decoder will be able to reach relative weights w/n outside the "easy" region $[\omega^-_{\text{easy}}, \omega^+_{\text{easy}}]$. Let us first explain how the idea works in the case of a $(U, U+V)$-code. It exploits the fundamental leverage of the Prange decoder : it consists in choosing the error $\mathbf{e}$ satisfying $\mathbf{e}\mathbf{H}^\intercal = \mathbf{s}$ as we want in k positions

when the code that we decode is of dimension k. When we want an error of low weight, we put zeroes on those positions, whereas if we want an error of large weight, we put non-zero values. This idea can be adapted in the case of a $(U, U+V)$-code. The parity-check matrix of such a code can be chosen to be

$$\mathbf{H} = \begin{pmatrix} \mathbf{H}_U & \mathbf{0} \\ -\mathbf{H}_V & \mathbf{H}_V \end{pmatrix} \tag{13}$$

Solving $\mathbf{e}\mathbf{H}^\intercal = \mathbf{s}$ with $\mathbf{H}$ as (13) amounts to solve

$$\mathbf{e}_U \mathbf{H}_U^\intercal = \mathbf{s}_U \tag{14}$$
$$\mathbf{e}_V \mathbf{H}_V^\intercal = \mathbf{s}_V \tag{15}$$

where we split $\mathbf{s} = (\mathbf{s}_U, \mathbf{s}_V)$ and we have $\mathbf{e} = (\mathbf{e}_U, \mathbf{e}_U + \mathbf{e}_V)$. Performing the two decoding (14) and (15) independently with Prange algorithm gains nothing. However if we first solve (15) with Prange algorithm, and then seek a solution of (15) which properly depends of $\mathbf{e}_V$ we increase the easy range of weights accessible for $\mathbf{e}$. It then turns out that the range $[\omega_{\mathrm{UV}}^-, \omega_{\mathrm{UV}}^+]$ of relative weights w/n for which the $(U, U+V)$ decoder is easy is larger than the generic easy domain $[\omega_{\mathrm{easy}}^-, \omega_{\mathrm{easy}}^+]$, see Fig. 3. This will provides an advantage to the trapdoor owner.

Tweaking the Prange decoder For Reaching Low Weights. In this case, we first look, with the help of the Prange decoder, for the $\mathbf{e}_V$ of lowest possible weight satisfying (15). We can attain a weight for $\mathbf{e}_V$ of $\frac{q-1}{q}(n/2 - k_V)$. In a second step we have to find $\mathbf{e}_U$ satisfying (14). The point is now that we will not look for $\mathbf{e}_U$ of lowest possible weight satisfying (14), we will instead use the knowledge of $\mathbf{e}_V$ to do better. To understand this point, what we want to do now is to minimize the weight of $\mathbf{e} = (\mathbf{e}_U, \mathbf{e}_U + \mathbf{e}_V)$ given $\mathbf{e}_V$ and $\mathbf{e}_U \mathbf{H}_U^\intercal = \mathbf{s}_U$. The strategy of Prange is to choose the value of $\mathbf{e}_U$ in k_U positions (the "information set") and to complete the rest of the values with the help of the equation $\mathbf{e}_U \mathbf{H}_U^\intercal = \mathbf{s}_U$. For this, we consider the set of positions i for which $\mathbf{e}_V(i) = 0$. Without loss of generality, we may assume that these are the first $n/2 - \frac{q-1}{q}(n/2 - k_V) = \frac{n}{2q} + \frac{q-1}{q}k_V$ positions of $\mathbf{e}_V$ and that $\mathbf{e}_U$ and $\mathbf{e}_V$ split as

$$\mathbf{e}_U = (\mathbf{e}'_U, \mathbf{e}''_U)$$
$$\mathbf{e}_V = (\mathbf{0}, \mathbf{e}''_V)$$

where $\mathbf{e}'_U$ is in $\mathbb{F}_q^{\frac{n}{2q} + \frac{q-1}{q}k_V}$ and $\mathbf{e}''_U$, $\mathbf{e}''_V$ are in $\mathbb{F}_q^{\frac{q-1}{q}(n/2 - k_V)}$. The error $\mathbf{e}$ we are looking for is therefore of the form

$$\mathbf{e} = (\mathbf{e}'_U, \mathbf{e}''_U, \mathbf{e}'_U, \mathbf{e}''_U + \mathbf{e}''_V).$$

This is also represented in Figure 4.

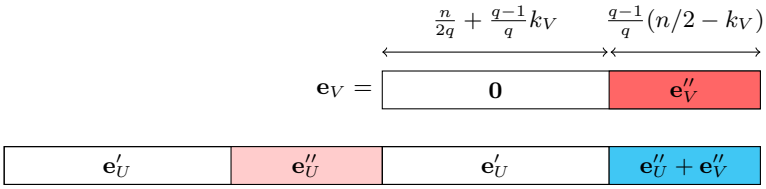


Fig. 4. The form of the errors $\mathbf{e}_V$ and $\mathbf{e}$

To get an error of smallest weight for $\mathbf{e}$ it really makes sense to choose as many zeros as we can in $\mathbf{e}'_U$: they are doubled in $\mathbf{e}$. Choosing a position of $\mathbf{e}''_U$ to be 0 is less helpful, since the corresponding position in $\mathbf{e}''_U + \mathbf{e}''_V$ is non-zero in this case. There are two cases to consider:
Case 1 : $k_U \leq \frac{n}{2q} + \frac{q-1}{q}k_V$. Here we choose an information set for the Prange decoder among

the positions in $\mathbf{e}'_U$ and ask in the Prange step applied to $\mathbf{H}_U$ and $\mathbf{s}_U$ that $\mathbf{e}_U$ is zero on these positions. Here, the expected weight of $\mathbf{e}$ is given by

$$\begin{aligned}\mathbb{E}(|\mathbf{e}|) &= 2\frac{q-1}{q}\left(\frac{n}{2q}+\frac{q-1}{q}k_V - k_U\right) + 2\frac{q-1}{q}\left(\frac{q-1}{q}(n/2-k_V)\right)\\ &= \frac{q-1}{q}n - 2\frac{q-1}{q}k_U \end{aligned} \tag{16}$$

Case 2 : $k_U > \frac{n}{2q} + \frac{q-1}{q}k_V$. In the Prange step applied to $\mathbf{H}_U$ and $\mathbf{s}_U$, we choose the information set for U, so that it contains all the $\frac{n}{2q} + \frac{q-1}{q}k_V$ first positions (i.e. those of $\mathbf{e}'_U$) and choose $\mathbf{e}_U$ to be zero on these positions. In this case, we have

$$\begin{aligned}\mathbb{E}(|\mathbf{e}|) &= k_U - \frac{n}{2q} - \frac{q-1}{q}k_V + 2\frac{q-1}{q}(n/2-k_U)\\ &= \left(1-\frac{3}{2q}\right)n - \frac{q-2}{q}k_U - \frac{q-1}{q}k_V \end{aligned} \tag{17}$$

It is readily seen that if we want to minimize the expected weight of $|\mathbf{e}|$ for a given dimension $k = k_U + k_V$ of the $(U, U+V)$-code, it is always better to use the first strategy. From this it can be verified that as long $k \leq \frac{n}{2q}$ the best we can do is to choose $k_U = k$ and $k_V = 0$. This leads in this regime to an expected weight given by (16) with $k_U = k$:

$$\mathbb{E}(|\mathbf{e}|) = \frac{q-1}{q}n - 2\frac{q-1}{q}k. \tag{18}$$

For larger values of k, it can be verified that the best we can do is to enforce $k_U = \frac{n}{2q} + \frac{q-1}{q}k_V$, which together with the relation $k = k_U + k_V$ implies that

$$k_U = \frac{n}{2(2q-1)} + \frac{(q-1)k}{2q-1}. \tag{19}$$

Plugging this expression into (16) leads to

$$\mathbb{E}(|\mathbf{e}|) = \frac{2(q-1)^2}{(2q-1)q}(n-k). \tag{20}$$

Tweaking the Prange Decoder for Reaching Large Weights. When $q = 2$, small and large weights play a symmetrical role. This is not the case anymore for $q \geq 3$. Indeed, in this case the best Prange strategy does not take into account the weight of $\mathbf{e}_V$. Assume here that we have chosen *any* $\mathbf{e}$ satisfying $\mathbf{e}_V\mathbf{H}^\intercal = \mathbf{s}_V$. How does the second decoding for $\mathbf{e}_U$ take into account this? We want here to find $\mathbf{e}_U$ that maximizes the weight of $(\mathbf{e}_U, \mathbf{e}_U + \mathbf{e}_V)$ given that $\mathbf{e}_U\mathbf{H}^\intercal = \mathbf{s}_U$. Recall that the Prange strategy consists in choosing the value of $\mathbf{e}_U$ in k_U positions and to complete the rest of the values with the help of the equation $\mathbf{e}_U\mathbf{H}^\intercal = \mathbf{s}_U$. Here for any position i, it is always possible to choose $\mathbf{e}_U(i)$ such that both $\mathbf{e}_U(i)$ and $\mathbf{e}_U(i) + \mathbf{e}_V(i)$ are non-zero. By using this strategy, we see that the expected weight of $\mathbf{e}$ becomes fro $q \geq 3$

$$\begin{aligned}\mathbb{E}(|\mathbf{e}|) &= 2k_U + (n-2k_U)\frac{q-1}{q}\\ &= \frac{q-1}{q}n + \frac{2k_U}{q} \end{aligned} \tag{21}$$

The best choice for k_U is to take $k_U = k$ up to the point where $\frac{q-1}{q}n + \frac{2k}{q} = n$, that is $k = n/2$. for larger values of k we choose $k_U = n/2$ and $k_V = k - k_U$.

Trapdoor Pseudocode. The decoder for $(U, U+V)$ codes that we just described when we want to reach large weights is given in details in Algorithm 2. We name it DecodeUV$(\cdot)$ and it parses its first argument $\mathbf{H}$ as a parity check matrix of some $(U, U+V)$ code. The decoding consists in two elementary steps, each using the basic Prange information set decoding. The second step is repeated in order to ensure that the final weight is w, the target for a valid signature. As it is described this algorithm may have a biased output and is susceptible to leak information on the secret key. Avoiding this is mandatory to obtain a proof in the GPV model. In the sequel we will show how to adapt the algorithm, in particular by adding rejection sampling, to avoid this leakage and complete the security proof.

Algorithm 2 DecodeUV$(\mathbf{H}, \mathbf{s})$ — Main signature building block

Parameters: $n, k, k_U, k_V = k - k_U, w$

Require: $\mathbf{H} = \begin{pmatrix} \mathbf{H}_U & \mathbf{0} \\ -\mathbf{H}_V & \mathbf{H}_V \end{pmatrix} \in \mathbb{F}_3^{(n-k)\times n}$, $\mathbf{H}_U \in \mathbb{F}_3^{(n/2-k_U)\times n/2}$, $\mathbf{s} = (\mathbf{s}_U, \mathbf{s}_V) \in \mathbb{F}_3^{n-k}$, $\mathbf{s}_U \in \mathbb{F}_3^{n/2-k_U}$

Ensure: $\mathbf{e}\mathbf{H}^\intercal = \mathbf{s}$ and $|\mathbf{e}| = w$ ▷ *implicit:* $\mathbf{H}_V \in \mathbb{F}_3^{(n/2-k_V)\times n/2}$ *and* $\mathbf{s}_V \in \mathbb{F}_3^{n/2-k_V}$

1: $\mathbf{e}_V \leftarrow \mathrm{D}_V(\mathbf{H}_V, \mathbf{s}_V)$
2: **repeat**
3: $\quad \mathbf{e}_U \leftarrow \mathrm{D}_U(\mathbf{H}_U, \mathbf{s}_U, \mathbf{e}_V)$
4: $\quad \mathbf{e} \leftarrow (\mathbf{e}_U, \mathbf{e}_U + \mathbf{e}_V)$
5: **until** $|\mathbf{e}| = w$
6: **return** $\mathbf{e}$

function $\mathrm{D}_V(\mathbf{H}_V, \mathbf{s}_V)$
$\quad \mathcal{I}_V \leftarrow$ InfoSet$(\mathbf{H}_V)$ ▷ InfoSet *returns an information set*
$\quad$ **return** PrangeStep$(\mathbf{H}_V, \mathbf{s}_V, \mathcal{I}_V, \mathbf{0})$

function $\mathrm{D}_U(\mathbf{H}_U, \mathbf{s}_U, \mathbf{e}_V)$
$\quad \mathcal{I}_U \leftarrow$ InfoSet$(\mathbf{H}_U)$ ▷ InfoSet *returns an information set*
$\quad \mathbf{x}_U \hookleftarrow \{\mathbf{x} \in \mathbb{F}_3^{n/2} \mid \mathrm{Supp}(\mathbf{x}) = \mathcal{I}_U \text{ and } \mathbf{x}_I = (\mathbf{e}_V)_I, I = \mathcal{I}_U \cap \mathrm{Supp}(\mathbf{e}_V)\}$
$\quad$ **return** PrangeStep$(\mathbf{H}_U, \mathbf{s}_U, \mathcal{I}_U, \mathbf{x}_U)$

All of this discussion is summarized in Figure 5 where we draw ω_{UV}^- and ω_{UV}^+ which are the highest and the smallest relative distances that our decoder can reach asymptotically in n when k/n is fixed and $q = 3$.

5 Obtaining a Preimage Sampleable Scheme

We restrict here our study to the case $q = 3$ but it can be generalized to larger values of q.

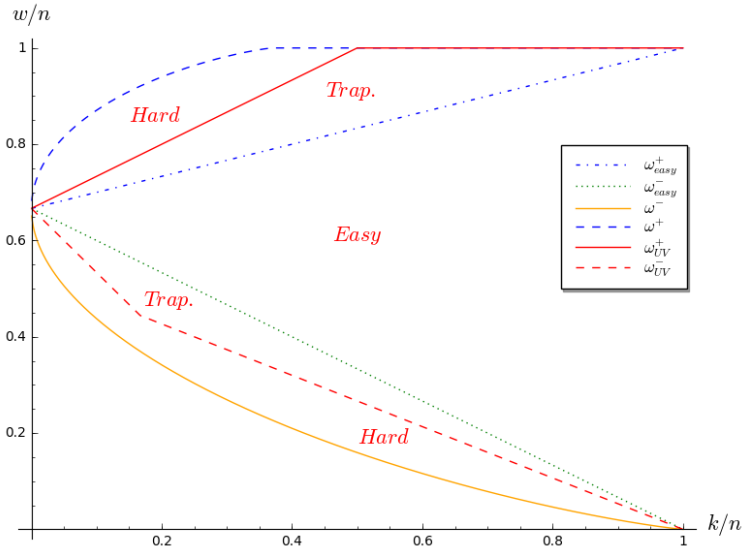
5.1 Achieving the Domain Sampling with the Generalized Admissible $(U, U+V)$ Code Family

We will denote in the rest of the article by $\mathbf{H}_{\mathrm{pk}}$ the random matrix chosen as the public parity-check matrix of our scheme $\mathcal{S}_{\mathrm{code}}$ which is defined in §3.1. Such a public-key is defined as:

$$\mathbf{H}_{\mathrm{pk}} = \mathbf{S}\mathbf{H}_{\mathrm{sk}}\mathbf{P} \quad \text{with} \quad \mathbf{H}_{\mathrm{sk}} \stackrel{\triangle}{=} \begin{pmatrix} \mathbf{H}_U\mathbf{D}_4\mathbf{M} & -\mathbf{H}_U\mathbf{D}_2\mathbf{M} \\ \mathbf{H}_V\mathbf{D}_3\mathbf{M} & -\mathbf{H}_V\mathbf{D}_1\mathbf{M} \end{pmatrix}$$

where $\mathbf{D}_1, \cdots, \mathbf{D}_4$ are four diagonal matrices which verify (12) and $\mathbf{M} \stackrel{\triangle}{=} (\mathbf{D}_1\mathbf{D}_4 - \mathbf{D}_3\mathbf{D}_2)^{-1}$, $\mathbf{S}$ is chosen uniformly at random among the invertible ternary matrices of size $(n-k)\times(n-k)$, $\mathbf{H}_U$ is chosen uniformly at random among the ternary matrices of size $(n/2-k_U)\times n/2$, $\mathbf{H}_V$ is chosen uniformly at random among the ternary matrices of size $(n/2-k_V)\times n/2$ and $\mathbf{P}$ is chosen uniformly at random among the permutation matrices of size $n \times n$. Thanks to the knowledge of

Fig. 5. Areas of relative signature distances with our trapdoor when $q = 3$



$\mathbf{H}_{\text{sk}}$ which is a parity-check matrix of an admissible generalized $(U, U+V)$-code and the previous algorithm we can invert $f_{\mathbf{H}_{\text{sk}}}$ on any input.

Let us give now the following definition which enables to better understand the structure of admissible generalized $(U, U+V)$-codes.

Definition 3. (number of V blocks of type I). *In a generalized $(U, U+V)$ code of length n associated to the 4-tuple of diagonal matrices of size $n/2$ $(\mathbf{D}_1, \mathbf{D}_2, \mathbf{D}_3, \mathbf{D}_4)$, the number of V blocks of type I, which we denote by n_I, is defined by:*

$$n_I \stackrel{\triangle}{=} |\{1 \leq i \leq n/2 : \mathbf{D}_2(i,i)\mathbf{D}_4(i,i) = 0\}| .$$

Remark 3. n_I can be viewed as the number of positions in which a codeword of the form $(\mathbf{vD}_2, \mathbf{vD}_4)$ is necessarily equal to 0: this comes from the fact that on a position where either $\mathbf{D}_2(i,i) = 0$ or $\mathbf{D}_4(i,i) = 0$, the other one is necessarily different from 0 as $\begin{pmatrix} \mathbf{D}_1 & \mathbf{D}_3 \\ \mathbf{D}_2 & \mathbf{D}_4 \end{pmatrix}$ is invertible. In other words we also have

$$n_I = |\{1 \leq i \leq n/2 : \mathbf{D}_2(i,i) = 0\}| + |\{1 \leq i \leq n/2 : \mathbf{D}_4(i,i) = 0\}| .$$

The random structure of such matrices $\mathbf{H}_{\text{pk}}$ (by choosing uniformly at random matrices $\mathbf{H}_U$ and $\mathbf{H}_V$) makes that the syndromes associated to matrices $\mathbf{H}_{\text{pk}}$ are indistinguishable in a very strong sense from random syndromes as the following proposition shows. In this way, our scheme achieves the Domain Sampling property of Definition 1.

Proposition 3. *Let $\mathcal{D}_w^{\mathbf{H}}$ be the distribution of the syndromes $\mathbf{eH}^\intercal$ when $\mathbf{e}$ is drawn uniformly at random among the ternary vectors of weight w and $\mathcal{U}$ be the uniform distribution over the syndrome space $\mathbb{F}_3^{n-k}$. We have*

$$\mathbb{E}_{\mathbf{H}_{\text{pk}}}\left(\rho(\mathcal{D}_w^{\mathbf{H}_{\text{pk}}}, \mathcal{U})\right) \leq \frac{1}{2}\sqrt{\varepsilon}$$

with

$$\varepsilon = \frac{3^{n-k}}{2^w\binom{n}{w}} + \sum_{j=0}^{w} \frac{3^{n/2-k_V}}{2^j\binom{n/2}{j}2^{2w}\binom{n/2}{w}^2} \left(\sum_{\substack{p=0 \\ w+p\equiv 0 \mod 2}}^{j} \binom{j}{p}\binom{n/2}{\frac{w+p}{2}}\binom{\frac{w+p}{2}}{j} 2^{\frac{w+3p}{2}} \right)^2 + 3^{n/2-k_U} \sum_{l=0}^{n_I} \binom{n_I}{l} 2^l \left(\frac{\binom{n-n_I}{w-l}2^{w-l}}{\binom{n}{w}2^w} \right)^2 \sum_{j=0}^{n/2-n_I} \frac{1}{2^j\binom{n/2-n_I}{j}} \left(\frac{1}{\binom{n}{w}2^w} \sum_{h=0}^{j} \sum_{p=0}^{n/2-n_I-j} \binom{n/2-n_I}{h}\binom{n/2-n_I-h}{j-h}\binom{n/2-n_I-j}{p}\binom{2n_I}{w-j-h-2p} 2^{w-j-h-2p} \right)^2 \tag{22}$$

Remark 4. In the paradigm of our code-based signatures we have w greater than the Gilbert-Varshamov bound, which gives $3^{n-k} \ll 2^w\binom{n}{w}$ and for the set of parameters we present in §9, $\log_2(\varepsilon) = -1034$.

The proof of this proposition is given in Appendix §B and relies among other thing on the following lemma which is a variation of the leftover hash lemma (see [BDK+11]) and which can be expressed as follows.

Lemma 1. *Consider a finite family $\mathcal{H} = (h_i)_{i\in I}$ of functions from a finite set E to a finite set F. Denote by ε the bias of the collision probability, i.e. the quantity such that*

$$\mathbb{P}_{h,e,e'}(h(e) = h(e')) = \frac{1}{|F|}(1+\varepsilon)$$

where h is drawn uniformly at random in $\mathcal{H}$, e and e' are drawn uniformly at random in E. Let $\mathcal{U}$ be the uniform distribution over F and $\mathcal{D}(h)$ be the distribution of the outputs $h(e)$ when e is chosen uniformly at random in E. We have

$$\mathbb{E}_h\left\{\rho(\mathcal{D}(h),\mathcal{U})\right\} \leq \frac{1}{2}\sqrt{\varepsilon}.$$

Remark 5. In the leftover hash lemma, there is the additional assumption that $\mathcal{H}$ is a universal family of hash functions, meaning that for any e and e' distinct in F, we have $\mathbb{P}_h(h(e) = h(e')) = \frac{1}{|F|}$. This assumption allows to have a general bound on the bias ε. In our case, where the h's are hash functions defined as $h(\mathbf{e}) = \mathbf{e}\mathbf{H}_{\text{pk}}^{\intercal}$, $\mathcal{H}$ does not form a universal family of hash functions (essentially because the distribution of the $\mathbf{H}_{\text{pk}}$'s is not the uniform distribution over $\mathbb{F}_3^{(n-k)\times n}$). However in our case we can still bound ε by a direct computation. This lemma is proved in Appendix B.

5.2 Achieving a Uniformly Distributed Output

To be a one-way preimage sampleable function, we have to enforce that the outputs are very close to be uniformly distributed over S_w. Algorithm 2 using directly the Prange decoder, does not meet this property. However, by changing it slightly, we will achieve this task by still keeping the property to output errors of weight w for which it is hard to solve the decoding problem for this weight. We summarize the situation in Figure 6.

The template remains the same but the functions D_U and D_V will be modified to included some rejection sampling. The Prange decoders that are used here achieve in a natural way some kind of uniformity on the output.

Definition 4 (uniform decoder). *A V-decoder $\mathrm{D}(\mathbf{H}_V, \mathbf{s}_V)$ taking as input $\mathbf{H}_V$ in $\mathbb{F}_3^{(n/2-k_V)\times n}$, $\mathbf{s}_V$ in $\mathbb{F}_3^{n/2-k_V}$ and outputting $\mathbf{e}_V$ in $\mathbb{F}_3^{n/2}$ satisfying $\mathbf{e}_V\mathbf{H}_V{}^{\intercal} = \mathbf{s}_V$ is uniform with respect to $\mathbf{H}_V$ if $\mathbb{P}(\mathbf{e}_V = \mathrm{D}(\mathbf{H}_V, \mathbf{s}_V))$ is just a function of $|\mathbf{e}|$ when $\mathbf{s}_V$ is chosen uniformly at random in $\mathbb{F}_3^{n/2-k_V}$.*

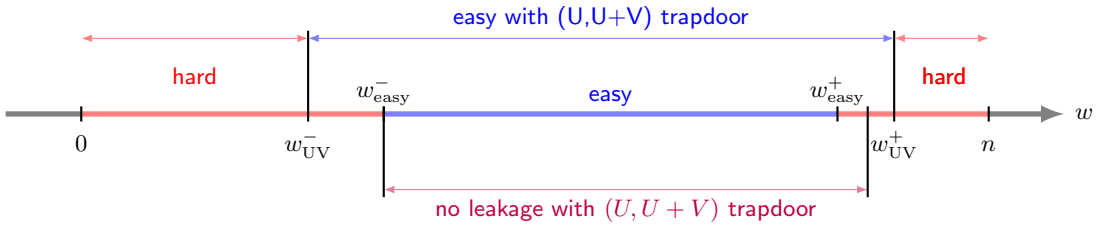


Fig. 6. Hardness of (U,U+V) Decoding with no leakage of signature

The Prange decoder used for producing $\mathbf{e}_U$ does not have such a nice property, its output depends in a crucial way on $\mathbf{e}_V$. The following notation will be useful.

Notation 1 *For $\mathbf{x} \in \mathbb{F}_3^n$ with n even, let $\mathbf{x}_U$ and $\mathbf{x}_V$ be in $\mathbb{F}_3^{n/2}$ such that $\mathbf{x} = (\mathbf{x}_U, \mathbf{x}_U + \mathbf{x}_V)$. We also denote by $\ell_1(\mathbf{x})$ and $\ell_{-1}(\mathbf{x})$ the following quantities:*

$$\ell_1(\mathbf{x}) \stackrel{\triangle}{=} |\{i \mid \mathbf{x}_V(i) = 1, \mathbf{x}_U(i) \neq 1\}|$$

$$\ell_{-1}(\mathbf{x}) \stackrel{\triangle}{=} |\{i \mid \mathbf{x}_V(i) = -1, \mathbf{x}_U(i) \neq -1\}|.$$

Definition 5 (weakly uniform decoder). *Let* D *be a decoder with input $\mathbf{H}_U$ in $\mathbb{F}_3^{(n/2-k_U)\times n}$, $\mathbf{s}_U$ in $\mathbb{F}_3^{n/2-k_U}$ and $\mathbf{e}_V$ in $\mathbb{F}_3^{n/2}$ and that outputs an $\mathbf{e} = (\mathbf{e}_U, \mathbf{e}_U + \mathbf{e}_V)$ in $\mathbb{F}_3^n$ such that $\mathbf{e}_U \mathbf{H}_U{}^\intercal = \mathbf{s}_U$. Let $f(\mathbf{e}) \stackrel{\triangle}{=} (|\mathbf{e}_V|_1, |\mathbf{e}_V|_{-1}, \ell_1(\mathbf{e}), \ell_{-1}(\mathbf{e}), |\mathbf{e}|)$. D is weakly uniform with respect to $\mathbf{H}_U$ if $\mathbb{P}(\mathbf{e} = \mathrm{D}(\mathbf{H}_U, \mathbf{s}_U, \mathbf{e}_V))$ is a function of $f(\mathbf{e})$ when $\mathbf{s}_U$ and $\mathbf{e}_V$ are chosen uniformly at random in their range.*

Our U-decoder based on the Prange decoder meets this property in a natural way. With such decoders, it is rather easy to obtain a uniformly distributed output for the combined $(U, U+V)$-decoder. The uniform distribution of our decoder follows at once from

Lemma 2. *Let $\mathbf{e} = (\mathbf{e}_U, \mathbf{e}_U + \mathbf{e}_V)$ be the output of Algorithm 2 when $\mathbf{s}_V$ and $\mathbf{s}_U$ were chosen uniformly at random in $\mathbb{F}_3^{n/2-k_V}$ and $\mathbb{F}_3^{n/2-k_U}$ respectively. Assume that D_U is weakly uniform whereas D_V is uniform. Let*

$$\mathbf{e}^{unif} = (\mathbf{e}_U^{unif}, \mathbf{e}_U^{unif} + \mathbf{e}_V^{unif})$$

be a uniformly distributed error of weight w. Let

$$g(\mathbf{e}_V) \stackrel{\triangle}{=} (|\mathbf{e}_V|_1, |\mathbf{e}_V|_{-1}) \quad \text{and} \quad h(\mathbf{e}) \stackrel{\triangle}{=} (\ell_1(\mathbf{e}), \ell_{-1}(\mathbf{e}), |\mathbf{e}|).$$

If $\rho(|\mathbf{e}_V|, |\mathbf{e}_V^{unif}|) = 0$ and $\mathbb{P}(h(\mathbf{e}) = z | g(\mathbf{e}_V) = y) = \mathbb{P}(h(\mathbf{e}^{unif}) = z | g(\mathbf{e}_V^{unif}) = y)$ for any possible y and z, then

$$\rho(\mathbf{e}, \mathbf{e}^{unif}) = 0.$$

Proof. Let $\mathbf{x}$ be in S_w, $\mathbf{H}_U$ be the matrix used in D_U and $f(\mathbf{e}) \stackrel{\triangle}{=} (g(\mathbf{e}), h(\mathbf{e}))$. We have

$$\mathbb{P}_{\mathbf{s}_U, \mathbf{s}_V}(\mathbf{e} = \mathbf{x}) = \mathbb{P}_{\mathbf{s}_U, \mathbf{s}_V}(\mathbf{e} = \mathbf{x} | f(\mathbf{e}) = (y, z)) \mathbb{P}_{\mathbf{s}_U, \mathbf{s}_V}(f(\mathbf{e}) = (y, z)),$$

where $f(\mathbf{x}) = (y, z)$. Observe now that

$$\begin{aligned} \mathbb{P}_{\mathbf{s}_U, \mathbf{s}_V}(f(\mathbf{e}) = (y, z)) &= \mathbb{P}_{\mathbf{s}_U, \mathbf{s}_V}(h(\mathbf{e}) = z | g(\mathbf{e}_V) = y) \mathbb{P}_{\mathbf{s}_V}(g(\mathbf{e}_V) = y) \\ &= \mathbb{P}(h(\mathbf{e}^{\text{unif}}) = z | g(\mathbf{e}_V^{\text{unif}}) = y) \mathbb{P}(g(\mathbf{e}_V^{\text{unif}}) = y). \end{aligned}$$

where $\mathbb{P}_{\mathbf{s}_V}(g(\mathbf{e}) = y) = \mathbb{P}(g(\mathbf{e}^{\text{unif}}) = y)$ because $\mathbf{e}_V$ and $\mathbf{e}_V^{\text{unif}}$ have the same distribution. This follows from the uniformity of D_V and $\rho(|\mathbf{e}_V|, |\mathbf{e}_V^{\text{unif}}|) = 0$. Since $\mathbf{e}_V$ is uniformly distributed conditioned on its weight, we have

$$\mathbb{P}_{\mathbf{s}_U, \mathbf{s}_V}(\mathbf{e} = \mathbf{x} | f(\mathbf{e}) = (y, z)) = \mathbb{P}_{\mathbf{s}_U, \mathbf{e}_V}(\mathbf{e} = \mathrm{D}_U(\mathbf{H}_U, \mathbf{s}_U, \mathbf{e}_V) | f(\mathbf{e}) = f(\mathbf{x})) = \mathbb{P}(\mathbf{e}^{\text{unif}} = \mathbf{x} | f(\mathbf{e}^{\text{unif}}) = f(\mathbf{x})).$$

where the last equality follows from the weak uniformity of D_U. Using all these equations, we get

$$\begin{aligned}\mathbb{P}_{\mathbf{s}_U,\mathbf{s}_V}(\mathbf{e}=\mathbf{x}) &= \mathbb{P}(\mathbf{e}^{\mathrm{unif}}=\mathbf{x}|f(\mathbf{e}^{\mathrm{unif}})=(y,z))\mathbb{P}(h(\mathbf{e}^{\mathrm{unif}})=z|g(\mathbf{e}_V^{\mathrm{unif}})=y)\mathbb{P}(g(\mathbf{e}_V^{\mathrm{unif}})=y)\\ &= \mathbb{P}\left(\mathbf{e}^{\mathrm{unif}}=\mathbf{x}\right)\end{aligned}$$

which concludes the proof.

The rationale of our algorithm is then to ensure that D_U and D_V behave as required by Lemma 2 by some mild rejection sampling. We summarize how we perform the decoding in Figure 7. It relies here among other thing on the crucial notion of information set in the Prange algorithm. The rejection sampling will be over the weight of $\mathbf{e}_V$ which is the output of D_V and over $\ell_1(\mathbf{e}), \ell_{-1}(\mathbf{e})$ which are functions of the output of D_U.

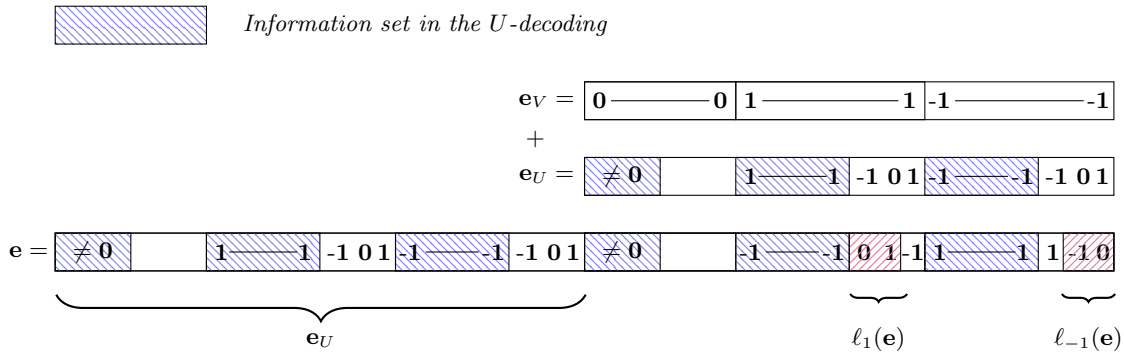


Fig. 7. Summary of the decoding

The pseudo-code of D_V is given in Algorithm 3. The algorithm for the U-decoder is slightly

Algorithm 3 D_V the V-decoder outputting an $\mathbf{e}_V$ such that $\mathbf{e}_V\mathbf{H}_V^\intercal = \mathbf{s}_V$.

Parameters: n, k_V $n/2$ is the length of the code we decode, k_V its dimension
$\mathcal{D}_V$ a distribution over $[\![0, k_V]\!]$
$(\mathbf{r}_V)_i$ rejection sampling vector, taking values in $[0,1]$, $\textsc{accept}(i, \mathbf{r})$ is true with probability r_i

```
function D_V(H_V, s_V)
    repeat
        I ← InfoSet(H_V)
        t ↩ D_V
        x ↩ {x ∈ F_3^{k_V} | |x| = t}
        e_V ← PrangeStep(H_V, s_V, I, x)
    until accept(|e_V|, r_V)
    return e_V
```

more involved and is given in Algorithm 4. By setting up the rejection vector $\mathbf{r}_U$ and $\mathbf{r}_V$ appropriately, we can reach the uniform distribution over S_w in Algorithm 2. Rejection sampling on D_V is here to meet the first condition in Lemma 2: $\rho(|\mathbf{e}_V|, |\mathbf{e}_V^{\mathrm{unif}}|) = 0$, whereas rejection sampling on D_U ensures that the conditional distribution of $h(\mathbf{e})$ given $g(\mathbf{e})$ is the same as the distribution of $h(\mathbf{e}^{\mathrm{unif}})$ given $g(\mathbf{e}^{\mathrm{unif}})$. This is shown in Theorem 1.

Theorem 1. *Let $\mathbf{e}^{unif}$ be a vector chosen uniformly at random over S_w. Let $\mathbf{r}_V$ be defined as for any $i \in \{0, \cdots, n/2\}$ as*

$$r_V(i) \stackrel{\triangle}{=} \frac{1}{M_V^{rs}} \frac{q_1^{unif}(i)}{q_1(i)}$$

Algorithm 4 D_U the U-decoder outputting an $\mathbf{e}_U$ such that $\mathbf{e}_U\mathbf{H}_U^\intercal = \mathbf{s}_U$.

Parameters: n, k_U $n/2$ is the length of the code we decode, k_U its dimension
$(\mathcal{D}_U^{t_1,t_{-1}})_{\substack{0\le t_1\le n/2\\ 0\le t_{-1}\le n/2}}$ a family of probability distribution over $\{(j_1,j_{-1})|0\le j_1\le t_1, 0\le j_{-1}\le t_{-1}, 0\le j_1+j_{-1}\le k_u\}$
$(\mathbf{r}_U^{t_1,t_{-1}})_{\substack{0\le t_1\le n/2\\ 0\le t_{-1}\le n/2}}$ a family of rejection sampling vectors, taking values in $[0,1]$, $\textsc{accept}((i,j),\mathbf{r})$ is true with probability $r(i,j)$

function $\mathrm{D}_U(\mathbf{H}_U, \mathbf{s}_U, \mathbf{e}_V)$
  $t_1, t_{-1} \leftarrow |\mathbf{e}_V|_1, |\mathbf{e}_V|_{-1}$
  repeat
    $j_1, j_{-1} \hookleftarrow \mathcal{D}_U^{t_1,t_{-1}}$
    $\mathcal{I} \leftarrow \textsc{InfoSetW}(\mathbf{H}_U, \mathbf{e}_V, j_1, j_{-1})$ ▷ $\textsc{InfoSetW}()$ *is defined below*
    $\mathbf{x} \hookleftarrow \{\mathbf{x}\in\mathbb{F}_3^{n/2} \mid \mathrm{Supp}(\mathbf{x}) = \mathcal{I} \text{ and } \mathbf{x}_\mathcal{J} = (\mathbf{e}_V)_\mathcal{J}, \mathcal{J} = \mathcal{I}\cap\mathrm{Supp}(\mathbf{e}_V)\}$
    $\mathbf{e}_U \leftarrow \textsc{PrangeStep}(\mathbf{H}_U, \mathbf{s}_U, \mathcal{I}, \mathbf{x})$
    $\ell_1, \ell_{-1} \leftarrow |\{i \mid \mathbf{e}_V(i) = 1, \mathbf{e}_U(i) \neq 1\}|, |\{i \mid \mathbf{e}_V(i) = -1, \mathbf{e}_U(i) \neq -1\}|$
  until $\textsc{accept}((\ell_1,\ell_{-1}), \mathbf{r}_U)$
  return $\mathbf{e}_U$

function $\textsc{InfoSetW}(\mathbf{H}, \mathbf{e}, j_1, j_{-1})$ — weighted information set

Require: $\mathbf{H}\in\mathbb{F}_3^{(n-k)\times n}$, $\mathbf{e}\in\mathbb{F}_3^n$, j_1, j_{-1} positive integers such that $j_1 + j_{-1} \le k$
Ensure: the returned value $\mathcal{I}$ is an information set of $\langle\mathbf{H}\rangle^\perp$ such that the number of positions i in $\mathcal{I}$ for which $\mathbf{e}_i = b$ is equal to j_b for b in $\{1,-1\}$.

with $q_1^{unif}(i) = \mathbb{P}(|\mathbf{e}_V^{unif}| = i)$, $q_1(i) \triangleq \mathbb{P}(|\mathrm{D}_V(\mathbf{H}_V, \mathbf{s}_V)| = i)$, *and* $M_V^{rs} \triangleq \sup_{0\le i\le n/2} \frac{q_1^{unif}(i)}{q_1(i)}$. *The no-rejection probability vector* $\mathbf{r}_U^{t_1,t_{-1}}$ *is chosen for any* $i\in[\![0,t_1]\!]$ *and* $j\in[\![0,t_{-1}]\!]$ *as*

$$r_U^{t_1,t_{-1}}(i,j) \triangleq \frac{1}{M_U^{rs}(t_1,t_{-1})}\frac{q_2^{unif}(i,j|t_1,t_{-1})}{q_2(i,j|t_1,t_{-1})}$$

with

$$q_2^{unif}(i,j|t_1,t_{-1}) \triangleq \mathbb{P}\left(\ell_1(\mathbf{e}^{unif}) = i, \ell_{-1}(\mathbf{e}^{unif}) = j \;\middle|\; |\mathbf{e}_V^{unif}|_b = t_b,\ b=-1,1\}\right)$$
$$q_2(i,j|t_1,t_{-1}) \triangleq \mathbb{P}(\ell_1(\tilde{\mathbf{e}}) = i, \ell_{-1}(\tilde{\mathbf{e}}) = j \;|\; |\tilde{\mathbf{e}}_V|_b = t_b,\ b=-1,1)$$
$$M_U^{rs}(t_1,t_{-1}) \triangleq \sup_{\substack{0\le i\le t_1\\ 0\le j\le t_{-1}}} \frac{q_2^{unif}(i,j|t_1,t_{-1})}{q_2(i,j|t_1,t_{-1})},$$

where $\tilde{\mathbf{e}} = (\tilde{\mathbf{e}}_U, \tilde{\mathbf{e}}_U + \mathbf{e}_V^{unif})$ *with* $\tilde{\mathbf{e}}_U$ *being the output of* $\textsc{PrangeStep}(\mathbf{H}_U, \mathbf{s}_U, \mathcal{I}, \mathbf{x})$ *in Algorithm 4 and* $(\mathbf{H}_U, \mathbf{s}_U, \mathbf{e}_V^{unif})$ *is its input, where* $\mathbf{s}_U$ *is uniformly distributed. Then if* D_U *is weakly uniform and* D_V *uniform, we have that the output* $\mathbf{e}$ *of Algorithm 2 satisfies*

$$\rho(\mathbf{e}, \mathbf{e}^{unif}) = 0.$$

To have an efficient algorithm, it is essential that $M_{rs}(1)$ and the $M_{rs}(t_1,t_{-1})$'s are as small as possible. These numbers represent the average number of times the `repeat` loops in those algorithms are performed. This is achieved by a particular choice of the parameters w, k_U, k_V. Roughly speaking, the idea for having $M_{rs}(1)$ and the $M_{rs}(t_1,t_{-1})$'s to be small is that the output $\tilde{\mathbf{e}_V}$ of $\textsc{PrangeStep}(\mathbf{H}_V, \mathbf{s}_V, \mathcal{I}, \mathbf{x})$ in Algorithm 3 and the output $\tilde{\mathbf{e}_U}$ of $\textsc{PrangeStep}(\mathbf{H}_U, \mathbf{s}_U, \mathcal{I}, \mathbf{x})$ in Algorithm 4 satisfy

$$\mathbb{E}(|\tilde{\mathbf{e}_V}|) = \mathbb{E}(|\mathbf{e}_V^{\mathrm{unif}}|) \quad \text{and} \quad \mathbb{E}(|\tilde{\mathbf{e}_U}|) = \mathbb{E}(|\mathbf{e}_U^{\mathrm{unif}}|).$$

We also require that $\mathbb{E}(|(\tilde{\mathbf{e}_U}, \tilde{\mathbf{e}_U} + \mathbf{e}_V^{\mathrm{unif}})|) = \mathbb{E}(|\mathbf{e}^{\mathrm{unif}}|)$. Set α by

$$(1-\alpha)k_V = \mathbb{E}(T),$$

where T is a random variable distributed like $\mathcal{D}_V$. These requirements lead to the following choice of parameters

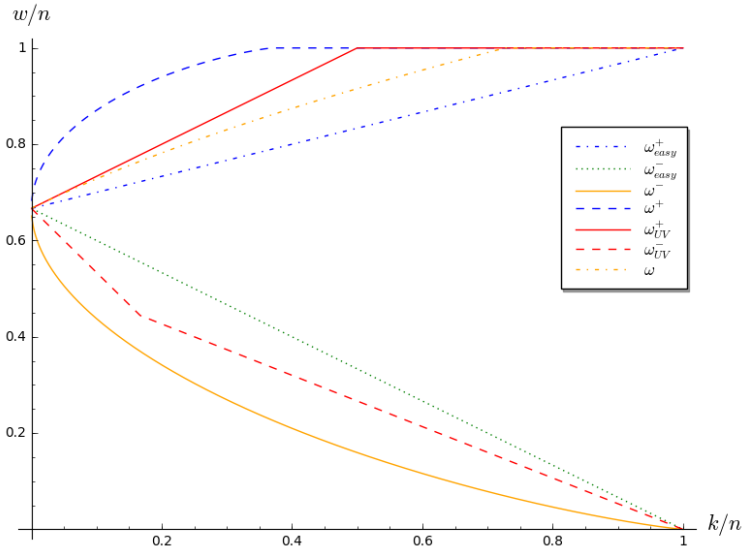
$$w = \left\lfloor n\left(1-\alpha+\frac{1}{3}\sqrt{(3\alpha-1)\left(3\alpha+4\frac{k}{n}-1\right)}\right)\right\rfloor \tag{23}$$

$$k_V = \left\lfloor \frac{n}{2}\frac{3}{3\alpha-1}\left(\left(1-\frac{w}{n}\right)^2+\frac{1}{2}\left(\frac{w}{n}\right)^2-\frac{1}{3}\right)\right\rfloor \tag{24}$$

$$k_U = \left\lfloor \frac{n}{2}\left(-2+3\frac{w}{n}\right)\right\rfloor \tag{25}$$

Figure 8 gives the relative error weight $\omega \stackrel{\triangle}{=} w/n$ as a function of R for $\alpha = 0.7$ of Algorithm 2 with rejection sampling and the previous choice of parameters asymptotically in n when k/n is fixed.

Fig. 8. Areas of relative signature distances with rejection sampling when $q = 3$



6 Security Proof

We give in this section a security proof of the signature scheme $\mathcal{S}_{\text{code}}$. This proof is extremely close in spirit of the security proof of [GPV08]. However we do not reduce the security of $\mathcal{S}_{\text{code}}$ to a problem of collision as the scheme imposes to be in a range of parameters where this problem becomes easy (for more details see §7.3) .

Furthermore, we would like to stress that this section is a slightly different version of the unpublished paper [DST17b, Section 3]. We added in order to have a self-contain paper and not to refer many times to results of [DST17b].

6.1 Basic Tools

Basic Definitions. Recall that the statistical distance ρ is defined in Section §2. We will need the following well known property for the statistical distance which can be easily proved by induction.

Proposition 4. *Let $(\mathcal{D}_1^0, \ldots, \mathcal{D}_n^0)$ and $(\mathcal{D}_1^1, \ldots, \mathcal{D}_n^1)$ be two n-tuples of discrete probability distributions where $\mathcal{D}_i^0$ and $\mathcal{D}_i^1$ are distributed over a same space $\mathcal{E}_i$. We have for all positive integers n:*

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

A *distinguisher* between two distributions $\mathcal{D}^0$ and $\mathcal{D}^1$ over the same space $\mathcal{E}$ is a randomized algorithm which takes as input an element of $\mathcal{E}$ that follows the distribution $\mathcal{D}^0$ or $\mathcal{D}^1$ and outputs $b \in \{0, 1\}$. It is characterized by its advantage:

$$Adv^{\mathcal{D}^0,\mathcal{D}^1}(\mathcal{A}) \triangleq \mathbb{P}_{\xi\sim\mathcal{D}^0}\left(\mathcal{A}(\xi) \text{ outputs } 1\right) - \mathbb{P}_{\xi\sim\mathcal{D}^1}\left(\mathcal{A}(\xi) \text{ outputs } 1\right).$$

We call this quantity the *advantage* of $\mathcal{A}$ against $\mathcal{D}^0$ and $\mathcal{D}^1$.

Definition 6 (Computational Distance and Indistinguishability). *The computational distance between two distributions $\mathcal{D}^0$ and $\mathcal{D}^1$ in time t is:*

$$\rho_c\left(\mathcal{D}^0, \mathcal{D}^1\right)(t) \triangleq \max_{|\mathcal{A}|\leq t}\left\{Adv^{\mathcal{D}^0,\mathcal{D}^1}(\mathcal{A})\right\}$$

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

The ensembles $\mathcal{D}^0 = (\mathcal{D}_n^0)$ and $\mathcal{D}^1 = (\mathcal{D}_n^1)$ 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.

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 7 (Signature Scheme). *A signature scheme $\mathcal{S}$ 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 $\sigma = \mathtt{Sgn}^{\mathrm{sk}}(\mathbf{m})$;*
- *The verification algorithm takes as input a message $\mathbf{m}$ and a signature σ. It returns $\mathtt{Vrfy}^{\mathrm{pk}}(\mathbf{m}, \sigma)$ which is 1 if the signature is accepted and 0 otherwise. It is required that $\mathtt{Vrfy}^{\mathrm{pk}}(\mathbf{m}, \sigma) = 1$ if $\sigma = \mathtt{Sgn}^{\mathrm{sk}}(\mathbf{m})$.*

For this kind of scheme, one of the strongest security notion is *existential unforgeability under an adaptive chosen message attack* (EUF-CMA). In this model the adversary has access to all signatures of its choice and its goal is to produce a valid forgery. A valid forgery is a message/signature pair $(\mathbf{m}, \sigma)$ such that $\mathtt{Vrfy}^{\mathrm{pk}}(\mathbf{m}, \sigma) = 1$ 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 8 (EUF-CMA Security). *Let $\mathcal{S}$ be a signature scheme.*
A forger $\mathcal{A}$ is a $(t, q_{\mathrm{hash}}, q_{\mathrm{sign}}, \varepsilon)$-adversary in EUF-CMA *against $\mathcal{S}$ 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 $\mathcal{S}$ as:*

$$Succ_{\mathcal{S}}^{\text{EUF-CMA}}(t, q_{\mathrm{hash}}, q_{\mathrm{sign}}) \triangleq \max\left(\varepsilon | \textit{it exists a } (t, q_{\mathrm{hash}}, q_{\mathrm{sign}}, \varepsilon)\textit{-adversary}\right).$$

The signature scheme $\mathcal{S}$ is said to be $(t, q_{\mathrm{hash}}, q_{\mathrm{sign}})$-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 of 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 $\mathcal{A}$ 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 $\mathcal{A}$ has terminated. The output of $\mathcal{A}$ is given as input to this procedure.

The output of the game G, which is denoted $G(\mathcal{A})$, is the output of the finalization procedure (which is a bit $b \in \{0,1\}$). The game G with $\mathcal{A}$ is said to be successful if $G(\mathcal{A}) = 1$. 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 $\mathcal{A}$ 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 $\mathcal{A}$ 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 9 (challenger procedures in the EUF-CMA Game). *The challenger procedures for the* EUF-CMA *Game corresponding to $\mathcal{S}_{code}$ are defined as:*

`proc Initialize`(λ)	`proc Hash`$(\mathbf{m}, \mathbf{r})$	`proc Sign`$(\mathbf{m})$	`proc Finalize`$(\mathbf{m}, \mathbf{e}, \mathbf{r})$
$(pk, sk) \leftarrow$ `Gen`(1^λ)	`return Hash`$(\mathbf{m}, \mathbf{r})$	$\mathbf{r} \hookleftarrow \{0,1\}^{\lambda_0}$	$\mathbf{s} \leftarrow$ `Hash`$(\mathbf{m}, \mathbf{r})$
$\mathbf{H}_{\text{pk}} \leftarrow pk$		$\mathbf{s} \leftarrow$ `Hash`$(\mathbf{m}, \mathbf{r})$	`return`
$(\mathbf{H}_{\text{sk}}, \mathbf{P}, \mathbf{S}) \leftarrow sk$		$\mathbf{e} \leftarrow D_{\mathbf{H}_{\text{sk}}, w}(\mathbf{s}(\mathbf{S}^{-1})^\intercal)$	$\mathbf{e}\mathbf{H}_{\text{pk}}^\intercal = \mathbf{s} \wedge \lvert\mathbf{e}\rvert = w$
`return` $\mathbf{H}_{\text{pk}}$		`return` $(\mathbf{e}\mathbf{P}, \mathbf{r})$	

6.2 Code-Based Problems

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 analysed in [Sen11]. We will come back to the best known algorithms to solve this problem as a function of the distance w in §7.

Problem 2. (DOOM – Decoding One Out of Many). Given $\mathbf{H} \in \mathbb{F}_q^{(n-k)\times n}$, $\mathbf{s}_1, \cdots, \mathbf{s}_N \in \mathbb{F}_q^{n-k}$, and an integer w, find $\mathbf{e} \in \mathbb{F}_q^n$ and $i, 1 \leq i \leq N$ such that such that $\mathbf{e}\mathbf{H}^\intercal = \mathbf{s}_i$ and $|\mathbf{e}| = w$.

Definition 10 (One-Wayness of DOOM). *We define the success of an algorithm $\mathcal{A}$ against* DOOM *with the parameters n, k, N, w as:*

$$Succ_{\text{DOOM}}^{n,k,N,w}(\mathcal{A}) = \mathbb{P}\big(\mathcal{A}(\mathbf{H}, \mathbf{s}_1, \cdots, \mathbf{s}_N) \textit{ solution of } \text{DOOM}\big)$$

where $\mathbf{H}$ is chosen uniformly at random in $\mathbb{F}_q^{(n-k)\times n}$, the $\mathbf{s}_i$'s are chosen uniformly at random in $\mathbb{F}_q^{n-k}$ and the probability is taken over these choices of $\mathbf{H}$, the $\mathbf{s}_i$'s and the internal coins of $\mathcal{A}$.

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

$$Succ_{\text{DOOM}}^{n,k,N,w}(t) = \max_{|\mathcal{A}| \leq t} \left\{ Succ_{\text{DOOM}}^{n,k,N,w}(\mathcal{A}) \right\}.$$

Another problem will appear in the security proof: distinguish random codes from a code drawn uniformly at random in the family used for public keys in the signature scheme. The public-keys, namely matrices $\mathbf{H}_{\text{pk}}$, follow a distribution over the parity-check matrices of size $(n-k_U-k_V)\times n$ which is described in §5.1. In the following we will denote by $\mathcal{D}_{\text{pub}}$ this distribution. On the other hand $\mathcal{D}_{\text{rand}}$ will denote the uniform distribution over the parity-check matrices of all $[n,k]$-codes with $k=k_U+k_V$. We will discuss about the difficulty of the task to distinguish $\mathcal{D}_{\text{pub}}$ and $\mathcal{D}_{\text{rand}}$ in §8. Let us recall that the syndromes associated to matrices $\mathbf{H}_{\text{pk}}$ are indistinguishable in a very strong sense from random syndromes as the proposition 3 (see §5.1) shows in the case of $q=3$.

6.3 EUF-CMA Security Proof

This subsection is devoted to our security reduction and its proof. We give it in the case of $q=3$. Let us first introduce some notations that will be used. We will denote by $\mathcal{D}_w$ the output distribution of Algorithm 2. Furthermore, we will denote Algorithm 2 by $\mathrm{D}_{\mathbf{H}_{\text{sk}},\text{w}}(\cdot)$ for a secret key $\mathbf{H}_{\text{sk}}$. Recall that $\mathcal{U}_w$ is the uniform distribution over S_w (which is the set of words of weight w in $\mathbb{F}_3^n$), $\mathcal{D}_{\text{pub}}$ is the distribution of public keys, $\mathcal{D}_{\text{rand}}$ is the uniform distribution over parity-check matrices of all $[n,k]$-codes and $\mathcal{S}_{\text{code}}$ is our signature scheme defined in §3.1 with the family of generalized admissible $(U,U+V)$-codes (Definition 2 in §4.2).

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

$$Succ^{\text{EUF-CMA}}_{\mathcal{S}_{\text{code}}}(t,q_{\text{hash}},q_{\text{sign}}) \le 2Succ^{n,k,q_{\text{hash}},w}_{\text{DOOM}}(t_c)+\rho_c\left(\mathcal{D}_{\text{rand}},\mathcal{D}_{\text{pub}}\right)(t_c) + q_{\text{sign}}\rho\left(\mathcal{D}_w,\mathcal{U}_w\right)+\frac{1}{2}q_{\text{hash}}\sqrt{\varepsilon}+\frac{1}{2^\lambda}$$

where $t_c=t+O\left(q_{\text{hash}}\cdot n^2\right)$ and ε given in Proposition 3.

Proof. Let $\mathcal{A}$ be a $(t,q_{\text{sign}},q_{\text{hash}},\varepsilon)$-adversary in the EUF-CMA model against $\mathcal{S}_{\text{code}}$ and let $(\mathbf{H}_0,\mathbf{s}_1,\cdots,\mathbf{s}_{q_{\text{hash}}})$ be drawn uniformly at random among all instances of DOOM for parameters n,k,q_{hash},w. We stress here that syndromes $\mathbf{s}_j$ are random and independent vectors of $\mathbb{F}_3^{n-k}$. We write $\mathbb{P}(S_i)$ to denote the probability of success for $\mathcal{A}$ of game G_i. Let

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

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:

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

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

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

Game 2 is modified from Game 1 as follows:

proc Hash($\mathbf{m},\mathbf{r}$)	proc Sign($\mathbf{m}$)
if $\mathbf{r}\in L_{\mathbf{m}}$	$\mathbf{r}\leftarrow L_{\mathbf{m}}$.next()
$\quad\mathbf{e}_{\mathbf{m},\mathbf{r}}\hookleftarrow S_w$	$\mathbf{s}\leftarrow$ Hash($\mathbf{m},\mathbf{r}$)
$\quad$return $\mathbf{e}_{\mathbf{m},\mathbf{r}}\mathbf{H}_{\text{pk}}^\intercal$	$\mathbf{e}\leftarrow\mathrm{D}_{\mathbf{H}_{\text{sk}},\text{w}}(\mathbf{s}(\mathbf{S}^{-1})^\intercal)$
else	return $(\mathbf{eP},\mathbf{r})$
$\quad j\leftarrow j+1$	
$\quad$return $\mathbf{s}_j$	

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 $\mathbf{r}\in L_{\mathbf{m}}$ returns true if and only if $\mathbf{r}$ is in the list. 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 $\mathbf{e}_{\mathbf{m},\mathbf{r}}\mathbf{H}_{\text{pk}}^{\intercal}$ with $\mathbf{e}_{\mathbf{m},\mathbf{r}} \hookleftarrow S_w$. Although we do not use it in this game, we remark that $(\mathbf{e}_{\mathbf{m},\mathbf{r}}, \mathbf{r})$ is a valid signature for $\mathbf{m}$. The error value is stored. If $\mathbf{r} \notin L_{\mathbf{m}}$ it outputs one of $\mathbf{s}_j$ of the instance $(\mathbf{H}_0, \mathbf{s}_1, \ldots, \mathbf{s}_{q_{\text{hash}}})$ 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 4.

$$\mathbb{P}(S_1) \leq \mathbb{P}(S_2) + \frac{q_{\text{hash}}}{2}\sqrt{\varepsilon} \text{ where } \varepsilon \text{ is given in Proposition 3.}$$

The proof of this lemma is given at the end of Appendix B and relies among other things on the following points:

- Proposition 4;
- Syndromes produced by matrices $\mathbf{H}_{\text{pk}}$ with errors of weight w have average statistical distance from the uniform distribution over $\mathbb{F}_3^{n-k}$ at most $\frac{1}{2}\sqrt{\varepsilon}$ (see Proposition 3).

Game 3 differs from Game 2 by changing in `proc Sign` calls "$\mathbf{e} \leftarrow \mathrm{D}_{\mathbf{H}_{\text{sk}},w}(\mathbf{s}(\mathbf{S}^{-1})^{\intercal})$" by "$\mathbf{e} \leftarrow \mathbf{e}_{\mathbf{m},\mathbf{r}}$" and "return $(\mathbf{eP}, \mathbf{r})$" by "return $(\mathbf{e}, \mathbf{r})$". Any signature $(\mathbf{e}, \mathbf{r})$ produced by `proc Sign` is valid. The error $\mathbf{e}$ is drawn according to the uniform distribution $\mathcal{U}_w$ while previously it was drawn according to Algorithm 2 distribution, that is $\mathcal{D}_w$. By using Proposition 4 it follows that

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

Game 4 is the game where we replace the public matrix $\mathbf{H}_{\text{pk}}$ by $\mathbf{H}_0$. 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 distributions $\mathcal{D}_{\text{rand}}$ and $\mathcal{D}_{\text{pub}}$:

$$\mathbb{P}(S_3) \leq \mathbb{P}(S_4) + \rho_c(\mathcal{D}_{\text{pub}}, \mathcal{D}_{\text{rand}})(t_c).$$

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 5 differs in the finalize procedure.

```
proc Finalize(m, e, r)
s ← Hash(m, r)
b ← eH_pk^T = s ∧ |e| = w
return b ∧ r ∉ L_m
```

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

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

Therefore

$$\mathbb{P}(S_5) \geq \frac{1}{2}\mathbb{P}(S_4). \tag{26}$$

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

$$\mathbb{P}(S_5) \leq \ Succ_{\text{DOOM}}^{n,k,q_{\text{hash}},w}(t_c). \tag{27}$$

(26) together with (27) imply that

$$\mathbb{P}(S_4) \leq 2 \cdot \ Succ^{n,k,q_{\text{hash}},w}_{\text{DOOM}}(t_c).$$

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

7 Hardness of Finding Errors of Large Weight with Prescribed Syndrome

We consider the syndrome decoding problem

Problem 3 (Syndrome Decoding). Given $\mathbf{H} \in \mathbb{F}_q^{(n-k)\times n}$, $\mathbf{s} \in \mathbb{F}_q^{n-k}$, and an integer w, find $\mathbf{e} \in \mathbb{F}_q^n$ such that $\mathbf{eH}^\intercal = \mathbf{s}$ and $|\mathbf{e}| = w$.

This problem is NP-hard [BMvT78], even when the weight w is fixed to any proportion of n [Ste93a]. We discuss here the best solvers with a focus on non binary alphabets and weights close to n. In the sequel, we will denote $\text{SD}(\mathbf{H}, \mathbf{s}, w)$ an instance of this problem.

7.1 Solving Syndrome Decoding for High Weights

In code-based cryptography, Problem 3 is usually considered for "small" values of w. The best known solvers derive from Algorithm 5 [Pra62], often referred to as Information Set Decoding (ISD). As it is described here, it runs repeatedly a polynomial time step, until it outputs $\mathbf{e}$ of

Algorithm 5 PRANGE$(\mathbf{H}, \mathbf{s})$

Parameters (q, n, k, w)

Require: $\mathbf{H} \in \mathbb{F}_q^{(n-k)\times n}$, $\mathbf{s} \in \mathbb{F}_q^{n-k}$
Ensure: $\mathbf{eH}^\intercal = \mathbf{s}$ and $|\mathbf{e}| = w$

repeat
  $\mathcal{I} \leftarrow$ INFOSET$(\mathbf{H})$
  $\mathbf{x} \hookleftarrow \{\mathbf{x} \in \mathbb{F}_q^n \mid |\mathbf{x}_\mathcal{I}| = t\}$ ▷ $t = \min\left(k, \max\left(0, \left[w - \frac{q-1}{q}(n-k)\right]\right)\right)$
  $\mathbf{e} \leftarrow$ PRANGESTEP$(\mathbf{H}, \mathbf{s}, \mathcal{I}, \mathbf{x})$
until $|\mathbf{e}| = w$
return $\mathbf{e}$

The functions INFOSET() *and* PRANGESTEP() *are defined in Algorithm 1.*

weight w. On input $\mathbf{M} \in \mathbb{F}_q^{a\times b}$, the call $\texttt{GaussElim}(\mathbf{M})$ returns an $a \times b$ matrix $\mathbf{M}'$ starting with an $a \times a$ identity block and such that for some non singular matrix $\mathbf{S}$, we have $\mathbf{SM} = \mathbf{M}'$. If $\mathbf{M}'$ doesn't exist, the call fails. The $|$ denotes matrix concatenation.

Dumer's variant of ISD [Dum91] is very similar to Stern's variant [Ste88]. We will limit the description and analysis to instances of Problem 3 with $q \geq 3$ and $w > k + \frac{q-1}{q}(n-k)$. On input $\mathbf{M} \in \mathbb{F}_q^{a\times b}$, the call $\texttt{GaussElimPartial}_\ell(\mathbf{M})$ returns an $a \times b$ matrix $\mathbf{M}'$ whose first $a - \ell$ rows start with an $(a-\ell) \times (a-\ell)$ identity block, whose last ℓ rows start with a $\ell \times (a-\ell)$ zero block, and such that for some non singular matrix $\mathbf{S}$, we have $\mathbf{SM} = \mathbf{M}'$. If $\mathbf{M}'$ doesn't exist, the call fails. The call $\texttt{CollisionSearch}(\mathbf{H}'', \mathbf{s}'', \mathcal{E}_1, \mathcal{E}_2)$ returns the set $\mathcal{E} = \{\mathbf{e} \in \mathcal{E}_1 \times \mathcal{E}_2 \mid \mathbf{eH}''^\intercal = \mathbf{s}''\}$.

7.2 Complexity Analysis

We want to estimate the expected running time of Algorithm 5 and Algorithm 6 when its input $(\mathbf{H}, \mathbf{s})$ is drawn uniformly at random in

$$\mathcal{I}_{q,n,k,w} = \left\{ (\mathbf{H}, \mathbf{s}) \in \mathbb{F}_q^{(n-k)\times n} \times \mathbb{F}_q^{n-k} \,\middle|\, \mathbf{rank}(\mathbf{H}) = n-k, \mathbf{s} \in \{\mathbf{eH}^\intercal \mid \mathbf{e} \in \mathbb{F}_q^n, |\mathbf{e}| = w\} \right\}.$$

Algorithm 6 DUMER$(\mathbf{H}, \mathbf{s})$

Parameters (q, n, k, w), $q \geq 3$, $w \geq k + \frac{q-1}{q}(n-k)$
Additional parameters p, ℓ, and L, tuned for each (q, n, k, w) such that $0 \leq p \leq n-w$, $0 \leq \ell \leq w-k+p$, and $1 \leq L \leq \binom{(k+\ell)/2}{p/2}(q-1)^{(k+\ell-p)/2}$

Require: $\mathbf{H} \in \mathbb{F}_q^{(n-k)\times n}$, $\mathbf{s} \in \mathbb{F}_q^{n-k}$
Ensure: $\mathbf{e}\mathbf{H}^\intercal = \mathbf{s}$ and $|\mathbf{e}| = w$

loop
  $\mathcal{I} \leftarrow$ EXTINFOSET$(\mathbf{H}, \ell)$
  $\mathbf{S}, \mathbf{H}', \mathbf{H}'' \leftarrow$ DECOMP$(\mathbf{H}, \mathcal{I})$
  $(\mathbf{s}', \mathbf{s}'') \leftarrow \mathbf{s}\mathbf{S}^\intercal$
  $\mathcal{E} \leftarrow$ BIRTHDAYSD$(\mathbf{H}'', \mathbf{s}'', k+\ell-p, \mathcal{I}, L)$
  for $\mathbf{e}' \in \mathcal{E}$ **do**
    $\mathbf{e} \leftarrow$ PRANGESTEP$(\mathbf{H}', \mathbf{s}', \mathcal{I}, \mathbf{e}')$
    if $|\mathbf{e}| = w$ **then**
      return $\mathbf{e}$

The functions INFOSET$(\cdot)$ *and* PRANGESTEP$(\cdot)$ *are defined in Algorithm 1.*

function EXTINFOSET$(\mathbf{H}, \ell)$ — extended information set

Require: $\mathbf{H} \in \mathbb{F}_q^{(n-k)\times n}$, $\ell \geq 0$ integer
Ensure: returns $\mathcal{I}$ a set of $k+\ell$ coordinate indices containing an information set of $\langle \mathbf{H} \rangle^\perp$

function DECOMP$(\mathbf{H}, \mathcal{I})$ — shorten and supplement

Require: $\mathbf{H} \in \mathbb{F}_q^{(n-k)\times n}$, $\mathcal{I}$ contains an information set of $\langle \mathbf{H} \rangle^\perp$ and $|\mathcal{I}| = k+\ell$
Ensure: returns $\mathbf{S}, \mathbf{H}', \mathbf{H}'' \in \mathbb{F}_q^{(n-k)\times(n-k)} \times \mathbb{F}_q^{(n-k-\ell)\times n} \times \mathbb{F}_q^{\ell\times n}$ such that $\mathbf{S}$ is non singular, $\mathrm{Supp}(\mathbf{H}'') \subset \mathcal{I}$, and $\begin{pmatrix} \mathbf{H}' \\ \hline \mathbf{H}'' \end{pmatrix} = \mathbf{S}\mathbf{H}$.

The space $\langle \mathbf{H}'' \rangle$ *is a shortened code, it consists of all words of* $\langle \mathbf{H} \rangle$ *whose support is included in* $\mathcal{I}$*. The space* $\langle \mathbf{H}' \rangle$ *supplements* $\langle \mathbf{H}'' \rangle$ *in* $\langle \mathbf{H} \rangle$*. The fact that* $\mathcal{I}$ *is an information set of* $\langle \mathbf{H} \rangle^\perp$ *guaranties the existence of* $\mathbf{S}, \mathbf{H}', \mathbf{H}''$*.*

function BIRTHDAYSD$(\mathbf{H}, \mathbf{s}, w, \mathcal{I}, L)$ — birthday syndrome decoding

Require: $\mathbf{H} \in \mathbb{F}_q^{\ell\times n}$, $\mathbf{s} \in \mathbb{F}_q^{\ell}$, w an integer, $0 \leq w \leq |\mathcal{I}|$, $\mathrm{Supp}(\mathbf{H}) \subset \mathcal{I}$ a set of coordinate indices, L an integer, $1 \leq L \leq \binom{|\mathcal{I}|/2}{w/2}(q-1)^{w/2}$
Ensure: $\mathcal{E} \subset \{\mathbf{e} \in \mathbb{F}_q^n \mid |\mathbf{e}| = w, \mathbf{e}\mathbf{H}^\intercal = \mathbf{s}, \mathrm{Supp}(\mathbf{e}) \subset \mathcal{I}\}$

1: $\mathcal{I}_1 \cup \mathcal{I}_2 \leftarrow \mathcal{I}$ ▷ disjoint union, split $\mathcal{I}$ evenly
2: **for** $i = 1, 2$ **do** $\mathcal{E}_i \leftarrow$ subset of L elements of $\{\mathbf{e} \in \mathbb{F}_q^n \mid |\mathbf{e}| = w/2, \mathrm{Supp}(\mathbf{e}) \subset \mathcal{I}_i\}$
3: $\mathcal{E} \leftarrow \{\mathbf{e} \in \mathcal{E}_1 \times \mathcal{E}_2 \mid \mathbf{e}\mathbf{H}^\intercal = \mathbf{s}\}$
4: **return** $\mathcal{E}$

We do not detail instruction 3*:. We will admit in the analysis that* $\mathcal{E}$ *has cardinality* $L^2 q^{-\ell}$ *on average and can be obtained for a cost* $\max(|\mathcal{E}_1|, |\mathcal{E}_2|, |\mathcal{E}|) = \max(L, L^2 q^{-\ell})$ *up to a polynomial factor.*

This forces the problem to admit a solution and corresponds to the typical situation in cryptanalysis.

Proposition 5 (Complexity of Prange's Algorithm). *We consider Algorithm 5 with parameters* (q, n, k, w) *and an input* $(\mathbf{H}, \mathbf{s})$ *drawn uniformly at random in* $\mathcal{I}_{q,n,k,w}$*. Up to a polynomial factor, the algorithm has an expected running time equal to* $\mathrm{WF}_0(q, n, k, w) = 1/P_0$ *where*

$$P_0 = \frac{\binom{n-k}{w-t}(q-1)^{w-t}}{\min\left(q^{n-k}, \binom{n}{w}(q-1)^w\right)} \text{ with } t = \begin{cases} 0 & \text{if } w \leq \frac{q-1}{q}(n-k) \\ k & \text{if } w \geq k + \frac{q-1}{q}(n-k) \\ w - \frac{q-1}{q}(n-k) & \text{else} \end{cases}, \tag{28}$$

is the probability, up to a constant factor, that the instruction 5*: produces a word of weight* w*.*

Lemma 5. *For any $\mathcal{E} \subset \mathbb{F}_q^n$, we have, on average over all $\mathbf{H} \in \mathbb{F}_q^{(n-k)\times n}$*

$$\frac{1}{2} \leq \frac{|\{\mathbf{e}\mathbf{H}^\intercal \mid \mathbf{e} \in \mathcal{E}\}|}{\min\left(q^{n-k}, |\mathcal{E}|\right)} \leq 1.$$

Proof (Proof of Proposition 5). We use the notation of Algorithm 5. The proof is similar to the proof of Proposition 2. The main difference is that we choose $(\mathbf{H}, \mathbf{s})$ in $\mathcal{I}_{q,n,k,w}$ instead of $\mathbb{F}_q^{(n-k)\times n} \times \mathbb{F}_q^{n-k}$. Choosing $\mathbf{H}$ of full rank has a negligible impact on the probabilities. The choice of $\mathbf{s}$ has an impact as we choose it in the set $\{\mathbf{e}\mathbf{H}^\intercal \mid \mathbf{e} \in \mathbb{F}_q^n, |\mathbf{e}| = w\}$ which might differ significantly from $\mathbb{F}_q^{n-k}$. From the above lemma, its size is equal to $\min\left(q^{n-k}, \binom{n}{w}(q-1)^w\right)$ up to a factor at most 2. In formula (11) of Proposition 2, we replace the denominator q^{n-k} by $\min\left(q^{n-k}, \binom{n}{w}(q-1)^w\right)$. Also, in Algorithm 5 the weight t has a fixed value depending on the parameters q, n, k, w and finally the probability to reach $|\mathbf{e}| = w$ is, from (11), equal to

$$P_0 = \mathbb{P}\left(|\mathbf{e}| = w\right) = \frac{\binom{n-k}{w-t}(q-1)^{w-t}}{\min\left(q^{n-k}, \binom{n}{w}(q-1)^w\right)}$$

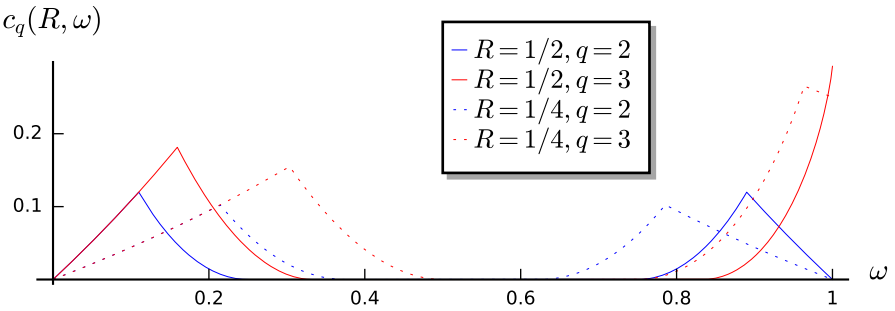
up to a small constant. We will thus iterate on average $1/P_0$ times an elementary step of polynomial cost, dominated by the linear algebra in PrangeStep$(\cdot)$. This concludes the proof.

When $t = w - \frac{q-1}{q}(n-k)$ in (28) the probability P_0 is proportional to $1/\sqrt{n-k}$ in all other cases it is exponentially small in n. For a fixed code rate $R = k/n$ and error rate $\omega = w/n$, the following limit is defined and is called the asymptotic exponent

$$c_q(R, \omega) = \lim_{n\to\infty} \frac{1}{n} \log_2 \mathrm{WF}_0(q, n, Rn, \omega n).$$

We give this exponent for $R = 1/2$ and $q \in \{2, 3\}$ in Figure 9. The exponent is null for middle values of ω, this means that the corresponding error weights can be obtained in polynomial time in n. We remark that for $q = 2$ the problem is symmetric with respect to the error weight w. In that case, finding a word of weight w with prescribed syndrome is *exactly* as hard as finding a word of weight $n-w$ with prescribed syndrome. Indeed, for $q = 2$, solving the instance $(\mathbf{H}, \mathbf{s}, w)$ of Problem 3 is the same thing as solving the instance $(\mathbf{H}, \mathbf{s}+\mathbf{1}\mathbf{H}^\intercal, n-w)$. For larger q, the symmetry disappears. The singular points correspond to the solutions of $\binom{n}{w}(q-1)^w = q^{n-k}$. On left-hand side it is the Gilbert-Varshamov distance. On the right hand side it exists only if $\frac{n-k}{n} > \frac{\log(q-1)}{\log q}$. For instance for $q = 3$, only if $k/n < 1 - 1/\log_2 3 \approx 0.37$.

Fig. 9. Asymptotic exponent of Prange's algorithm for code rate $R \in \{1/4, 1/2\}$ and $q \in \{2, 3\}$



In the sequel, *low weight* will refer to the left-hand side of Figure 9, with $w \leq \frac{q-1}{q}(n-k)$ which corresponds to the traditional syndrome decoding parameters while *high weight* will refer to the

right-hand side of Figure 9, with $w > k + \frac{q-1}{q}(n-k)$, corresponding to the parameters of interest in this work.

Proposition 6 (Complexity of Dumer Algorithm). *We consider Algorithm 6 with parameters (q, n, k, w, p, ℓ, L) and an input $(\mathbf{H}, \mathbf{s})$ drawn uniformly at random in $\mathcal{I}_{q,n,k,w}$. Up to a polynomial factor, the algorithm has an expected running time equal to*

$$\mathrm{WF}_{p,\ell,L}(q,n,k,w) = \max\left(1, \frac{q^\ell}{P_{p,\ell}L^2}\right)\max\left(L, \frac{L^2}{q^\ell}\right), \tag{29}$$

where

$$P_{p,\ell} = \frac{q^\ell \binom{n-k-\ell}{w-k-\ell+p}(q-1)^{w-k-\ell+p}}{\min\left(q^{n-k}, \binom{n}{w}(q-1)^w\right)}. \tag{30}$$

is the probability, up to a constant factor, that the instruction 7: *produces a word of weight w.*

Proof. Any $\mathbf{e}' \in \mathcal{E}$ has weight $t = k + \ell - p$ and it is completed by PrangeStep$(\cdot)$ with $n-k-\ell$ random coordinates. We want to estimate the probability that this completion give a word of weight w. We apply Proposition 2, $\mathbf{H}' \in \mathbb{F}_q^{(n-k-\ell)\times n}$ and $\mathbf{s}' \in \mathbb{F}_q^{n-k-\ell}$ with a distribution of $|\mathbf{e}'|$ which is concentrated on $t = k+\ell-p$. The denominator is the number of possible input syndrome $\mathbf{s}$ divided by q^ℓ because $\mathbf{s}'' \in \mathbb{F}_q^\ell$ is fixed, that is $q^{-\ell}\min\left(q^{n-k}, \binom{n}{w}(q-1)^w\right)$. We obtain

$$P_{p,\ell} = \frac{q^\ell \binom{n-k-\ell}{w-t}(q-1)^{w-t}}{\min\left(q^{n-k}, \binom{n}{w}(q-1)^w\right)} \text{ with } t = k-\ell+p.$$

The instruction 7: will be executed $1/P_{p,\ell}$ times on average. We have $|\mathcal{E}| = L^2 q^{-\ell}$ on average, thus the main loop is executed $1/(P_{p,\ell}|\mathcal{E}|)$ times, rounded up. Instructions 2: to 5: will cost $\max(L, |\mathcal{E}|)$ up to a polynomial factor. In total, up to a polynomial factor, we thus have (29).

In the sequel, we will denote

$$\mathrm{WF}_{\mathrm{Dumer}}(q,n,k,w) = \min_{(p,\ell,L)\in\mathcal{P}} \mathrm{WF}_{p,\ell,L}(q,n,k,w)$$

where $\mathcal{P} = \left\{(p,\ell,L) \mid 0 \le p \le n-w, 0 \le \ell \le w-k+p, 1 \le L \le \binom{k+\ell}{p}(q-1)^{(k+\ell-p)}\right\}$ is the set of admissible optimization parameters.

Corollary 1. *For any (q,n,k,w), let ℓ_0 denote the solution of $P_{0,\ell_0} = q^{-\ell_0}$. We have*

$$\max\left(1, \frac{q^{\ell_0}}{(q-1)^{(k+\ell_0)/2}}\right) \ge \frac{\mathrm{WF}_{\mathrm{Dumer}}(q,n,k,w)}{q^{\ell_0}} \ge 1 \tag{31}$$

up to a polynomial factor.

Proof. (Sketch) We will minimize the formula (29) over $\mathcal{P}' = \{(p,\ell,L) \mid 0 \le p \le n-w, 0 \le \ell \le w-k+p, L \ge 1\}$, that is we ignore the upper bound for L. Since $\mathcal{P} \subset \mathcal{P}'$, this will give us a lower bound for the work factor (whose tightness is discussed later).

When $q > 2$ and $w \ge k + \frac{q-1}{q}(n-k)$, and L is not upper bounded, an easy analysis shows the following, up to a polynomial factor.

- It is best to choose $1 = \frac{q^\ell}{P_{p,\ell}L^2}$ and $L = \frac{L^2}{q^\ell}$. Thus parameters such that $L = q^\ell = P_{p,\ell}^{-1}$ are optimal.
- The function $p \mapsto P_{p,\ell}$ is decreasing with p in the neighbourhood of the optimal parameters. It follows that $p = 0$ is optimal.
- The equation $q^\ell P_{0,\ell} = 1$ has a single solution $\ell_0 \in [0, w-k]$.

It follows that $(p,\ell,L) = (0,\ell_0,q^{\ell_0})$ minimizes (29) over $\mathcal{P}'$, and the right-hand side inequality of the statement holds. Let $L_0 = \min(q^{\ell_0}, (q-1)^{(k+\ell_0)/2})$, we have $(0,\ell_0,L_0) \in \mathcal{P}$. The evaluation of the work factor for this set of parameter provides the left-hand side upper bound.

Tightness of Corollary 1. Every time $q^{\ell_0} \leq (q-1)^{(k+\ell_0)/2}$ the bound (31) is tight. For high weights and non-binary alphabet, this happens for many system parameters of interest. It corresponds to situations where $(q-1)^{(k+\ell)/2}$ is large enough, and allows the choice of a list size L large enough to succeed in a constant number of iterations even with $p = 0$.

In contrast, in the low weight case, the upper bound for L is $\binom{k+\ell}{p}(q-1)^p$ and $p = 0$ correspond to an irrelevant degenerated case where $L = 1$.

The bound of Corollary 1 is tight when

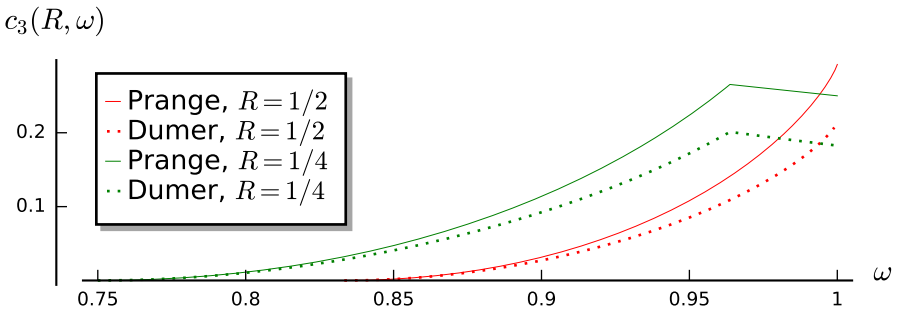
- $k/n \geq 1 - \log_q(q-1)$ and all w
- $k/n < 1 - \log_q(q-1)$ and $w \leq k + (n-k)\omega - (1-\lambda)\omega$ where $\lambda = \frac{\log(q-1)}{2\log q - \log(q-1)}$ and ω is the solution[3] of $H_q(\omega) = \frac{n-(1+2\lambda)k}{n-(1+\lambda)k}$ larger than $\frac{q-1}{q}$. Note that ω exists only if $k < 1 - \log_q(q-1)$.

For $q = 3$, the bound is tight when $k/n > 0.369$. In the current work we will never use smaller code rates. As for Prange's algorithm, we may define the asymptotic exponent for a given $R = k/n$ and $\omega = w/n$ as

$$c_q(R, \omega) = \lim_{n\to\infty} \frac{1}{n} \log_2 \mathrm{WF}_{\mathrm{Dumer}}(q, n, Rn, \omega n).$$

Exponents are plotted for Prange and Dumer (bound) for $q = 3$, $R \in \{1/2, 1/4\}$, and large values of ω.

Fig. 10. Asymptotic exponent of Dumer (dotted) *vs.* Prange for $q = 3$



Further Improvement The Algorithm 6 for high error weights and $q > 2$ has an interesting feature when the code rate is large enough ($k/n \geq 1 - \log_q(q-1)$, see Corollary 1). The optimal value for parameter p is zero and the optimal value of the list size L is below its upper bound $(q-1)^{(k+\ell)/2}$. It corresponds to a situation where the algorithm requires a single or a few iterations, and increasing L would simply increase the cost of each iteration without the benefit of reducing the number of iterations.

For low weights, Dumer's algorithm performs an exponential number of loop iterations and the list size L meets its upper bounded $\sqrt{\binom{k+\ell}{p}(q-1)^p}$. The best improvements [MMT11, BJMM12] are using the representation technique, they essentially improve the `CollisionSearch` to allow a larger upper bound for L. Those algorithms perform a smaller, but still exponential, number of iterations, each having a larger list size and thus an improved "birthday effect". This won't happen in the high weight situation when $k/n \geq 1 - \log_q(q-1)$ ($k/n \geq 0.369$ for $q = 3$). The nearest neighbour approach [MO15] might still allow some improvement but needs to be thoroughly revisited.

[3] $H_q(x) = -x\log_q(x/(q-1)) - (1-x)\log_q(1-x)$ is the q-ary entropy.

Previous Works In the binary case, we have seen earlier that the syndrome decoding problem is equally hard for low and high weights. For larger alphabets there is no obvious reduction one way or the other between low and high weights problems and the high weight syndrome decoding does not seem easier than its low weight counterpart.

Except for the "Furthest neighbour" algorithm mentioned in [Ind03], we are not aware in the literature of any algorithm related to decoding far from the received word.

Indeed, more research is needed to improve the understanding of high weight syndrome decoding.

7.3 Application to the Proposed Signature

In the current work, we will focus on the case $q = 3$ with high weight $w \geq k + \frac{q-1}{q}(n-k)$, and a large enough code rate $k/n \geq 1 - \log_q(q-1) = 0.369$. This corresponds to a situation where Problem 3 has an exponentially large number of solutions and the best known solver is Dumer's algorithm with $p = 0$ and a constant number of iterations.

DOOM *vs.* Collisions

Problem 4 (Syndrome Collision). Given $\mathbf{H} \in \mathbb{F}_q^{(n-k)\times n}$ and an integer w, find $\mathbf{e}, \mathbf{e}'$ distinct in $\mathbb{F}_q^n$ such that $\mathbf{e}\mathbf{H}^\intercal = \mathbf{e}'\mathbf{H}^\intercal$ and $|\mathbf{e}| = |\mathbf{e}'| = w$.

When $q \geq 3$ the above problem is not harder than finding a non-zero word $\mathbf{x} \in \mathbb{F}_q^n$ of weight $\leq 2w$ such that $\mathbf{x}\mathbf{H}^\intercal = 0$. Indeed, given $\mathbf{x} \neq 0$ it is a simple matter to find $\mathbf{e}, \mathbf{e}'$ such that $\mathbf{e} - \mathbf{e}' = \mathbf{x}$ and $|\mathbf{e}| = |\mathbf{e}'| = w$ for any $w \geq |\mathbf{x}|/2$. It follows that for high weights and a non binary alphabet the Syndrome Collision problem is always easy.

The key primitive in our design is the syndrome $\mathbf{e} \mapsto \mathbf{e}\mathbf{H}^\intercal$ with $|\mathbf{e}| = w$. In the GPV setting [GPV08], one of the feature needed for designing a secure signature scheme is collision-freeness. From the above remark, this cannot be achieved for syndromes with high w. Instead, our reduction will require the primitive to be resistant to multi-target preimage, namely the Decoding One Out of Many (DOOM) problem.

Problem 2. (DOOM – Decoding One Out of Many). Given $\mathbf{H} \in \mathbb{F}_q^{(n-k)\times n}$, $\mathbf{s}_1, \cdots, \mathbf{s}_N \in \mathbb{F}_q^{n-k}$, and an integer w, find $\mathbf{e} \in \mathbb{F}_q^n$ and $i, 1 \leq i \leq N$ such that such that $\mathbf{e}\mathbf{H}^\intercal = \mathbf{s}_i$ and $|\mathbf{e}| = w$.

In DOOM, the adversary considers an arbitrary large number of instances of syndrome decoding in which only syndrome $\mathbf{s}$ varies, and needs to solve only one of them.

What the DOOM variant of ISD [Sen11] does to deal with N instances consists essentially in (i) multiplying the cost of the iteration, in fact the list size by $\sqrt{N}$, and (ii) dividing the number of iteration by N. The number iterations cannot be lower than 1, and this limits in practice the number of instances that can be efficiently treated simultaneously. In our case, Dumer's algorithm is already reduced to a single iteration and no improvement is possible with this method. With the current state-of-the-art the DOOM problem for high weights does not appear easier to solve than the Syndrome Decoding problem.

8 Distinguishing a Permuted Admissible Generalized $(U, U+V)$ Code

A permissible generalized $(U, U+V)$ code where U and V are random seems very close to a random linear code. We assume in the whole section that such a code is defined from a 4-tuple of matrices that we denote by $(\mathbf{D}_1, \mathbf{D}_2, \mathbf{D}_3, \mathbf{D}_4)$. 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 admissible generalized $(U, U+V)$-code of the same length and dimension. This slight difference happens for small and large weights and is due to codewords where $\mathbf{v} = \mathbf{0}$ or $\mathbf{u} = \mathbf{0}$ which are of the form $(\mathbf{u}\mathbf{D}_1, \mathbf{u}\mathbf{D}_3)$ where $\mathbf{u}$ belongs to U or codewords of the form $(\mathbf{v}\mathbf{D}_2, \mathbf{v}\mathbf{D}_4)$ where $\mathbf{v}$ belongs to V. This weight distribution will depend on the matrices $\mathbf{D}_i$'s. The definition of number of V blocks (see Definition 3 in §5.1) is helpful for describing the codewords of the form $(\mathbf{v}\mathbf{D}_2, \mathbf{v}\mathbf{D}_4)$. With this definition at hand, we have the following proposition

Proposition 7. *Assume that we choose an admissible generalized $(U, U+V)$ code over $\mathbb{F}_3$ with a number n_I of linear combinations of type I by picking the parity-check matrices of U and V uniformly at random among the ternary matrices of size $(n/2-k_U)\times n/2$ and $(n/2-k_V)\times n/2$ respectively. Let $a_{(\mathbf{u},\mathbf{v})}(w)$, $a_{(\mathbf{u},\mathbf{0})}(w)$ and $a_{(\mathbf{0},\mathbf{v})}(w)$ be the expected number of codewords of weight w that are respectively in the admissible generalized $(U, U+V)$ code, of the form $(\mathbf{uD}_1, \mathbf{uD}_3)$ where $\mathbf{u}$ belongs to U and of the form $(\mathbf{vD}_2, \mathbf{vD}_4)$ where $\mathbf{v}$ belongs to V. These numbers are given for even w in $\{0,\dots,n\}$ by*

$$a_{(\mathbf{u},\mathbf{0})}(w) = \frac{\binom{n/2}{w/2}2^{w/2}}{3^{n/2-k_U}} \quad ; \quad a_{(\mathbf{0},\mathbf{v})}(w) = \frac{1}{3^{n/2-k_V}} \sum_{\substack{j=0 \\ j \text{ even}}}^{w} \binom{n_I}{j}\binom{n/2-n_I}{\frac{w-j}{2}}2^{(w+j)/2}$$

$$a_{(\mathbf{u},\mathbf{v})}(w) = a_{(\mathbf{u},\mathbf{0})}(w) + a_{(\mathbf{0},\mathbf{v})}(w) + \frac{1}{3^{n-k_U-k_V}}\left(\binom{n}{w}2^w - \binom{n/2}{w/2}2^{w/2} - \sum_{\substack{j=0 \\ j \text{ even}}}^{w} \binom{n_I}{j}\binom{n/2-n_I}{\frac{w-j}{2}}2^{(w+j)/2}\right)$$

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

$$a_{(\mathbf{u},\mathbf{0})}(w) = 0 \quad ; \quad a_{(\mathbf{0},\mathbf{v})}(w) = \frac{1}{3^{n/2-k_V}} \sum_{\substack{j=0 \\ j \text{ odd}}}^{w} \binom{n_I}{j}\binom{n/2-n_I}{\frac{w-j}{2}}2^{(w+j)/2}$$

$$a_{(\mathbf{u},\mathbf{v})}(w) = a_{(\mathbf{0},\mathbf{v})}(w) + \frac{1}{3^{n-k_U-k_V}}\left(\binom{n}{w}2^w - \sum_{\substack{j=0 \\ j \text{ odd}}}^{w} \binom{n_I}{j}\binom{n/2-n_I}{\frac{w-j}{2}}2^{(w+j)/2}\right)$$

On the other hand, when we choose a linear code of length n over $\mathbb{F}_3$ 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

$$a(w) = \frac{\binom{n}{w}2^w}{3^{n-k_U-k_V}}.$$

The proof of this proposition is in Appendix §C

Remark 6. When the generalized $(U, U+V)$ code is chosen in this way, its dimension is $k_U + k_V$ with probability $1 - O\left(\max(3^{k_U-n/2}, 3^{k_V-n/2})\right)$. This also holds for the random codes of length n.

We have plotted in Figure 11 the normalized logarithm of the density of codewords of the form $(\mathbf{uD}_1, \mathbf{uD}_3)$ and $(\mathbf{vD}_2\,\mathbf{vD}_4)$ of relative *even* weight $x \stackrel{\triangle}{=} \frac{w}{n}$ against x in the case where U is of rate $\frac{k_U}{n/2} = 0.7$, V is of rate $\frac{k_V}{n/2} = 0.3$ and $\frac{n_I}{n/2} = \frac{1}{2}$. These two relative densities are defined respectively by

$$\alpha_{\mathbf{u}}(w/n) \stackrel{\triangle}{=} \frac{\log_2(a_{(\mathbf{u},\mathbf{0})}(w)/a_{(\mathbf{u},\mathbf{v})}(w))}{n} \quad ; \quad \alpha_{\mathbf{v}}(w/n) \stackrel{\triangle}{=} \frac{\log_2(a_{(\mathbf{0},\mathbf{v})}(w)/a_{(\mathbf{u},\mathbf{v})}(w))}{n}$$

We see that for a relative weight w/n below approximately 0.26 almost all the codewords are of the form $(\mathbf{uD}_1, \mathbf{uD}_3)$ in this case.

Since the weight distribution is invariant by permuting the positions, this slight difference also survives in the permuted version of the admissible generalized $(U, U+V)$ code. These considerations lead to the best attack we have found for recovering the structure of a permuted admissible

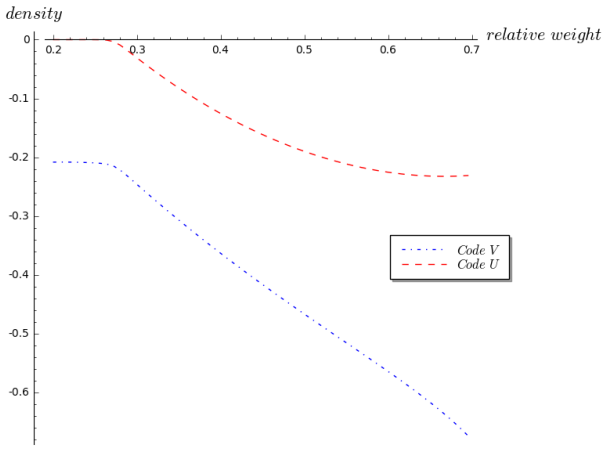


Fig. 11. $\alpha_{\mathbf{u}}(w/n)$ and $\alpha_{\mathbf{v}}(w/n)$ against $x \stackrel{\triangle}{=} \frac{w}{n}$.

generalized $(U, U+V)$ 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 $(\mathbf{u}\mathbf{D}_1, \mathbf{u}\mathbf{D}_3)$ codeword where $\mathbf{u}$ is in U or a permuted $(\mathbf{v}\mathbf{D}_2, \mathbf{v}\mathbf{D}_4)$ codeword where $\mathbf{v}$ belongs to V. The rationale behind this algorithm is that the density of codewords of the form $(\mathbf{u}\mathbf{D}_1, \mathbf{u}\mathbf{D}_3)$ or $(\mathbf{v}\mathbf{D}_2, \mathbf{v}\mathbf{D}_4)$ 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.

8.1 Recovering the U Code up to Permutation

We consider here the permuted code

$$U' \stackrel{\triangle}{=} (U\mathbf{D}_1, U\mathbf{D}_3)\mathbf{P} = \{(\mathbf{u}\mathbf{D}_1, \mathbf{u}\mathbf{D}_3)\mathbf{P} : \mathbf{u} \in U\}.$$

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 either always equal or always sum to 0 in U'. The basic algorithm for recovering the code U' is given in Algorithm 7.

It uses other auxiliary functions

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

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

Algorithm 7 ComputeU: algorithm that computes a set of independent elements in U'.

Parameters: (i) ℓ : small integer (typically $\ell \leqslant 40$),
(ii) p : very small integer (typically $1 \leqslant p \leqslant 10$).
Input: (i) $\mathcal{C}_{\text{pk}}$ the public code used for verifying signatures.
(ii) N a certain number of iterations
Output: an independent set of elements in U'

1: **function** ComputeU($\mathcal{C}_{\text{pk}}$,N)
2: **for** $i = 1, \ldots, N$ **do**
3: $B \leftarrow \emptyset$
4: Choose a set $I \subset \{1, \ldots, n\}$ of size $n - k - \ell$ uniformly at random
5: $\mathcal{L} \leftarrow$ Codewords(Punc$_I(\mathcal{C}_{\text{pk}}), p)$
6: **for all** $\mathbf{x} \in \mathcal{L}$ **do**
7: $\mathbf{x} \leftarrow$ Complete$(\mathbf{x}, I, \mathcal{C}_{\text{pk}})$
8: **if** CheckU$(\mathbf{x})$ **then**
9: add $\mathbf{x}$ to B if $\mathbf{x} \notin < B >$
10: **return** B

Proposition 8. *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*

$$P_{succ} \geq \sum_{w=0}^{n/2} \frac{\binom{n/2}{w}\binom{n/2-w}{k+l-2w}2^{k+l-2w}}{\binom{n}{k+l}} f\left(\frac{\binom{k+\ell-2w}{p-2i}\binom{w}{i}2^{p-i}}{3^{\max(0,k+\ell-w-k_U)}}\right) \tag{32}$$

where f is the function defined by $f(x) \stackrel{\triangle}{=} \max\left(x(1-x/2), 1-\frac{1}{x}\right)$. Algorithm 7 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 6. *Choose a random code $\mathcal{C}_{rand}$ of length n from a parity-check matrix of size $r \times n$ chosen uniformly at random in $\mathbb{F}_3^{r \times n}$. Let X be some subset of $\mathbb{F}_3^n$ of size m. We have*

$$\mathbb{P}(X \cap \mathcal{C}_{rand} \neq \emptyset) \geq f\left(\frac{m}{3^r}\right).$$

We say that two positions i and j are matched (for U') if and only if there exists $\lambda \in \{\pm 1\}$ such that $c_i = \lambda c_j$ for every $\mathbf{c} \in U'$. From the fact that we only consider admissible generalized $(U, U+V)$-codes, there are clearly $n/2$ pairs of matched positions. W will now be defined by the number of matched pairs that are included in $\{1, \ldots, n\} \setminus I$ where I is the random set of size $n-k-l$ which is drawn in Instruction 4 of Algorithm 7. We compute the probability of success by conditioning on the values taken by W:

$$P_{\text{succ}} = \sum_{w=0}^{n/2} \mathbb{P}(W = w)\mathbb{P}\left(\exists \mathbf{x} \in U' : |\mathbf{x}_{\bar{I}}| = p \mid W = w\right) \tag{33}$$

where $\bar{I} \stackrel{\triangle}{=} \{1, \ldots, n\} \setminus I$. 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 $n-(n-k-\ell+w)=k+\ell-w$ as $|J_{22}|=w$ and $|I|=n-k-l$. 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 6 to it due to the matched positions it contains (the code is not random). 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 6 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 $\{0,\ldots,\lfloor p/2\rfloor\}$ the sets X_i defined as follows

$$X_i \stackrel{\triangle}{=}\{\mathbf{x}=(x_i)_{i\in\bar{I}\setminus J_{22}}\in\mathbb{F}_3^{k+\ell-w} : |\mathbf{x}_{J_1}|=p-2i,\ |\mathbf{x}_{J_{21}}|=i\}$$

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

$$\mathbb{P}\left\{\exists \mathbf{x}\in U' : |\mathbf{x}_{\bar{I}}|=p \mid W=w\right\} \geq \max_{i=0}^{\lfloor p/2\rfloor}\left\{\mathbb{P}(X_i\cap U'''\neq\emptyset)\right\} \tag{34}$$

By using Lemma 6 we have

$$\mathbb{P}(X_i\cap U'''\neq\emptyset)\geq f\left(\frac{\binom{k+\ell-2w}{p-2i}\binom{w}{i}2^{p-i}}{3^{\max(0,k+\ell-w-k_U)}}\right). \tag{35}$$

On the other hand, we may notice that

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

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

$$P_{\mathrm{succ}}\geq\sum_{w=0}^{n/2}\frac{\binom{n/2}{w}\binom{n/2-w}{k+l-2w}2^{k+l-2w}}{\binom{n}{k+l}}f\left(\frac{\binom{k+\ell-2w}{p-2i}\binom{w}{i}2^{p-i}}{3^{\max(0,k+\ell-w-k_U)}}\right)$$

Complexity of Recovering a Permuted Version of U. The complexity of a call to ComputeU can be estimated as follows. We denote the complexity of computing the list of codewords of weight p in a code of length $k+\ell$ and dimension k by $C_1(p,k,\ell)$. It depends on the particular algorithm used here. For more details see [Dum91, FS09, MMT11, BJMM12, MO15]. This is the complexity of the call $\textsc{Codewords}(\mathrm{Punc}_I(\mathcal{C}_{\mathrm{pk}}),p)$ in Step 5 in Algorithm 7. The complexity of ComputeU and hence the complexity of recovering a permuted version of U is clearly lower bounded by $\Omega\left(\frac{C_1(p,k,\ell)}{P_{\mathrm{succ}}}\right)$. It turns out that the whole complexity of recovering a permuted version of U is actually of this order, namely $\Theta\left(\frac{C_1(p,k,\ell)}{P_{\mathrm{succ}}}\right)$. This can be done by a combination of two techniques

- Once a non-zero element of U' 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 ComputeU, 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 CheckU 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_U. 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 U', 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 CheckU.

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 U.

8.2 Recovering the V Code up to a Permutation

We consider here the permuted code

$$V' \stackrel{\triangle}{=} (V\mathbf{D}_2, V\mathbf{D}_4)\mathbf{P} = \{(\mathbf{v}\mathbf{D}_2, \mathbf{v}\mathbf{D}_4)\mathbf{P} \text{ where } \mathbf{v} \in V\}.$$

The attack in this case consists in recovering a basis of V'. Once this is achieved, the support $\mathrm{Supp}(V')$ of V' can easily 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 easily recover the code V up to some permutation. The algorithm for recovering V' is the same as the algorithm for recovering U'. We call the associated function ComputeV 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 analyse how we have to choose N in order that ComputeV returns $\Omega(1)$ elements of V'. We have in this case the following result.

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

$$P_{succ} \geq \sum_{w=0}^{\min(n-k-l,n-n_I)} \sum_{m=0}^{n/2-n_I} \frac{\binom{\frac{n}{2}-n_I}{m}\binom{n_I}{n-k-l-w}}{\binom{n}{n-k-l}} \max_{i=0}^{\lfloor p/2 \rfloor} f\left(\frac{\binom{n-n_I-w-2m}{p-2i}\binom{m}{i}2^{p-i}}{3^{\max(0,n-n_I-w-m-k_V)}}\right) \sum_{j=0}^{n/2-n_I-m} \binom{n/2-n_I-m}{j} 2^j \binom{n_I}{w-n+2n_I+2m+j}$$

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

Proof. To lower-bound the probability P_{succ} that an iteration is successful, let us first introduce the concept of *matched positions* (for V'). We say that two positions i and j are matched if and only if there exists $\lambda \in \{\pm 1\}$ such that $c_i = \lambda c_j$ for every $\mathbf{c} \in V'$. There are clearly $\frac{n}{2} - n_I$ pairs of matched positions. Let us define the following set: J is the set of positions that are of the images of the permutation $\mathbf{P}$ of the positions $1 \leq i \leq n/2$ such that $\mathbf{D}_2(i,i) \neq 0$ and the images of positions $n/2 + j$ with $0 \leq j \leq n/2$ such that $\mathbf{D}_4(j,j) \neq 0$.

Remark 7. From Definition 3 it follows that

$$|J| = n - n_I$$

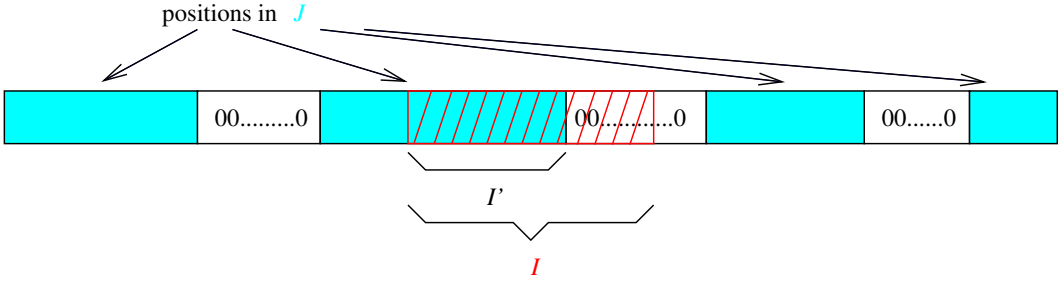
(see Remark 3 in §5.1).

Let us now bring in the following random variables

$$I' \stackrel{\triangle}{=} I \cap J \quad \text{and} \quad W \stackrel{\triangle}{=} |I'|$$

and M be the number of matched pairs which are included in $J \setminus I'$. $J \setminus I'$ represents the set of positions that are not necessarily equal to 0 in the punctured code $\mathrm{Punc}_I(V')$ (see Figure 12).

Fig. 12. A figure representing J, I and I' and the form of a codeword in V'.



ComputeV outputs at least one element of V' if there is an element of weight p in $\mathrm{Punc}_{I'}(V')$. Therefore the probability of success P_{succ} is given by

$$P_{\mathrm{succ}} = \sum_{w=0}^{\min(n-k-l,n-n_I)} \sum_{m=0}^{n/2-n_I} \mathbb{P}\left(\exists \mathbf{x} \in V' : |\mathbf{x}_{J'}| = p \mid W = w, M = m\right) \mathbb{P}(W = w, M = m) \quad (36)$$

where

$$J' \stackrel{\triangle}{=} J \setminus I'.$$

Notice that we can partition J' as $J' = J_1 \cup J_2$ where J_2 consists in the union of the matched pairs in J'. Note that $|J_2| = 2m$. We may further partition J_2 as $J_2 = J_{21} \cup J_{22}$ where the elements of a matched pair are divided in two sets. In other words, neither J_{21} nor J_{22} contains a matched pair. We are going to consider the following codes

$$V'' \stackrel{\triangle}{=} \mathrm{Punc}_{I \cup \bar{J}}(V')$$
$$V''' \stackrel{\triangle}{=} \mathrm{Punc}_{I \cup \bar{J} \cup J_{22}}(V').$$

V'' is of length $n - n_I - w$, whereas the last code is of length $n - n_I - w - m$. The point of defining the first code is that

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

is equal to the probability that V'' contains a codeword of weight p. The problem is that we can not apply Lemma 6 to it due to the matched positions it contains. This is precisely the point of defining V'''. 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 $\max(0, n - n_I - w - m - k_V) \times (n_V - w - m)$. We can therefore apply Lemma 6 to it. We have to be careful about the words of weight p in V'' though, since they do not have the same probability of occurring in V'' due to the possible presence of matched pairs in the support. This is why we introduce for i in $\{0, \ldots, \lfloor p/2 \rfloor\}$ the sets X_i defined as follows

$$X_i \stackrel{\triangle}{=} \{\mathbf{x} = (x_i)_{i \in J' \setminus J_{22}} \in \mathbb{F}_3^{n-n_I-w-m} : |\mathbf{x}_{J_1}| = p - 2i,\ |\mathbf{x}_{J_{21}}| = i\}$$

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

$$\mathbb{P}\left\{\exists \mathbf{x} \in V' : |\mathbf{x}_{\bar{I}}| = p \mid W = w, M = m\right\} \geq \max_{i=0}^{\lfloor p/2 \rfloor} \left\{\mathbb{P}(X_i \cap V''' \neq \emptyset)\right\} \quad (37)$$

By using Lemma 6 we have

$$\mathbb{P}(X_i \cap V''' \neq \emptyset) \geq f\left(\frac{\binom{n-n_I-w-2m}{p-2i}\binom{m}{i}2^{p-i}}{3^{\max(0,n-n_I-w-m-k_V)}}\right). \tag{38}$$

On the other hand, we have

$$\mathbb{P}(W=w, M=m) = \frac{\binom{\frac{n}{2}-n_I}{m}\binom{n_I}{n-k-l-w}}{\binom{n}{n-k-l}} \sum_{j=0}^{n/2-n_I-m} \binom{n/2-n_I-m}{j} 2^j \binom{n_I}{w-n+2n_I+2m+j}$$

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

$$P_{\text{succ}} \geq \sum_{w=0}^{\min(n-k-l,n-n_I)} \sum_{m=0}^{n/2-n_I} \frac{\binom{\frac{n}{2}-n_I}{m}\binom{n-n-n_I}{n-k-l-w}}{\binom{n}{n-k-l}} \max_{i=0}^{\lfloor p/2 \rfloor} f\left(\frac{\binom{n-n_I-w-2m}{p-2i}\binom{m}{i}2^{p-i}}{3^{\max(0,n-n_I-w-m-k_V)}}\right) \sum_{j=0}^{n/2-n_I-m} \binom{n/2-n_I-m}{j} 2^j \binom{n_I}{w-n+2n_I+2m+j}$$

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

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

8.3 Distinguishing a Generalized $(U, U+V)$ Code

It is not clear in the second 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 generalized $(U, U+V)$-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 the following proposition

Proposition 10. *The aforementioned algorithms lead to a distinguishing attack whose complexity is given by* $\min\left(O\left(\min_{p,l} C_U(p,l)\right), O\left(\min_{p,l} C_U(p,l)\right)\right)$

$$C_U(p,l) \stackrel{\triangle}{=} \frac{C_1(p,k,\ell)}{\sum_{w=0}^{n/2} \frac{\binom{n/2}{w}\binom{n/2-w}{k+l-2w}2^{k+l-2w}}{\binom{n}{k+l}} \max_{i=0}^{\lfloor p/2 \rfloor} f\left(\frac{\binom{k+\ell-2w}{p-2i}\binom{w}{i}2^{p-i}}{3^{\max(0,k+\ell-w-k_U)}}\right)} \tag{39}$$

$$C_V(p,l) \stackrel{\triangle}{=} \frac{C_1(p,k,\ell)}{\sum_{\mathcal{I}} \frac{\binom{\frac{n}{2}-n_I}{m}\binom{n_I}{n-k-l-w}}{\binom{n}{n-k-l}} \max_{i=0}^{\lfloor p/2 \rfloor} f\left(\frac{\binom{n-n_I-w-2m}{p-2i}\binom{m}{i}2^{p-i}}{3^{\max(0,n-n_I-w-m-k_V)}}\right) \binom{n/2-n_I-m}{j} 2^j \binom{n_I}{w-n+2n_I+2m+j}}. \tag{40}$$

where $C_1(p,k,\ell)$ *is the the complexity of a computing a constant fraction (say half of them) of the codewords of weight* p *in a code of length* $k+\ell$ *and dimension* k *and* f *is the function* $f(x) \stackrel{\triangle}{=} \max\left(x(1-x/2), 1-\frac{1}{x}\right)$. *The sum in the denominator of* (40) *is over the domain* $\mathcal{I} = \{(w,m,j) \mid 0 \leq w \leq \min(n-k-\ell, n-n_I), 0 \leq m \leq n/2-n_I, 0 \leq j \leq n/2-n_I-m\}$.

9 Parameter Selection

In the light of the security proof in §6 and the reject sampling method in §5, 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 generalized admissible $(U, U+V)$ code family from random matrices of same size.

To specify the signature scheme, we need to choose the salt size λ_0. From the security proof, it is sufficient to have $\lambda_0 = \log_2(q_{\text{sign}})$ where q_{sign} is the number of signature queries allowed to the adversary. Since $q_{\text{sign}} \leq 2^\lambda$ (λ the security parameter) we choose a conservative $\lambda_0 = \lambda$. We gave in §7 and §8 state-of-the-art algorithms for the two problems mentioned above. This served as a basis for the parameters proposed in Table 1.

For any set of parameters (n, k, w, k_U, k_V) the message security is based on the cost of Dumer's algorithm, as mentioned in §7, in the range of parameters we consider here, this cost cannot be improved as it is done in the binary case. Moreover, considering multiple target (DOOM) does not seem to give an advantage to the attacker. For the range of parameters we have explored, the key security seems to depend solely of the attacks on U, that is (39).

The essential point for parameter selection is the number α introduced at the end of §5. It is a number between 0 and 1 which affects the distribution of the signature weight. We need $\alpha < 1$ to allow an efficient rejection sampling. Large values of α favour the message security while small values favour key security. For each value of α and k/n we derive from equations (23), (24), and (25), the values of w/n, k_U/n, and k_V/n and thus the security estimates from §7 and §8.

The optimal pair in this respect is $(\alpha, k/n) = (0.545, 0.7555)$ for which, in the current state-of-the-art, the best attack has a cost 2^{cn} with $c = 0.02464$. The choice is made such that key attacks and message attacks have the same cost. For 128 bits of security against a classical adversary we obtain the numbers of Table 1. Recall that we are using the ternary alphabet $\mathbb{F}_3$. Those parameters scale linearly, except for the key size which grows as the square of the security. We did not investigate specific quantum attacks, but since known attacks are based on decoding problems, it should be more than enough to increase the classical exponent by a factor two. This leads to quantum safe parameters that are the double of those given in Table 1, except for the key size which would be slightly below 4 megabytes.

Table 1. Proposed Parameters for the Wave Signature Scheme and 128 bits of (classical) security

(n, k, w)	$(5172, 3908, 4980)$
(k_U, k_V)	$(2299, 1609)$
Signature length (bits)	8326
Public key size (MBytes)	0.98

Rejection Sampling Cost. Each of the two steps of our decoding algorithm takes as parameter a weight distribution, respectively $\mathcal{D}_V$ and $\mathcal{D}_U$. For the parameters of Table 1, the first decoding step with $\mathcal{D}_V$ a normalized Laplace distribution of mean $(1-\alpha)k_V$ and variance 18.81 yields a rejection rate of 11%. For the second decoding step, we also choose a Laplace distribution. The mean and variance of that distribution is optimized according to the first step output weight. In the average case the rejection rate is 19%. That is one rejection every 3 or 4 signatures.

10 Concluding Remarks and Further Work

We have presented Wave the first code-based "hash-and-sign" signature scheme which strictly follows the GPV strategy [GPV08]. This strategy provides a very high level of security, but because

of the multiple constraints it imposes, very few schemes managed to comply to it. For instance, only one such scheme based on hard lattice problems [FHK$^+$] was proposed to the recent NIST standardization effort. Our scheme is secure under two assumptions from coding theory. Both of those assumptions relate closely to hard decoding problems. 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 and assess its security. Still, it has a few issues and extensions that are of interest.

Decoding Problems. The message security of Wave relates to the hardness of finding a codeword *far* from a given word. We derived a solver from existing decoding techniques, namely ISD [Pra62, Dum91], but the evolutions of ISD [MMT11, BJMM12] that successfully improved decoding in the *close* codeword setting, fail for high weights. Similarly, multiple target decoders [Sen11] are ineffective here. We believe the problem is exponential by nature, but further studies certainly need to be conducted to understand if, and how much, the exponent can be lowered by new techniques.

Distinguishability. Deciding whether a matrix is a parity check matrix of a generalized $(U, U+V)$ code is also a new problem. As shown in [DST17b] it is hard in the worst case since the problem is NP-complete. In the binary case, $(U, U+V)$ codes have a large hull dimension for some set of parameters which are precisely those used in [DST17b]. In the ternary case the admissible generalized $(U, U+V)$ codes do not suffer from this flaw. The freedom of the choice on the diagonal matrices $\mathbf{D}_i$ is very likely to make the distinguishing problem much harder for generalized $(U, U+V)$ codes than for plain $(U, U+V)$-codes. Coming up with non-metric based distinguishers in the generalized case seems a tantalizing problem here.

Rejection Sampling. Rejection sampling in our algorithm is relatively unobtrusive: a rejection every few signatures with a crude tuning of the decoder. We believe that it can be further improved. Our decoding has two steps. Each step is parametrized by a weight distribution which conditions the output weight distribution. We believe that we can tune those distributions to reduce the probability of rejection to an arbitrarily small value. This task requires a better understanding of the distributions involved. This could offer an interesting trade-off in which the designer/signer would have to precompute and store a set of distributions but in exchange would produce a signing algorithm that emulates a uniform distribution without rejection sampling.

References

[ABB$^+$17] Erdem Alkim, Nina Bindel, Johannes A. Buchmann, Özgür Dagdelen, Edward Eaton, Gus Gutoski, Juliane Krämer, and Filip Pawlega. Revisiting TESLA in the quantum random oracle model. In *Post-Quantum Cryptography 2017*, volume 10346 of *LNCS*, pages 143–162, Utrecht, The Netherlands, June 2017. Springer.

[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.

[BCS13] Daniel J. Bernstein, Tung Chou, and Peter Schwabe. Mcbits: Fast constant-time code-based cryptography. In Guido Bertoni and Jean-Sébastien Coron, editors, *Cryptographic Hardware and Embedded Systems - CHES 2013*, volume 8086 of *LNCS*, pages 250–272. Springer, 2013.

[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.

[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.

[BMvT78] Elwyn Berlekamp, Robert McEliece, and Henk van Tilborg. On the inherent intractability of certain coding problems. *IEEE Trans. Inform. Theory*, 24(3):384–386, May 1978.

[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.

[CD17] André Chailloux and Thomas Debris-Alazard. Tight security reduction in the quantum random oracle model for code-based signature schemes. preprint, September 2017. arXiv:1709.06870.

[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.

[CGG+14] Alain Couvreur, Philippe Gaborit, Valérie Gauthier-Umaña, Ayoub Otmani, and Jean-Pierre Tillich. Distinguisher-based attacks on public-key cryptosystems using Reed-Solomon codes. *Des. Codes Cryptogr.*, 73(2):641–666, 2014.

[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.

[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.

[DST17a] Thomas Debris-Alazard, Nicolas Sendrier, and Jean-Pierre Tillich. A new signature scheme based on $(U|U+V)$ codes. preprint, June 2017. arXiv:1706.08065v1.

[DST17b] Thomas Debris-Alazard, Nicolas Sendrier, and Jean-Pierre Tillich. The problem with the surf scheme. preprint, November 2017. arXiv:1706.08065.

[DT17] Thomas Debris-Alazard and Jean-Pierre Tillich. Statistical decoding. preprint, January 2017. arXiv:1701.07416.

[DT18] Thomas Debris-Alazard and Jean-Pierre Tillich. Two attacks on rank metric code-based schemes: Ranksign and an identity-based-encryption scheme. In *Advances in Cryptology - ASIACRYPT 2018*, LNCS, Brisbane, Australia, December 2018. Springer.

[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.

[FHK+] Pierre-Alain Fouque, Jeffrey Hoffstein, Paul Kirchner, Vadim Lyubashevsky, Thomas Pornin, Thomas Prest, Thomas Ricosset, Gregor Seiler, William Whyte, and Zhenfei Zhang. Falcon: Fast-fourier lattice-based compact signatures over ntru.

[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.

[GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 197–206. ACM, 2008.

[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/`.

[HBPL18] Andreas Huelsing, Daniel J. Bernstein, Lorenz Panny, and Tanja Lange. Official NIST comments made for RaCoSS, 2018. Official NIST comments made for RaCoSS, see https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/round-1/official-com

[Ind03] Piotr Indyk. Better algorithms for high-dimensional proximity problems via asymmetric embeddings. In *Proc. ACM-SIAM SODA*, pages 539–545, 2003.

[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, Evgenii 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, Evgenii Krouk, and Sergei Semenov. *Error Correcting Coding and Security for Data Networks: Analysis of the Superchannel Concept.* John Wiley & Sons, 2005.

[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.

[LKLN17] Wijik Lee, Young-Sik Kim, Yong-Woo Lee, and Jong-Seon No. Post quantum signature scheme based on modified Reed-Muller code pqsigRM. first round submission to the NIST post-quantum cryptography call, November 2017.

[LS12] Gregory Landais and Nicolas Sendrier. Implementing CFS. In *Progress in Cryptology - INDOCRYPT 2012*, volume 7668 of *LNCS*, pages 474–488. 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. In *Proc. IEEE Int. Symposium Inf. Theory - ISIT*, pages 930–934, 2016. for a full version see, 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.

[MP16] Dustin Moody and Ray A. Perlner. Vulnerabilities of "McEliece in the World of Escher". In *Post-Quantum Cryptography 2016*, LNCS. Springer, 2016.

[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.

[PMIB17] Sven Puchinger, Sven Müelich, Karim Ishak, and Martin Bossert. Code-based cryptosystems using generalized concatenated codes. In Ilias S. Kotsireas and Edgar Martínez-Moro, editors, *Applications of Computer Algebra, ACA 2015*, volume 198 of *Proceedings in Mathematics & Statistics*, pages 397–423, Kalamata, Greece, 2017. Springer.

[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.

[SS96] Michael Sipser and Daniel A. Spielman. Expander codes. *IEEE Trans. Inform. Theory*, 42(6):1710–1722, 1996.

[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.

[Ste93a] Jacques Stern. Approximating the number of error locations within a constant ratio is NP-complete. In Gérard Cohen, Teo Mora, and Oscar Moreno, editors, *AAECC 1993*, volume 673 of *LNCS*, pages 325–331. Springer, 1993.

[Ste93b] 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.

A Proofs for §6

A.1 List Emulation

In the security proof, we need to build lists of indices (salts) in $\mathbb{F}_3^{\lambda_0}$. 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 $\mathbb{F}_3^{\lambda_0}$, when calling 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 $L_{\mathbf{m}}$, when calling $L_{\mathbf{m}}$.`contains`($\mathbf{r}$).

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	method list.contains($\mathbf{r}$)
elt, index	return $\mathbf{r} \in \{\texttt{elt}[i], 1 \leq i \leq q_{\text{sign}}\}$
list()	
`index` $\leftarrow 0$	method list.next()
for $i = 1, \ldots, q_{\text{sign}}$	`index` $\leftarrow$ `index` $+ 1$
`elt`$[i] \leftarrow$ `randint`(2^{λ_0})	return `elt`[`index`]

Fig. 13. Standard implementation of the list operations.

Note that in our game we condition on the event that *all elements of $L_{\mathbf{m}}$ are different*. This implies that now $L_{\mathbf{m}}$ is obtained by choosing among the subsets of size q_{sign} of $\mathbb{F}_3^{\lambda_0}$ uniformly at random. We wish to emulate the list operations and never construct them explicitly such that the probabilistic model for $L_{\mathbf{m}}$.`next`() and $L_{\mathbf{m}}$.`contains`($\mathbf{r}$) stays the same as above (but again conditioned on the event that all elements of $L_{\mathbf{m}}$ are different). For this purpose, we want to ensure that at any time we call either $L_{\mathbf{m}}$.`contains`($\mathbf{r}$) or $L_{\mathbf{m}}$.`next`() we have

$$\mathbb{P}(L_{\mathbf{m}}.\texttt{contains}(\mathbf{r}) = \texttt{true}) = \mathbb{P}(\mathbf{r} \in L_{\mathbf{m}} | \mathcal{Q}) \tag{41}$$

$$\mathbb{P}(\mathbf{r} = L_{\mathbf{m}}.\texttt{next}()) = p(\mathbf{r}|\mathcal{Q}) \tag{42}$$

for every $\mathbf{r} \in \mathbb{F}_3^{\lambda_0}$. Here $\mathcal{Q}$ represents the queries to $\mathbf{r}$ made so far and whether or not these $\mathbf{r}$'s belong to $L_{\mathbf{m}}$. Queries to $\mathbf{r}$ can be made through two different calls. The first one is a call of the form `Sign`($\mathbf{m}$) when it chooses $\mathbf{r}$ during the random assignment $\mathbf{r} \hookleftarrow \{0,1\}^{\lambda_0}$. This results in a call to `Hash`($\mathbf{m}, \mathbf{r}$) which queries itself whether $\mathbf{r}$ belongs to $L_{\mathbf{m}}$ or not through the call $L_{\mathbf{m}}$.`contains`($\mathbf{r}$). The answer is necessarily positive in this case. The second way to query $\mathbf{r}$ is by calling `Hash`($\mathbf{m}, \mathbf{r}$) directly. In this case, both answers `true` and `false` are possible. $p(\mathbf{r}|\mathcal{Q})$ represents the probability distribution of $L_{\mathbf{m}}$.`next`() that we have in the above implementation of the list operations given the previous queries $\mathcal{Q}$.

A convenient way to represent $\mathcal{Q}$ is through three lists S, H_{true} and H_{false}. S is the list of $\mathbf{r}$'s that have been queried through a call `Sign`($\mathbf{m}$). They belong necessarily to $L_{\mathbf{m}}$. H_{true} is the set of $\mathbf{r}$'s that have not been queried so far through a call to `Sign`($\mathbf{m}$) but have been queried through a direct call `Hash`($\mathbf{m}, \mathbf{r}$) and for which $L_{\mathbf{m}}$.`contains`($\mathbf{r}$) returned `true`. H_{false} is the list of $\mathbf{r}$'s that have been queried by a call of the form `Hash`($\mathbf{m}, \mathbf{r}$) and $L_{\mathbf{m}}$.`contains`($\mathbf{r}$) returned `false`.

We clearly have

$$\mathbb{P}(\mathbf{r} \in L_{\mathbf{m}}|\mathcal{Q}) = 0 \text{ if } \mathbf{r} \in H_{\text{false}} \tag{43}$$

$$\mathbb{P}(\mathbf{r} \in L_{\mathbf{m}}|\mathcal{Q}) = 1 \text{ if } \mathbf{r} \in S \cup H_{\text{true}} \tag{44}$$

$$\mathbb{P}(\mathbf{r} \in L_{\mathbf{m}}|\mathcal{Q}) = \frac{q_{\text{sign}} - |H_{\text{true}}| - |S|}{2^{\lambda_0} - |H_{\text{true}}| - |S| - |H_{\text{false}}|} \text{ else.} \tag{45}$$

To compute the probability distribution $p(\mathbf{r}|\mathcal{Q})$ it is helpful to notice that

$$\mathbb{P}(L_{\mathbf{m}}.\texttt{next}() \text{ outputs an element of } H_{\text{true}}) = \frac{|H_{\text{true}}|}{q_{\text{sign}} - |S|}. \tag{46}$$

This can be used to derive $p(\mathbf{r}|\mathcal{Q})$ as follows

$$p(\mathbf{r}|\mathcal{Q}) = 0 \text{ if } \mathbf{r} \in H_{\text{false}} \cup S \tag{47}$$

$$p(\mathbf{r}|\mathcal{Q}) = \frac{1}{q_{\text{sign}} - S} \text{ if } \mathbf{r} \in H_{\text{true}} \tag{48}$$

$$p(\mathbf{r}|\mathcal{Q}) = \frac{q_{\text{sign}} - |S| - |H_{\text{true}}|}{(q_{\text{sign}} - S)(2^{\lambda_0} - |H_{\text{true}}| - |S| - |H_{\text{false}}|)} \text{ else.} \tag{49}$$

(47) is obvious. (48) follows from that all elements of H_{true} have the same probability to be chosen as return value for $L_{\mathbf{m}}.\texttt{next}()$ and (46). (49) follows by a similar reasoning by arguing (i) that all the elements of $\mathbb{F}_3^{\lambda_0} \setminus (S \cup H_{\text{true}} \cup H_{\text{false}})$ have the same probability to be chosen as return value for $L_{\mathbf{m}}.\texttt{next}()$, (ii) the probability that $L_{\mathbf{m}}.\texttt{next}()$ outputs an element of $\mathbb{F}_3^{\lambda_0} \setminus (S \cup H_{\text{true}} \cup H_{\text{false}})$ is the probability that it does not output an element of H_{true} which is $1 - \frac{|H_{\text{true}}|}{q_{\text{sign}} - |S|} = \frac{q_{\text{sign}} - |S| - |H_{\text{true}}|}{q_{\text{sign}} - |S|}$.

Figure 14 explains how we perform the emulation of the list operations so that they perform similarly to genuine list operations as specified above. The idea is to create and to operate explicitly on the lists S, H_{true} and H_{false} described earlier. We have chosen there

$$\beta = \frac{q_{\text{sign}} - |H_{\text{true}}| - |S|}{2^{\lambda_0} - |H_{\text{true}}| - |S| - |H_{\text{false}}|} \text{ and } \gamma = \frac{|H_{\text{true}}|}{q_{\text{sign}} - |S|}.$$

we also assume that when we call `randomPop()` on a list it outputs an element of the list uniformly at random and removes this element from it. The method `push` adds an element in a list. The procedure `rand`() picks a real number between 0 and 1 uniformly at random.

class list	method list.contains($\mathbf{r}$)	method list.next()
H_{true}, H_{false}, S	if $\mathbf{r} \notin H_{\text{true}} \cup H_{\text{false}} \cup S$	if $\texttt{rand}() \leq \gamma$
list()	if rand() $\leq \beta$	$\mathbf{r} \leftarrow H_{\text{true}}.\texttt{randomPop}()$
$H_{\text{true}} \leftarrow \emptyset$	$H_{\text{true}}.\texttt{push}(\mathbf{r})$	else
$H_{\text{false}} \leftarrow \emptyset$	else	$\mathbf{r} \hookleftarrow \mathbb{F}_3^{\lambda_0} \setminus (H_{\text{true}} \cup S \cup H_{\text{false}})$
$S \leftarrow \emptyset$	$H_{\text{false}}.\texttt{push}(\mathbf{r})$	$S.\texttt{push}(\mathbf{r})$
	return $\mathbf{r} \in H_{\text{true}} \cup S$	return $\mathbf{r}$

Fig. 14. Emulation of the list operations.

The correctness of this emulation follows directly from the calculations given above. For instance the correctness of the call $L_{\mathbf{m}}.\texttt{next}()$ follows from the fact that with probability $\frac{|H_{\text{true}}|}{q_{\text{sign}} - |S|} = \gamma$ it outputs an element of H_{true} chosen uniformly at random (see (46)). In such a case the corresponding element has to be moved from H_{true} to S (since it has been queried now through a call to $\texttt{Sign}(\mathbf{m})$). The correctness of $L_{\mathbf{m}}.\texttt{contains}(\mathbf{r})$ is a direct consequence of the formulas for $\mathbb{P}(\mathbf{r} \in L_{\mathbf{m}}|\mathcal{Q})$ given in (43), (44) and (45). All `push`, `pop`, membership testing above can be implemented in time proportional to λ_0.

A.2 Proof of Lemma 3

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 3 of §6.3. We recall that in $\mathcal{S}_{\text{code}}$ 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 7. *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:

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

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 3.

B Proof of Proposition 3

Our goal in this subsection is to prove Proposition 3 of §5.1.

Probabilistic notation. Recall that we denote by $\mathcal{U}_w$ the uniform distribution over S_w.

Vector notation. If $\mathbf{e} \in \mathbb{F}_3^n$ and $i \in [\![1,n]\!]$, we will denote by $\mathbf{e}(i)$ the i-th component of $\mathbf{e}$. Furthermore, if we split $\mathbf{e}$ as $(\mathbf{e}_1,\mathbf{e}_2)$ where $\mathbf{e}_1$ (resp. $\mathbf{e}_2$) is the vector formed by its first (resp. last) $n/2$ coordinates. We define $\mathbf{e}^{(i)}$ for all $i \in [\![1,n/2]\!]$ as:

$$\mathbf{e}^{(i)} \triangleq \boxed{\begin{matrix}\mathbf{e}_1(i)\end{matrix}}\boxed{\begin{matrix}\mathbf{e}_2(i)\end{matrix}}$$

We start by computing here a distribution that will be useful in the following.

Proposition 11. *Let $\mathbf{e} = (\mathbf{e}_1,\mathbf{e}_2)$ be random a random variable whose distribution is $\mathcal{U}_w$ where the $\mathbf{e}_i$'s are vectors of $\mathbb{F}_3^{n/2}$. We have for all $j \in [\![0,n/2]\!]$:*

$$\mathbb{P}_{\mathbf{e}}(|\mathbf{e}_1-\mathbf{e}_2|=j) = \mathbb{P}_{\mathbf{e}}(|\mathbf{e}_1+\mathbf{e}_2|=j) = \frac{1}{2^w\binom{n}{w}} \sum_{\substack{p=0 \\ w+p\equiv 0 \mod 2}}^{j} \binom{j}{p}\binom{\frac{w+p}{2}}{j}\binom{n/2}{\frac{w+p}{2}} 2^{\frac{w+3p}{2}}$$

Lemma 8. *Let $\mathbf{e} = (\mathbf{e}_1,\mathbf{e}_2)$ be a word of S_w. The weight $|\mathbf{e}_1+\mathbf{e}_2|$ is given by the number of $i \in [\![1,n/2]\!]$ such that $\mathbf{e}^{(i)}$ is one of the following element:*

$$\boxed{\begin{matrix}-1\\0\end{matrix}} \;;\; \boxed{\begin{matrix}0\\-1\end{matrix}} \;;\; \boxed{\begin{matrix}1\\0\end{matrix}} \;;\; \boxed{\begin{matrix}0\\1\end{matrix}} \;;\; \boxed{\begin{matrix}-1\\-1\end{matrix}} \;;\; \boxed{\begin{matrix}1\\1\end{matrix}}$$

Proof (Proposition 11). Let us enumerate the number of errors $\mathbf{e} = (\mathbf{e}_1, \mathbf{e}_2) \in S_w$ which verify $|\mathbf{e}_1 + \mathbf{e}_2| = j$. For this let introduce the following intermediary weights:

$$|\mathbf{e}|^1 \triangleq \left| \left\{ i \in [\![1, n/2]\!] : \mathbf{e}^{(i)} \in \left\{ \begin{bmatrix} -1 \\ 0 \end{bmatrix} \; ; \; \begin{bmatrix} 0 \\ -1 \end{bmatrix} \; ; \; \begin{bmatrix} 1 \\ 0 \end{bmatrix} \; ; \; \begin{bmatrix} 0 \\ 1 \end{bmatrix} \right\} \right\} \right|$$

$$|\mathbf{e}|^2 \triangleq \left| \left\{ i \in [\![1, n/2]\!] : \mathbf{e}^{(i)} \in \left\{ \begin{bmatrix} -1 \\ -1 \end{bmatrix} \; ; \; \begin{bmatrix} 1 \\ 1 \end{bmatrix} \right\} \right\} \right|$$

$$|\mathbf{e}|^3 \triangleq \left| \left\{ i \in [\![1, n/2]\!] : \mathbf{e}^{(i)} \in \left\{ \begin{bmatrix} 1 \\ -1 \end{bmatrix} \; ; \; \begin{bmatrix} -1 \\ 1 \end{bmatrix} \right\} \right\} \right|$$

It is easily verified that,

$$|\mathbf{e}| = |\mathbf{e}|^1 + 2|\mathbf{e}|^2 + 2|\mathbf{e}|^3 \quad \text{and} \quad |\mathbf{e}_1 + \mathbf{e}_2| = |\mathbf{e}|^1 + |\mathbf{e}|^2$$

In this way,

$$\begin{aligned}
\mathbb{P}_{\mathbf{e}}\left(|\mathbf{e}_1 + \mathbf{e}_2| = j\right) &= \sum_{p=0}^{j} \mathbb{P}_{\mathbf{e}}\left(|\mathbf{e}|^2 = j - p \;\middle|\; |\mathbf{e}|^1 = p\right) \mathbb{P}_{\mathbf{e}}\left(|\mathbf{e}|^1 = p\right) \\
&= \sum_{p=0}^{j} \mathbb{P}_{\mathbf{e}}\left(|\mathbf{e}|^2 = j - p, |\mathbf{e}|^1 = p\right) \\
&= \frac{1}{2^w \binom{n}{w}} \sum_{p=0}^{j} \binom{n/2}{p} 4^p \binom{n/2 - p}{j - p} 2^{j-p} \binom{n/2 - p - (j-p)}{\frac{w-p-2(j-p)}{2}} 2^{\frac{w-p-2(j-p)}{2}} \\
&= \frac{1}{2^w \binom{n}{w}} \sum_{p=0}^{j} \binom{n/2}{p} \binom{n/2 - p}{j - p} 2^{2p+j-p} \binom{n/2 - j}{\frac{w-2j+p}{2}} 2^{\frac{w-2j+p}{2}} \\
&= \frac{1}{2^w \binom{n}{w}} \sum_{p=0}^{j} \binom{n/2}{p} \binom{n/2 - p}{j - p} \binom{n/2 - j}{\frac{w+p}{2} - j} 2^{\frac{w+3p}{2}}
\end{aligned}$$

It remains now the following computation to conclude:

$$\begin{aligned}
\binom{n/2}{p} \binom{n/2 - p}{j - p} \binom{n/2 - j}{\frac{w+p}{2} - j} &= \frac{(n/2)!}{(n/2-p)!p!} \frac{(n/2-p)!}{(n/2-j)!(j-p)!} \frac{(n/2-j)!}{((n-w-p)/2)!((w+p)/2-j)!} \\
&= \frac{(n/2)!}{p!} \frac{1}{(j-p)!} \frac{1}{((n-w-p)/2)!((w+p)/2-j)!} \\
&= \frac{(n/2)!}{((n-w-p)/2)!((w+p)/2)!} \frac{((w+p)/2)!}{((w+p)/2-j)!j!} \frac{j!}{(j-p)!p!} \\
&= \binom{n/2}{\frac{w+p}{2}} \binom{\frac{w+p}{2}}{j} \binom{j}{p}
\end{aligned}$$

which concludes the proof.

Let us recall now Proposition 3 that we want to prove. We recall first that we denote by $\mathbf{H}_{\text{pk}}$ the random matrix chosen as the public parity-check matrix of our scheme. Let us recall that it is obtained as

$$\mathbf{H}_{\text{pk}} = \mathbf{S}\mathbf{H}_{\text{sk}}\mathbf{P} \quad \text{with} \quad \mathbf{H}_{\text{sk}} \triangleq \begin{pmatrix} \mathbf{H}_U\mathbf{D}_4\mathbf{M} & -\mathbf{H}_U\mathbf{D}_2\mathbf{M} \\ \mathbf{H}_V\mathbf{D}_3\mathbf{M} & -\mathbf{H}_V\mathbf{D}_1\mathbf{M} \end{pmatrix}$$

where $\mathbf{D}_1, \cdots, \mathbf{D}_4$ are four diagonal matrices which verify (12) and $\mathbf{M} \triangleq (\mathbf{D}_1\mathbf{D}_4 - \mathbf{D}_3\mathbf{D}_2)^{-1}$, $\mathbf{S}$ is chosen uniformly at random among the invertible ternary matrices of size $(n-k) \times (n-k)$, $\mathbf{H}_U$ is chosen uniformly at random among the ternary matrices of size $(n/2 - k_U) \times n/2$, $\mathbf{H}_V$ is chosen uniformly at random among the ternary matrices of size $(n/2 - k_V) \times n/2$ and $\mathbf{P}$ is chosen uniformly at random among the permutation matrices of size $n \times n$.

Proposition 3. *Let $\mathcal{D}_w^{\mathbf{H}}$ be the distribution of the syndromes $\mathbf{eH}^\intercal$ when $\mathbf{e}$ is drawn uniformly at random among the ternary vectors of weight w and $\mathcal{U}$ be the uniform distribution over the syndrome space $\mathbb{F}_3^{n-k}$. We have*

$$\mathbb{E}_{\mathbf{H}_{\mathrm{pk}}}\left(\rho(\mathcal{D}_w^{\mathbf{H}_{\mathrm{pk}}},\mathcal{U})\right) \leq \frac{1}{2}\sqrt{\varepsilon}$$

with

$$\varepsilon = \frac{3^{n-k}}{2^w\binom{n}{w}} + \sum_{j=0}^{w} \frac{3^{n/2-k_V}}{2^j\binom{n/2}{j}2^{2w}\binom{n/2}{w}^2}\left(\sum_{\substack{p=0\\ w+p\equiv 0 \bmod 2}}^{j}\binom{j}{p}\binom{n/2}{\frac{w+p}{2}}\binom{\frac{w+p}{2}}{j}2^{\frac{w+3p}{2}}\right)^2$$
$$+ 3^{n/2-k_U}\sum_{l=0}^{n_I}\binom{n_I}{l}2^l\left(\frac{\binom{n-n_I}{w-l}2^{w-l}}{\binom{n}{w}2^w}\right)^2\sum_{j=0}^{n/2-n_I}\frac{1}{2^j\binom{n/2-n_I}{j}}$$
$$\left(\frac{1}{\binom{n}{w}2^w}\sum_{h=0}^{j}\sum_{p=0}^{n/2-n_I-j}\binom{n/2-n_I}{h}\binom{n/2-n_I-h}{j-h}\binom{n/2-n_I-j}{p}\binom{2n_I}{w-j-h-2p}2^{w-j-h-2p}\right)^2 \tag{22}$$

Proposition 3 is based on two lemmas. The first one is the following:

Lemma 9. *Let $\mathbf{y}$ be a non-zero vector of $\mathbb{F}_3^n$ and $\mathbf{s}$ an arbitrary element in $\mathbb{F}_3^r$. 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*

$$\mathbb{P}\left(\mathbf{yH}^\intercal = \mathbf{s}\right) = \frac{1}{3^r}$$

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

$$\sum_j h_{ij}y_j = s_i \tag{50}$$

for all i in $\{1,\ldots,r\}$. 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 (50) as $h_{i1} = \sum_{j>1} h_{ij}y_j$. This event happens with probability $\frac{1}{3}$ for a given i and with probability $\frac{1}{3^r}$ on all r events simultaneously due to the independence of the h_{ij}'s.

Let us now consider the following lemma which is a variation of the left over hash lemma:

Lemma 1. *Consider a finite family $\mathcal{H} = (h_i)_{i\in I}$ of functions from a finite set E to a finite set F. Denote by ε the bias of the collision probability, i.e. the quantity such that*

$$\mathbb{P}_{h,e,e'}(h(e) = h(e')) = \frac{1}{|F|}(1+\varepsilon)$$

where h is drawn uniformly at random in $\mathcal{H}$, e and e' are drawn uniformly at random in E. Let $\mathcal{U}$ be the uniform distribution over F and $\mathcal{D}(h)$ be the distribution of the outputs $h(e)$ when e is chosen uniformly at random in E. We have

$$\mathbb{E}_h\left\{\rho(\mathcal{D}(h),\mathcal{U})\right\} \leq \frac{1}{2}\sqrt{\varepsilon}.$$

Proof. Let $q_{h,f}$ be the probability distribution of the discrete random variable $(h_0, h_0(e))$ where h_0 is drawn uniformly at random in $\mathcal{H}$ and e drawn uniformly at random in E (i.e. $q_{h,f} = \mathbb{P}_{h_0,e}(h_0 =$

$h, h_0(e) = f)$). By definition of the statistical distance we have

$$\begin{aligned}\mathbb{E}_h\left\{\rho(\mathcal{D}(h),\mathcal{U})\right\} &= \sum_{h\in\mathcal{H}} \frac{1}{|\mathcal{H}|}\rho(\mathcal{D}(h),\mathcal{U}) \\ &= \sum_{h\in\mathcal{H}} \frac{1}{2|\mathcal{H}|}\sum_{f\in F}\left|\mathbb{P}_e(h(e)=f) - \frac{1}{|F|}\right| \\ &= \frac{1}{2}\sum_{(h,f)\in\mathcal{H}\times F}\left|\mathbb{P}_{h_0,e}(h_0=h, h_0(e)=f) - \frac{1}{|\mathcal{H}|\cdot|F|}\right| \\ &= \frac{1}{2}\sum_{(h,f)\in\mathcal{H}\times F}\left|q_{h,f} - \frac{1}{|\mathcal{H}|\cdot|F|}\right|. \end{aligned} \tag{51}$$

Using the Cauchy-Schwarz inequality, we obtain

$$\sum_{(h,f)\in\mathcal{H}\times F}\left|q_{h,f} - \frac{1}{|\mathcal{H}|\cdot|F|}\right| \leq \sqrt{\sum_{(h,f)\in\mathcal{H}\times F}\left(q_{h,f} - \frac{1}{|\mathcal{H}|\cdot|F|}\right)^2}\cdot\sqrt{|\mathcal{H}|\cdot|F|}. \tag{52}$$

Let us observe now that

$$\begin{aligned}\sum_{(h,f)\in\mathcal{H}\times F}\left(q_{h,f} - \frac{1}{|\mathcal{H}|\cdot|F|}\right)^2 &= \sum_{h,f}\left(q_{h,f}^2 - 2\frac{q_{h,f}}{|\mathcal{H}|\cdot|F|} + \frac{1}{|\mathcal{H}|^2\cdot|F|^2}\right) \\ &= \sum_{h,f} q_{h,f}^2 - 2\frac{\sum_{h,f} q_{h,f}}{|\mathcal{H}|\cdot|F|} + \frac{1}{|\mathcal{H}|\cdot|F|} \\ &= \sum_{h,f} q_{h,f}^2 - \frac{1}{|\mathcal{H}|\cdot|F|}. \end{aligned} \tag{53}$$

Consider for $i \in \{0,1\}$ independent random variables h_i and e_i that are drawn uniformly at random in $\mathcal{H}$ and E respectively. We continue this computation by noticing now that

$$\begin{aligned}\sum_{h,f} q_{h,f}^2 &= \sum_{h,f}\mathbb{P}_{h_0,e_0}(h_0=h, h_0(e_0)=f)\mathbb{P}_{h_1,e_1}(h_1=h, h_1(e_1)=f) \\ &= \mathbb{P}_{h_0,h_1,e_0,e_1}\left(h_0=h_1, h_0(e_0)=h_1(e_1)\right) \\ &= \frac{\mathbb{P}_{h_0,e_0,e_1}\left(h_0(e_0)=h_0(e_1)\right)}{|\mathcal{H}|} \\ &= \frac{1+\varepsilon}{|\mathcal{H}|\cdot|F|}. \end{aligned} \tag{54}$$

By substituting for $\sum_{h,f} q_{h,f}^2$ the expression obtained in (54) into (53) and then back into (52) we finally obtain

$$\sum_{(h,f)\in\mathcal{H}\times F}\left|q_{h,f} - \frac{1}{|\mathcal{H}|\cdot|F|}\right| \leq \sqrt{\frac{1+\varepsilon}{|\mathcal{H}|\cdot|F|} - \frac{1}{|\mathcal{H}|\cdot|F|}}\sqrt{|\mathcal{H}|\cdot|F|} = \sqrt{\frac{\varepsilon}{|\mathcal{H}|\cdot|F|}}\sqrt{|\mathcal{H}|\cdot|F|} = \sqrt{\varepsilon}.$$

This finishes the proof of our lemma. □

In order to use this lemma to bound the statistical distance we are interested in, we perform now the following computation

Lemma 10. *Assume that* $\mathbf{x}$ *and* $\mathbf{y}$ *are random vectors of* S_w *that are drawn uniformly at random in this set. We have*

$$\mathbb{P}_{\mathbf{H}_{\mathrm{pk}},\mathbf{x},\mathbf{y}}\left(\mathbf{x}\mathbf{H}_{\mathrm{pk}}^{\intercal} = \mathbf{y}\mathbf{H}_{\mathrm{pk}}^{\intercal}\right) \leq \frac{1}{3^{n-k}}(1+\varepsilon) \text{ with } \varepsilon \text{ given in Proposition 3.}$$

Proof. Recall that $\mathbf{H}_{\text{pk}}$ is obtained as

$$\mathbf{H}_{\text{pk}} = \mathbf{S}\mathbf{H}_{\text{sk}}\mathbf{P} \quad \text{with} \quad \mathbf{H}_{\text{sk}} \stackrel{\triangle}{=} \begin{pmatrix} \mathbf{H}_U\mathbf{D}_4\mathbf{M} & -\mathbf{H}_U\mathbf{D}_2\mathbf{M} \\ \mathbf{H}_V\mathbf{D}_3\mathbf{M} & -\mathbf{H}_V\mathbf{D}_1\mathbf{M} \end{pmatrix}$$

where $\mathbf{D}_1, \cdots, \mathbf{D}_4$ are four diagonal matrices which verify (12) in Definition 2 in §4.2 and $\mathbf{M} \stackrel{\triangle}{=} (\mathbf{D}_1\mathbf{D}_4 - \mathbf{D}_3\mathbf{D}_2)^{-1}$, $\mathbf{S}$ is chosen uniformly at random among $\mathbb{F}_3^{(n-k)\times(n-k)}$, $\mathbf{H}_U$ is chosen uniformly at random among $\mathbb{F}_3^{(n/2-k_U)\times n/2}$, $\mathbf{H}_V$ is chosen uniformly at random among $\mathbb{F}_3^{(n/2-k_V)\times n/2}$ and $\mathbf{P}$ is chosen uniformly at random among the permutation matrices of size $n \times n$. As $\mathbf{S}$ is non-singular and $\mathbf{P}$ is a permutation, the probability of the event $\mathbf{x}\mathbf{H}_{\text{pk}}^{\intercal} = \mathbf{y}\mathbf{H}_{\text{pk}}^{\intercal}$ is the same as the probability of the event

$$\begin{pmatrix} \mathbf{H}_U\mathbf{D}_4\mathbf{M} & -\mathbf{H}_U\mathbf{D}_2\mathbf{M} \\ \mathbf{H}_V\mathbf{D}_3\mathbf{M} & -\mathbf{H}_V\mathbf{D}_1\mathbf{M} \end{pmatrix} \mathbf{x}^{\intercal} = \begin{pmatrix} \mathbf{H}_U\mathbf{D}_4\mathbf{M} & -\mathbf{H}_U\mathbf{D}_2\mathbf{M} \\ \mathbf{H}_V\mathbf{D}_3\mathbf{M} & -\mathbf{H}_V\mathbf{D}_1\mathbf{M} \end{pmatrix} \mathbf{y}^{\intercal}.$$

Let $\mathbf{x}$ be a vector of $\mathbb{F}_3^n$, we will denote in the following by $\mathbf{x}_1$ (resp. $\mathbf{x}_2$) the vector formed by its first (resp. last) $n/2$ coordinates. In other words, the probability we are looking for is

$$\mathbb{P}\big(((\mathbf{x}_1 - \mathbf{y}_1)\mathbf{D}_4\mathbf{M} - (\mathbf{x}_2 - \mathbf{y}_2)\mathbf{D}_2\mathbf{M})\,\mathbf{H}_U^{\intercal} = \mathbf{0}, ((\mathbf{x}_1 - \mathbf{y}_1)\mathbf{D}_3\mathbf{M} - (\mathbf{x}_2 - \mathbf{y}_2)\mathbf{D}_1\mathbf{M})\,\mathbf{H}_V^{\intercal} = \mathbf{0}\big).$$

where the probability is taken over $\mathbf{H}_U, \mathbf{H}_V, \mathbf{x}, \mathbf{y}$. To compute the previous probability we use Lemma 9 which says that:

$$\mathbb{P}_{\mathbf{H}}\left(\mathbf{e}\mathbf{H}^{\intercal} = \mathbf{0}\right) = \frac{1}{3^{n-k}} \text{ if } \mathbf{e} \neq 0 \text{ and 1 otherwise} \tag{55}$$

when $\mathbf{H}$ is chosen uniformly at random in $\mathbb{F}_3^{(n-k)\times n}$. This lemma motivates to distinguish between four disjoint events

Event 1:

$$\mathcal{E}_1 \stackrel{\triangle}{=} \{(\mathbf{x}_1 - \mathbf{y}_1)\mathbf{D}_4\mathbf{M} = (\mathbf{x}_2 - \mathbf{y}_2)\mathbf{D}_2\mathbf{M}, \quad (\mathbf{x}_1 - \mathbf{y}_1)\mathbf{D}_3\mathbf{M} \neq (\mathbf{x}_2 - \mathbf{y}_2)\mathbf{D}_1\mathbf{M}\}$$

Event 2:

$$\mathcal{E}_2 \stackrel{\triangle}{=} \{(\mathbf{x}_1 - \mathbf{y}_1)\mathbf{D}_4\mathbf{M} \neq (\mathbf{x}_2 - \mathbf{y}_2)\mathbf{D}_2\mathbf{M}, \quad (\mathbf{x}_1 - \mathbf{y}_1)\mathbf{D}_3\mathbf{M} = (\mathbf{x}_2 - \mathbf{y}_2)\mathbf{D}_1\mathbf{M}\}$$

Event 3:

$$\mathcal{E}_3 \stackrel{\triangle}{=} \{(\mathbf{x}_1 - \mathbf{y}_1)\mathbf{D}_4\mathbf{M} \neq (\mathbf{x}_2 - \mathbf{y}_2)\mathbf{D}_2\mathbf{M}, \quad (\mathbf{x}_1 - \mathbf{y}_1)\mathbf{D}_3\mathbf{M} \neq (\mathbf{x}_2 - \mathbf{y}_2)\mathbf{D}_1\mathbf{M}\}$$

Event 4:

$$\mathcal{E}_4 \stackrel{\triangle}{=} \{(\mathbf{x}_1 - \mathbf{y}_1)\mathbf{D}_4\mathbf{M} = (\mathbf{x}_2 - \mathbf{y}_2)\mathbf{D}_2\mathbf{M}, \quad (\mathbf{x}_1 - \mathbf{y}_1)\mathbf{D}_3\mathbf{M} = (\mathbf{x}_2 - \mathbf{y}_2)\mathbf{D}_1\mathbf{M}\}$$

Under these events we get thanks to (55) and $k = k_U + k_V$:

$$\begin{aligned}
&\mathbb{P}_{\mathbf{H}_{\text{sk}},\mathbf{x},\mathbf{y}}\left(\mathbf{x}\mathbf{H}_{\text{sk}}^{\intercal} = \mathbf{y}\mathbf{H}_{\text{sk}}^{\intercal}\right) \\
&= \sum_{i=1}^{4} \mathbb{P}_{\mathbf{H}_{\text{sk}}}\left(\mathbf{x}\mathbf{H}_{\text{sk}}^{\intercal} = \mathbf{y}\mathbf{H}_{\text{sk}}^{\intercal} | \mathcal{E}_i\right) \mathbb{P}_{\mathbf{x},\mathbf{y}}\left(\mathcal{E}_i\right) \\
&= \frac{\mathbb{P}_{\mathbf{x},\mathbf{y}}\left(\mathcal{E}_1\right)}{3^{n/2-k_V}} + \frac{\mathbb{P}_{\mathbf{x},\mathbf{y}}\left(\mathcal{E}_2\right)}{3^{n/2-k_U}} + \frac{\mathbb{P}_{\mathbf{x},\mathbf{y}}\left(\mathcal{E}_3\right)}{3^{n-k}} + \mathbb{P}_{\mathbf{x},\mathbf{y}}\left(\mathcal{E}_4\right) \\
&= \frac{1}{3^{n-k}}\left(\frac{\mathbb{P}\left(\mathcal{E}_1\right)}{3^{n/2-k_V-n+k}} + \frac{\mathbb{P}\left(\mathcal{E}_2\right)}{3^{n/2-k_U-n+k}} + \mathbb{P}\left(\mathcal{E}_3\right) + 3^{n-k}\mathbb{P}\left(\mathcal{E}_4\right)\right) \\
&\leq \frac{1}{3^{n-k}}\left(1 + 3^{n/2-k_U}\mathbb{P}\left(\mathcal{E}_1\right) + 3^{n/2-k_V}\mathbb{P}\left(\mathcal{E}_2\right) + 3^{n-k}\mathbb{P}(\mathcal{E}_4)\right),
\end{aligned} \tag{56}$$

where we used for the last inequality the trivial upper-bound $\mathbb{P}(\mathcal{E}_3) \leq 1$. Let us now upper-bound (or compute) the probabilities of the events $\mathcal{E}_1$, $\mathcal{E}_2$ and $\mathcal{E}_4$. For $\mathcal{E}_4$, recall that from the definition of admissible generalized $(U, U+V)$-codes and especially the fact that $\mathbf{D} = \begin{pmatrix} \mathbf{D}_1 & \mathbf{D}_2 \\ \mathbf{D}_3 & \mathbf{D}_4 \end{pmatrix}$ and $\mathbf{M}$ are invertible we clearly have

$$\mathcal{E}_4 = \{\mathbf{x}_1 = \mathbf{y}_1, \mathbf{x}_2 = \mathbf{y}_2\}$$

which easily gives

$$\mathbb{P}_{\mathbf{x},\mathbf{y}}(\mathcal{E}_4) = \mathbb{P}(\mathbf{x} = \mathbf{y}) = \frac{1}{2^w \binom{n}{w}}. \tag{57}$$

Let us now estimate to probability of $\mathcal{E}_2$ for which derive the following upper-bound:

$$\mathbb{P}(\mathcal{E}_2) \leq \mathbb{P}\left((\mathbf{x}_1 - \mathbf{y}_1)\mathbf{D}_3\mathbf{M} = (\mathbf{x}_2 - \mathbf{y}_2)\mathbf{D}_1\mathbf{M}\right)$$

But now from the condition (12):

$$\forall i \in [\![1, n/2]\!], \quad \mathbf{D}_3(i,i)\mathbf{D}_1(i,i) \neq 0$$

of the definition of admissible generalized $(U, U+V)$-codes and the fact that $\mathbf{M}$ is an invertible diagonal matrix we have:

$$(\mathbf{x}_1 - \mathbf{y}_1)\mathbf{D}_3\mathbf{M} = (\mathbf{x}_2 - \mathbf{y}_2)\mathbf{D}_1\mathbf{M} \iff \forall i \in [\![1, n/2]\!], \quad (\mathbf{x}_1 - \mathbf{y}_1)(i) = \pm(\mathbf{x}_2 - \mathbf{y}_2)(i)$$

Let us notice that distribution of a vector $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2)$ uniformly picked at random in S_w is the same as by multiplying some of its components to -1 . In this way we have

$$\mathbb{P}(\mathcal{E}_2) \leq \mathbb{P}(\mathbf{x}_1 - \mathbf{y}_1 = \mathbf{x}_2 - \mathbf{y}_2)$$

To upper-bound $\mathbb{P}(\mathbf{x}_1 - \mathbf{x}_2 = \mathbf{y}_1 - \mathbf{y}_2)$, let us derive the distribution of $\mathbf{x}_1 - \mathbf{x}_2$. We first observe that

$$\begin{aligned}
\mathbb{P}(\mathbf{x}_1 - \mathbf{x}_2 = \mathbf{e}) &= \mathbb{P}\Big(\mathbf{x}_1 - \mathbf{x}_2 = \mathbf{e} \Big| \, |\mathbf{x}_1 - \mathbf{x}_2| = w_e\Big)\mathbb{P}(|\mathbf{x}_1 - \mathbf{x}_2| = w_e) \\
&\leq \frac{1}{2^{w_e}\binom{n/2}{w_e}} \frac{1}{2^w \binom{n}{w}} \sum_{\substack{p=0 \\ w+p \equiv 0 \mod 2}}^{w_e} \binom{w_e}{p} \binom{n/2}{\frac{w+p}{2}} \binom{\frac{w+p}{2}}{w_e} 2^{\frac{w+3p}{2}} \quad \text{(see Proposition 11)}
\end{aligned} \tag{58}$$

From this we deduce that

$$\begin{aligned}
\mathbb{P}(\mathbf{x}_1 - \mathbf{y}_1 = \mathbf{x}_1 - \mathbf{y}_2) &= \sum_{j=0}^{n/2} \sum_{\mathbf{e} \in \mathbb{F}_3^{n/2} : |\mathbf{e}| = j} \mathbb{P}_{\mathbf{x}}(\mathbf{x}_1 - \mathbf{x}_2 = \mathbf{e})^2 \\
&\leq \sum_{j=0}^{n/2} 2^j \binom{n/2}{j} \left(\frac{1}{2^j \binom{n/2}{j} 2^w \binom{n}{w}} \sum_{\substack{p=0 \\ w+p \equiv 0 \mod 2}}^{j} \binom{j}{p} \binom{n/2}{\frac{w+p}{2}} \binom{\frac{w+p}{2}}{j} 2^{\frac{w+3p}{2}} \right)^2 \quad \text{(by Eq. (58))} \\
&= \sum_{j=0}^{n/2} \frac{1}{2^j \binom{n/2}{j} 2^{2w} \binom{n}{w}^2} \left(\sum_{\substack{p=0 \\ w+p \equiv 0 \mod 2}}^{j} \binom{j}{p} \binom{n/2}{\frac{w+p}{2}} \binom{\frac{w+p}{2}}{j} 2^{\frac{w+3p}{2}} \right)^2
\end{aligned}$$

Therefore we deduce that

$$\mathbb{P}(\mathcal{E}_2) \leq \sum_{j=0}^{n/2} \frac{1}{2^j \binom{n/2}{j} 2^{2w} \binom{n}{w}^2} \left(\sum_{\substack{p=0 \\ w+p \equiv 0 \mod 2}}^{j} \binom{j}{p} \binom{n/2}{\frac{w+p}{2}} \binom{\frac{w+p}{2}}{j} 2^{\frac{w+3p}{2}} \right)^2 \tag{59}$$

To upper bound $\mathcal{E}_1$ let us first recall the following definition

Definition 3. (number of V blocks of type I). *In a generalized $(U, U+V)$ code of length n associated to the 4-tuple of diagonal matrices of size $n/2$ $(\mathbf{D}_1, \mathbf{D}_2, \mathbf{D}_3, \mathbf{D}_4)$, the number of V blocks of type I, which we denote by n_I, is defined by:*

$$n_I \stackrel{\triangle}{=} |\{1 \leq i \leq n/2 : \mathbf{D}_2(i,i)\mathbf{D}_4(i,i) = 0\}| .$$

In other words from the fact that $\mathbf{M}$ is invertible, the event $\mathcal{E}_1 = \{(\mathbf{x}_1 - \mathbf{y}_1)\mathbf{D}_4\mathbf{M} = (\mathbf{x}_2 - \mathbf{y}_2)\mathbf{D}_2\mathbf{M}, \quad (\mathbf{x}_1 - \mathbf{y}_1)\mathbf{D}_3\mathbf{M} \neq (\mathbf{x}_2 - \mathbf{y}_2)\mathbf{D}_1\mathbf{M}\}$ is the same (up to a permutation of indices of $\mathbf{x}$ and $\mathbf{y}$) as:

$$\forall i \in [\![1, n_I]\!], \quad (\mathbf{x}_1 - \mathbf{y}_1)(i) = 0 \text{ or } (\mathbf{x}_2 - \mathbf{y}_2)(i) = 0 \quad ; \quad \forall i \in [\![n_I+1, n/2]\!], \quad (\mathbf{x}_1 - \mathbf{y}_1)(i) = \pm(\mathbf{x}_2 - \mathbf{y}_2)(i)$$

Now by using the fact that distribution of a vector $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2)$ which is uniformly picked at random in S_w is the same as by multiplying some of its components by -1 or exchanging some of its component we have

$$\mathbb{P}(\mathcal{E}_1) \leq \mathbb{P}(\forall i \in [\![1, n_I]\!], (\mathbf{x}_1 - \mathbf{y}_1)(i) = 0, \quad \forall i \in [\![n_I + 1, n/2]\!], (\mathbf{x}_1 - \mathbf{y}_1)(i) = (\mathbf{x}_2 - \mathbf{y}_2)(i))$$

Let us now derive the following upper bound

$$\begin{aligned}
&\mathbb{P}(\forall i \in [\![1, n_I]\!], \quad (\mathbf{x}_1 - \mathbf{y}_1)(i) = 0, \quad \forall i \in [\![n_I + 1, n/2]\!], \quad (\mathbf{x}_1 - \mathbf{y}_1)(i) = (\mathbf{x}_2 - \mathbf{y}_2)(i)) \\
&\quad = \sum_{l=0}^{n_I} \mathbb{P}\left((\mathbf{x}_1 - \mathbf{y}_1)_{[\![1,n_I]\!]} = \mathbf{0}, \quad (\mathbf{x}_1 - \mathbf{x}_2)_{[\![n_I+1,n/2]\!]} = (\mathbf{y}_1 - \mathbf{y}_2)_{[\![n_I+1,n/2]\!]} \,\middle|\, |(\mathbf{x}_1)_{[\![1,n_I]\!]}| = l, |(\mathbf{y}_1)_{[\![1,n_I]\!]}| = l\right) \\
&\quad \mathbb{P}\left(|(\mathbf{x}_1)_{[\![1,n_I]\!]}| = l, |(\mathbf{y}_1)_{[\![1,n_I]\!]}| = l\right) \\
&\quad = \sum_{l=0}^{n-n_I} \left(\frac{\binom{n_I}{l} 2^l \binom{n-n_I}{w-l} 2^{w-l}}{\binom{n}{w} 2^w} \right)^2 \\
&\quad \mathbb{P}\left((\mathbf{x}_1 - \mathbf{y}_1)_{[\![1,n_I]\!]} = \mathbf{0}, \quad (\mathbf{x}_1 - \mathbf{x}_2)_{[\![n_I+1,n/2]\!]} = (\mathbf{y}_1 - \mathbf{y}_2)_{[\![n_I+1,n/2]\!]} \,\middle|\, |(\mathbf{x}_1)_{[\![1,n_I]\!]}| = l, |(\mathbf{y}_1)_{[\![1,n_I]\!]}| = l\right) \\
&\quad = \sum_{l=0}^{n-n_I} \left(\frac{\binom{n_I}{l} 2^l \binom{n-n_I}{w-l} 2^{w-l}}{\binom{n}{w} 2^w} \right)^2 \sum_{\mathbf{e}_1 \in \mathbb{F}_3^{n_I}, \mathbf{e}_2 \in \mathbb{F}_3^{n/2-n_I}} \mathbb{P}\left((\mathbf{x}_1)_{[\![1,n_I]\!]} = \mathbf{e}_1, \quad (\mathbf{x}_1 - \mathbf{x}_2)_{[\![n_I+1,n/2]\!]} = \mathbf{e}_2 \middle| |(\mathbf{x}_1)_{[\![1,n_I]\!]}| = l\right)^2 \\
&\quad = \sum_{l=0}^{n_I} \left(\frac{\binom{n_I}{l} 2^l \binom{n-n_I}{w-l} 2^{w-l}}{\binom{n}{w} 2^w} \right)^2 \sum_{\mathbf{e}_1 \in \mathbb{F}_3^{n_I} : |\mathbf{e}_1| = l} \sum_{j=0}^{n/2-n_I} \sum_{\mathbf{e}_2 \in \mathbb{F}_3^{n/2-n_I} : |\mathbf{e}_2| = j} \\
&\left(\mathbb{P}\left((\mathbf{x}_1)_{[\![1,n_I]\!]} = \mathbf{e}_1, \quad (\mathbf{x}_1 - \mathbf{x}_2)_{[\![n_I+1,n/2]\!]} = \mathbf{e}_2 \middle| |(\mathbf{x}_1)_{[\![1,n_I]\!]}| = l, |(\mathbf{x}_1 - \mathbf{x}_2)_{[\![n_I+1,n/2]\!]}| = j\right)\right. \\
&\quad \left. \mathbb{P}\left(|(\mathbf{x}_1 - \mathbf{x}_2)_{[\![n_I+1,n/2]\!]}| = j\right)\right)^2 \\
&\quad = \sum_{l=0}^{n_I} \left(\frac{\binom{n_I}{l} 2^l \binom{n-n_I}{w-l} 2^{w-l}}{\binom{n}{w} 2^w} \right)^2 \sum_{\mathbf{e}_1 \in \mathbb{F}_3^{n_I} : |\mathbf{e}_1| = l} \sum_{j=0}^{n/2-n_I} \sum_{\mathbf{e}_2 \in \mathbb{F}_3^{n/2-n_I} : |\mathbf{e}_2| = j} \\
&\quad \left(\frac{1}{2^l \binom{n_I}{l} 2^j \binom{n/2-n_I}{j}} \mathbb{P}\left(|(\mathbf{x}_1 - \mathbf{x}_2)_{[\![n_I+1,n/2]\!]}| = j\right) \right)^2 \\
&\quad = \sum_{l=0}^{n_I} \binom{n_I}{l} 2^l \left(\frac{\binom{n-n_I}{w-l} 2^{w-l}}{\binom{n}{w} 2^w} \right)^2 \sum_{j=0}^{n/2-n_I} \frac{1}{2^j \binom{n/2-n_I}{j}} \left(\mathbb{P}\left(|(\mathbf{x}_1 - \mathbf{x}_2)_{[\![n_I+1,n/2]\!]}| = j\right)\right)^2 \\
&\quad = \sum_{l=0}^{n_I} \binom{n_I}{l} 2^l \left(\frac{\binom{n-n_I}{w-l} 2^{w-l}}{\binom{n}{w} 2^w} \right)^2 \sum_{j=0}^{n/2-n_I} \frac{1}{2^j \binom{n/2-n_I}{j}} \\
&\quad \left(\frac{1}{\binom{n}{w} 2^w} \sum_{h=0}^{j} \sum_{p=0}^{n/2-n_I-j} \binom{n/2-n_I}{h} 2^h \binom{n/2-n_I-h}{j-h} 2^{2(j-h)} \binom{n/2-n_I-j}{p} 2^p \binom{2n_I}{w-j-h-2p} 2^{w-j-p-2h} \right)^2
\end{aligned}$$

which gives

$$\mathbb{P}\left(\mathcal{E}_1\right) \leq \sum_{l=0}^{n_I} \binom{n_I}{l} 2^l \left(\frac{\binom{n-n_I}{w-l} 2^{w-l}}{\binom{n}{w} 2^w} \right)^2 \sum_{j=0}^{n/2-n_I} \frac{1}{2^j \binom{n/2-n_I}{j}}$$
$$\left(\frac{1}{\binom{n}{w} 2^w} \sum_{h=0}^{j} \sum_{p=0}^{n/2-n_I-j} \binom{n/2-n_I}{h} \binom{n/2-n_I-h}{j-h} \binom{n/2-n_I-j}{p} \binom{2n_I}{w-j-h-2p} 2^{w-j-h-2p} \right)^2 \tag{60}$$

Therefore, with Equations (56),(57),(60) and (59) we finally obtain

$$\mathbb{P}_{\mathbf{H}_{\mathrm{pk}},\mathbf{x},\mathbf{y}} \left(\mathbf{x}\mathbf{H}_{\mathrm{pk}}{}^{\intercal} = \mathbf{y}\mathbf{H}_{\mathrm{pk}}{}^{\intercal} \right) \leq \frac{1}{3^{n-k}} (1+\varepsilon)$$

with

$$\varepsilon = \frac{3^{n-k}}{2^w \binom{n}{w}} + \sum_{j=0}^{w} \frac{3^{n/2-k_V}}{2^j \binom{n/2}{j} 2^{2w} \binom{n/2}{w}^2} \left(\sum_{\substack{p=0 \\ w+p \equiv 0 \mod 2}}^{j} \binom{j}{p} \binom{n/2}{\frac{w+p}{2}} \binom{\frac{w+p}{2}}{j} 2^{\frac{w+3p}{2}} \right)^2$$
$$+ 3^{n/2-k_U} \sum_{l=0}^{n_I} \binom{n_I}{l} 2^l \left(\frac{\binom{n-n_I}{w-l} 2^{w-l}}{\binom{n}{w} 2^w} \right)^2 \sum_{j=0}^{n/2-n_I} \frac{1}{2^j \binom{n/2-n_I}{j}}$$
$$\left(\frac{1}{\binom{n}{w} 2^w} \sum_{h=0}^{j} \sum_{p=0}^{n/2-n_I-j} \binom{n/2-n_I}{h} \binom{n/2-n_I-h}{j-h} \binom{n/2-n_I-j}{p} \binom{2n_I}{w-j-h-2p} 2^{w-j-h-2p} \right)^2 \tag{61}$$

which concludes the proof.

Lemmas 10 and 1 imply directly Proposition 3.

Proof (Proposition 3). Indeed we let in Lemma 1, $E \stackrel{\triangle}{=} \mathbb{F}_3^n$, $F \stackrel{\triangle}{=} \mathbb{F}_3^{n-k}$ and $\mathcal{H}$ be the set of functions associated to the 4-tuples $(\mathbf{H}_U, \mathbf{H}_V, \mathbf{S}, \mathbf{P})$ used to generate a public parity-check matrix $\mathbf{H}_{\mathrm{pk}}$ through (**??**). These functions h are given by $h(\mathbf{e}) = \mathbf{e}\mathbf{H}_{\mathrm{pk}}^{\intercal}$. Lemma 10 gives an upper-bound for the ε term in Lemma 1 and this finishes the proof of Proposition 3.

We are now able to prove Lemma 4 (we use here notations of the security proof in §6.3).

Lemma 4.

$$\mathbb{P}(S_1) \leq \mathbb{P}(S_2) + \frac{q_{\mathrm{hash}}}{2} \sqrt{\varepsilon} \text{ where } \varepsilon \text{ is given in Proposition 3.}$$

Proof (Lemma 4). To simplify notation we let $q \stackrel{\triangle}{=} q_{\mathrm{hash}}$. Then we notice that

$$\mathbb{P}(S_1) \leq \mathbb{P}(S_2) + \rho(\mathcal{D}_{w,q}^{\mathrm{pub}}, \mathcal{D}_{\mathrm{pub}} \otimes \mathcal{U}^{\otimes q}), \tag{62}$$

where

- $\mathcal{U}$ is the uniform distribution over $\mathbb{F}_2^{n-k}$;
- $\mathcal{D}_{w,q}^{\mathrm{pub}}$ is the distribution of the $(q+1)$-tuples $(\mathbf{H}_{\mathrm{pk}}, \mathbf{e}_1\mathbf{H}_{\mathrm{pk}}^{\intercal}, \cdots, \mathbf{e}_q\mathbf{H}_{\mathrm{pk}}^{\intercal})$ where the $\mathbf{e}_i$'s are independent and uniformly distributed in S_w;
- $\mathcal{D}_{\mathrm{pub}} \otimes \mathcal{U}^{\otimes q}$ is the distribution of the $(q+1)$-tuples $(\mathbf{H}_{\mathrm{pk}}, \mathbf{s}_1, \cdots, \mathbf{s}_q)$ where the $\mathbf{s}_i$'s are independent and uniformly distributed in $\mathbb{F}_2^{n-k}$.

We now observe that

$$\begin{aligned}\rho(\mathcal{D}^{\mathrm{pub}}_{w,q}, \mathcal{D}_{\mathrm{pub}} \otimes \mathcal{U}^{\otimes q}) &= \sum_{\mathbf{H}\in\mathbb{F}_2^{(n-k)\times n}} \mathbb{P}(\mathbf{H}_{\mathrm{pk}} = \mathbf{H})\rho((\mathcal{D}^{\mathbf{H}}_w)^{\otimes q}, \mathcal{U}^{\otimes q}) \\ &\leq q \sum_{\mathbf{H}\in\mathbb{F}_2^{(n-k)\times n}} \mathbb{P}(\mathbf{H}_{\mathrm{pk}} = \mathbf{H})\rho(\mathcal{D}^{\mathbf{H}}_w, \mathcal{U}) \text{ (by Prop. 4)} \\ &= q\mathbb{E}_{\mathbf{H}_{\mathrm{pk}}}\left\{\rho(\mathcal{D}^{\mathrm{pub}}_w, \mathcal{U})\right\} \\ &\leq q\frac{\sqrt{\varepsilon}}{2} \text{ (by Prop. 3).}\end{aligned}$$

C Proof of Proposition 7

Let us recall Proposition 7

Proposition 7. *Assume that we choose an admissible generalized $(U, U+V)$ code over $\mathbb{F}_3$ with a number n_I of linear combinations of type I by picking the parity-check matrices of U and V uniformly at random among the ternary matrices of size $(n/2-k_U)\times n/2$ and $(n/2-k_V)\times n/2$ respectively. Let $a_{(\mathbf{u},\mathbf{v})}(w)$, $a_{(\mathbf{u},\mathbf{0})}(w)$ and $a_{(\mathbf{0},\mathbf{v})}(w)$ be the expected number of codewords of weight w that are respectively in the admissible generalized $(U, U+V)$ code, of the form $(\mathbf{uD}_1, \mathbf{uD}_3)$ where $\mathbf{u}$ belongs to U and of the form $(\mathbf{vD}_2, \mathbf{vD}_4)$ where $\mathbf{v}$ belongs to V. These numbers are given for even w in $\{0,\ldots,n\}$ by*

$$a_{(\mathbf{u},\mathbf{0})}(w) = \frac{\binom{n/2}{w/2}2^{w/2}}{3^{n/2-k_U}} \quad ; \quad a_{(\mathbf{0},\mathbf{v})}(w) = \frac{1}{3^{n/2-k_V}} \sum_{\substack{j=0 \\ j \text{ even}}}^{w} \binom{n_I}{j}\binom{n/2-n_I}{\frac{w-j}{2}}2^{(w+j)/2}$$

$$a_{(\mathbf{u},\mathbf{v})}(w) = a_{(\mathbf{u},\mathbf{0})}(w) + a_{(\mathbf{0},\mathbf{v})}(w) + \frac{1}{3^{n-k_U-k_V}}\left(\binom{n}{w}2^w - \binom{n/2}{w/2}2^{w/2} - \sum_{\substack{j=0 \\ j \text{ even}}}^{w} \binom{n_I}{j}\binom{n/2-n_I}{\frac{w-j}{2}}2^{(w+j)/2}\right)$$

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

$$a_{(\mathbf{u},\mathbf{0})}(w) = 0 \quad ; \quad a_{(\mathbf{0},\mathbf{v})}(w) = \frac{1}{3^{n/2-k_V}} \sum_{\substack{j=0 \\ j \text{ odd}}}^{w} \binom{n_I}{j}\binom{n/2-n_I}{\frac{w-j}{2}}2^{(w+j)/2}$$

$$a_{(\mathbf{u},\mathbf{v})}(w) \quad = \quad a_{(\mathbf{0},\mathbf{v})}(w) \quad + \quad \frac{1}{3^{n-k_U-k_V}}\left(\binom{n}{w}2^w - \sum_{\substack{j=0 \\ j \text{ odd}}}^{w} \binom{n_I}{j}\binom{n/2-n_I}{\frac{w-j}{2}}2^{(w+j)/2}\right)$$

On the other hand, when we choose a linear code of length n over $\mathbb{F}_3$ 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

$$a(w) = \frac{\binom{n}{w}2^w}{3^{n-k_U-k_V}}.$$

Lemma 9 in Appendix §B will be useful four the proof. The last part of Proposition 7 is a direct application of this lemma. We namely have

Proposition 12. *Let $a(w)$ be the expected number of codewords of weight w in a ternary linear code $\mathcal{C}$ 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*

$$a(w) = \frac{\binom{n}{w}}{3^r}.$$

Proof. Let $Z \stackrel{\triangle}{=} \sum_{\mathbf{x} \in \mathbb{F}_3^n : |\mathbf{x}| = w} Z_{\mathbf{x}}$ where $Z_{\mathbf{x}}$ is the indicator function of the event "$\mathbf{x}$ is in $\mathcal{C}$". We have

$$\begin{aligned}
a(w) &= \mathbb{E}(Z) \\
&= \sum_{\mathbf{x} \in \mathbb{F}_3^n : |\mathbf{x}| = w} \mathbb{E}(Z_{\mathbf{x}}) \\
&= \sum_{\mathbf{x} \in \mathbb{F}_3^n : |\mathbf{x}| = w} \mathbb{P}(\mathbf{x} \in \mathcal{C}) \\
&= \sum_{\mathbf{x} \in \mathbb{F}_3^n : |\mathbf{x}| = w} \mathbb{P}(\mathbf{x}\mathbf{H}^\intercal = 0) \\
&= \sum_{\mathbf{x} \in \mathbb{F}_3^n : |\mathbf{x}| = w} \frac{1}{3^r} \\
&= \frac{\binom{n}{w}}{3^r}.
\end{aligned}$$

This proves the part of Proposition 7 dealing with the expected weight distribution of a random linear code. We are ready now to prove Proposition 7 concerning the expected weight distribution of a random $(U, U+V)$ code.

Weight distributions of $(U\mathbf{D}_1, U\mathbf{D}_3) \stackrel{\triangle}{=} \{(\mathbf{u}\mathbf{D}_1, \mathbf{u}\mathbf{D}_3) : \mathbf{u} \in U\}$ and $(\mathbf{D}_2 V, \mathbf{D}_4 V) \stackrel{\triangle}{=} \{(\mathbf{v}\mathbf{D}_2, \mathbf{v}\mathbf{D}_4) : \mathbf{v} \in V\}$. Let us recall the $(\mathbf{D}_1 U + \mathbf{D}_2 V, \mathbf{D}_3 U + \mathbf{D}_4)$ is an admissible generalized code which enforces that

$$\forall i \in [\![1, n/2]\!], \quad \mathbf{D}_1(i,i)\mathbf{D}_3(i,i) \neq 0$$

and therefore it follows directly from Proposition 12 since $a_{(\mathbf{u},\mathbf{0})}(w) = 0$ for odd and $a_{(\mathbf{u},\mathbf{0})}(w)$ 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, the weight distribution of $(\mathbf{D}_2\mathbf{v}, \mathbf{D}_4\mathbf{v})$ for $\mathbf{v} \in V$ is little more sophisticate. Let us recall the following definition:

Definition 3. (number of V blocks of type I). *In a generalized $(U, U+V)$ code of length n associated to the 4-tuple of diagonal matrices of size $n/2$ $(\mathbf{D}_1, \mathbf{D}_2, \mathbf{D}_3, \mathbf{D}_4)$, the number of V blocks of type I, which we denote by n_I, is defined by:*

$$n_I \stackrel{\triangle}{=} |\{1 \leq i \leq n/2 : \mathbf{D}_2(i,i)\mathbf{D}_4(i,i) = 0\}|.$$

where from the definition of generalized $(U, U+V)$ when either $\mathbf{D}_2(i,i) = 0$ or $\mathbf{D}_4(i,i) = 0$, the other one is necessarily different from 0. In this way, $a_{(\mathbf{0},\mathbf{v})}(w)$ is equal to the expected number of weight $j + \frac{w-j}{2}$ for all j in $[\![1, n_I]\!]$ in a random linear code of length $n/2$ where j positions correspond to the n_I positions which gives the number of block of type I and $\frac{w-j}{2}$ for the others as there are involved in components which count twice in the weight. Furthermore this code has a parity-check matrix of size $(n/2 - k_V) \times n/2$. This easily gives from Proposition 12:

$$a_{(\mathbf{0},\mathbf{v})} = \frac{1}{3^{n/2-k_V}} \sum_{j=0}^{n_I} \binom{n_I}{j} \binom{n/2 - n_I}{\frac{w-j}{2}} 2^{j + \frac{w-j}{2}}$$

Weight distributions of $(U\mathbf{D}_1 + V\mathbf{D}_2, U\mathbf{D}_3 + V\mathbf{D}4)$. The admissible generalized $(U, U+V)$-code is chosen randomly by picking up a parity-check matrix $\mathbf{H}_U$ of U uniformly at random among

the set of $(n/2-k_U)\times n/2$ ternary matrices and a parity-check matrix $\mathbf{H}_V$ of V uniformly at random among the set of $(n/2-k_V)\times n/2$ ternary matrices. Let $Z \stackrel{\triangle}{=} \sum_{\mathbf{x}\in\mathbb{F}_3^n:|\mathbf{x}|=w} Z_{\mathbf{x}}$ where $Z_{\mathbf{x}}$ is the indicator function of the event "$\mathbf{x}$ is in $(U\mathbf{D}_1+V\mathbf{D}_2, U\mathbf{D}_3+V\mathbf{D}_4)$".

We have

$$\begin{aligned} a_{(\mathbf{u},\mathbf{v})}(w) &= \mathbb{E}(Z) \\ &= \sum_{\mathbf{x}\in\mathbb{F}_3^n:|\mathbf{x}|=w} \mathbb{E}(Z_{\mathbf{x}}) \\ &= \sum_{\mathbf{x}\in\mathbb{F}_3^n:|\mathbf{x}|=w} \mathbb{P}(Z_{\mathbf{x}}=1) \\ &= \sum_{\mathbf{x}\in\mathbb{F}_3^n:|\mathbf{x}|=w} \mathbb{P}(\mathbf{x}\in(U\mathbf{D}_1+V\mathbf{D}_2, U\mathbf{D}_3+V\mathbf{D}_4)) \end{aligned} \tag{63}$$

Let us recall now that a parity-check matrix of the code $(U\mathbf{D}_1+V\mathbf{D}_2, U\mathbf{D}_3+V\mathbf{D}_4)$ is:

$$\begin{pmatrix} \mathbf{H}_U\mathbf{D}_4\mathbf{M} & -\mathbf{H}_U\mathbf{D}_2\mathbf{M} \\ \mathbf{H}_V\mathbf{D}_3\mathbf{M} & -\mathbf{H}_V\mathbf{D}_1\mathbf{M} \end{pmatrix}$$

where $\mathbf{M}\stackrel{\triangle}{=}(\mathbf{D}_1\mathbf{D}_4-\mathbf{D}_3\mathbf{D}_2)^{-1}$ is a diagonal invertible matrix. Therefore, by writing $\mathbf{x}=(\mathbf{x}_1,\mathbf{x}_2)$ where $\mathbf{x}_i$ is in $\mathbb{F}_3^{n/2}$ we know that $\mathbf{x}$ is in $(\mathbf{D}_1U+\mathbf{D}_2, \mathbf{D}_3U+\mathbf{D}_4V)$ if and only if at the same time:

$$\mathbf{x}_1\mathbf{D}_4{\mathbf{H}_U}^{\mathsf{T}} = \mathbf{x}_2\mathbf{D}_2{\mathbf{H}_U}^{\mathsf{T}},\ \ \mathbf{x}_1\mathbf{D}_3{\mathbf{H}_V}^{\mathsf{T}} = \mathbf{x}_2\mathbf{D}_1{\mathbf{H}_V}^{\mathsf{T}}.$$

There are three disjoint cases (see $\mathbf{D}=\begin{pmatrix}\mathbf{D}_1 & \mathbf{D}_2\\ \mathbf{D}_3 & \mathbf{D}_4\end{pmatrix}$ is invertible) to consider

Case 1: $\mathbf{x}_1\mathbf{D}_4=\mathbf{x}_2\mathbf{D}_2$. In this case

$$\mathbb{P}(\mathbf{x}\in(U\mathbf{D}_1+V\mathbf{D}_2, U\mathbf{D}_3+V\mathbf{D}_4)) = \mathbb{P}((\mathbf{x}_1\mathbf{D}_3-\mathbf{x}_2\mathbf{D}_1){\mathbf{H}_V}^{\mathsf{T}}=\mathbf{0}) = \frac{1}{3^{n/2-k_V}} \tag{64}$$

Case 2: $\mathbf{x}_1\mathbf{D}_4{\mathbf{H}_U}^{\mathsf{T}} = \mathbf{x}_2\mathbf{D}_2{\mathbf{H}_U}^{\mathsf{T}}$. In this case

$$\mathbb{P}(\mathbf{x}\in(U\mathbf{D}_1+V\mathbf{D}_2, U\mathbf{D}_3+V\mathbf{D}_4)) = \mathbb{P}((\mathbf{x}_1\mathbf{D}_4-\mathbf{x}_2\mathbf{D}_2){\mathbf{H}_U}^{\mathsf{T}}=\mathbf{0}) = \frac{1}{3^{n/2-k_U}} \tag{65}$$

Case 3: $\mathbf{x}_1\mathbf{D}_4\neq\mathbf{x}_2\mathbf{D}_2$ and $\mathbf{x}_1\mathbf{D}_4{\mathbf{H}_U}^{\mathsf{T}}\neq\mathbf{x}_2\mathbf{D}_2{\mathbf{H}_U}^{\mathsf{T}}$. In this case

$$\begin{aligned} &\mathbb{P}(\mathbf{x}\in(U\mathbf{D}_1+V\mathbf{D}_2, U\mathbf{D}_3+V\mathbf{D}_4)) = \\ &\qquad \mathbb{P}((\mathbf{x}_1\mathbf{D}_3-\mathbf{x}_2\mathbf{D}_1){\mathbf{H}_V}^{\mathsf{T}}=\mathbf{0}, (\mathbf{x}_1\mathbf{D}_4-\mathbf{x}_2\mathbf{D}_2){\mathbf{H}_U}^{\mathsf{T}}=\mathbf{0}) = \frac{1}{3^{n/2-k_U}}\frac{1}{3^{n/2-k_V}} \end{aligned} \tag{66}$$

Note that we used in each case Lemma 9.

By substituting $\mathbb{P}(\mathbf{x}\in(U\mathbf{D}_1+V\mathbf{D}_2, U\mathbf{D}_3+V\mathbf{D}_4))$ in (63) and using definition of number of blocks of type I (Definition 3) we obtain for even $0<w\leq n$

$$\begin{aligned} a_{(\mathbf{u},\mathbf{v})}(w) = a_{(\mathbf{u},\mathbf{v})}(w) = a_{(\mathbf{u},\mathbf{0})}(w) + a_{(\mathbf{0},\mathbf{v})}(w) + \\ \frac{1}{3^{n-k_U-k_V}}\left(\binom{n}{w}2^w - \binom{n/2}{w/2}2^{w/2} - \sum_{\substack{j=0\\ j \text{ even}}}^{w}\binom{n_I}{j}\binom{n/2-n_I}{\frac{w-j}{2}}2^{(w+j)/2}\right) \end{aligned}$$

and for odd $w\leq n$

$$a_{(\mathbf{u},\mathbf{v})}(w) \quad = \quad a_{(\mathbf{0},\mathbf{v})}(w) \quad + \quad \frac{1}{3^{n-k_U-k_V}}\left(\binom{n}{w}2^w - \sum_{\substack{j=0\\ j \text{ odd}}}^{w}\binom{n_I}{j}\binom{n/2-n_I}{\frac{w-j}{2}}2^{(w+j)/2}\right)$$

which concludes the proof.