Proving modern code-based dual attacks with second-order techniques

Charles Meyer-Hilfiger

Univ Rennes, Inria, CNRS, IRISA, Rennes, France
charles.meyer-hilfiger@inria.fr

Abstract. In code-based cryptography, dual attacks for solving the decoding problem have recently been improved. They are now competitive and beat information set decoders for a significant regime. These recent dual attacks, starting from Carrier et al. (Asiacrypt 2022), work by reducing decoding to an LPN problem where the secret and the noise involve parts of the error vector coming from the decoding problem. However, currently, the analysis of all these dual attacks is heuristic. In the original Asiacrypt 2022 work, a simple LPN modeling was used to carry out the analysis but Meyer-Hilfiger and Tillich (TCC 2023) showed that this assumption could not be used. Consequently, they proposed an alternative analysis based on Fourier theory and on heuristically modeling the weight enumerator of a random linear code as a Poisson variable. The analysis of the newest and most efficient dual attack, doubleRLPN, introduced by Carrier et al. (Eurocrypt 2024) also relies on this technique and on this model.

Our main contribution is to devise a variant of doubleRLPN that we can fully prove without using any model. We show that our variant has the same performance, up to polynomial factors, as the original doubleRLPN algorithm. The final algorithm and its analysis are also simpler. Our technique involves flipping the coordinates of the noisy codeword and observing the fine changes in the amount of noise in the related LPN problem to reconstruct the entire error. The analysis is based on the second-order behavior of the bias of the noise which was already used in the original analysis.

Secondly, the performance of our algorithm, as it was the case for doubleRLPN, heavily depends on having access to a good code along with an efficient decoder. We instantiate this code by choosing a Cartesian product of a constant (instead of sublinear in the original proposal by Carrier et al.) number of random linear codes. We use a decoder based on blockwise error enumeration that was already used by Guo et al. (Asiacrypt 2014). We show that our approach is optimal up to polynomial (instead of superpolynomial) factors.

1 Introduction

The security of code-based schemes, see for example HQC [AAB+22b] a to be NIST standard or [AAB+22a, ABC+22, AMBB+24], relies on the computational hardness of the Binary Decoding Problem.

Definition 1.1 (Binary Decoding Problem). *Let $\mathcal{C}$ be a binary linear code of dimension k and length n,* i.e. *a linear subspace of dimension k of $\mathbb{F}_2^n$. Given a basis of $\mathcal{C}$ and a noisy codeword $\mathbf{y} = \mathbf{c} + \mathbf{e}$ where $\mathbf{c} \in \mathcal{C}$ and $\mathbf{e}$ has Hamming weight $|\mathbf{e}| = t$, the goal is to find an error $\mathbf{e}'$ of weight t such that $\mathbf{y} - \mathbf{e} \in \mathcal{C}$.*

For appropriate choices of the parameters n, k, t, the complexity of all algorithms solving this problem is exponential in t. In particular, when the code rate $R \stackrel{\text{def}}{=} k/n$ is constant and the relative decoding weight $\tau = t/n$ is a well-chosen constant, the runtime of all algorithms is in $O\left(2^{\alpha(R,\tau)n}\right)$ where the constant $\alpha(R,\tau)$ depends on the algorithm.

The two main families of algorithms for solving this problem are, on the one hand, Information Set Decoders (ISDs) and, on the other hand, Dual Attacks. ISDs are essentially improvements of Prange's algorithm from 1962 [Pra62]. They are the most widely studied decoders and have benefited from many improvements over the years : [Ste88, Dum89, MMT11, BJMM12, MO15, BM18] to cite a few. In contrast, Dual Attacks can be seen as improvements of Al-Jabri's statistical decoding algorithm from 2001 [Jab01]. However, this decoder was forgotten for a long time because it was shown to be uncompetitive for attacking the McEliece cryptosystem [Ove06] and asymptotically uncompetitive [DT17] compared to Prange's decoder, the simplest of the ISDs. Recently, however, new Dual Attacks have been developed [CDMT22, MT23, CDMT24] that are now competitive and even outperform the ISDs for significant regimes. More precisely, the best Dual Attack, doubleRLPN [CDMT24], asymptotically outperforms the best ISD [BM18] in terms of time complexity when decoding codes of constant rate $R \stackrel{\text{def}}{=} k/n$ smaller than 0.42 at the Gilbert-Varshamov distance. The Gilbert-Varshamov distance is the distance at which we expect a unique non-planted solution to the decoding problem and is also the distance at which the problem is the hardest. More precisely, it is defined as the distance t such that $t/n = h_2^{-1}(1-R)(1+o(1))$ where $h_2(x) = -x\log_2(x) - (1-x)\log_2(1-x)$ is the binary entropy function.

1.1 Dual Attacks

The main ingredient for dual attacks is the dual code, defined as $\mathcal{C}^\perp \stackrel{\text{def}}{=} \{\mathbf{h} \in \mathbb{F}_2^n : \langle \mathbf{h}, \mathbf{c}\rangle = 0 \ \forall \mathbf{c} \in \mathcal{C}\}$ where $\langle \mathbf{h}, \mathbf{c}\rangle = \sum_{i=0}^n h_i c_i \in \mathbb{F}_2$ and $h_i \in \mathbb{F}_2^n$ is the i'th coordinate of $\mathbf{h}$. To decode, dual attacks leverage the fact that, for a dual vector $\mathbf{h} \in \mathcal{C}^\perp$, the inner product between $\mathbf{h}$ and the noisy codeword $\mathbf{y} = \mathbf{c} + \mathbf{e}$,

$$\langle \mathbf{y}, \mathbf{h}\rangle = \langle \mathbf{c} + \mathbf{e}, \mathbf{h}\rangle = \langle \mathbf{e}, \mathbf{h}\rangle$$

is more biased toward 0 as the Hamming weights of $\mathbf{e}$ and $\mathbf{h}$ decrease. The Hamming weight of a vector $\mathbf{e}$ is the number of non-zero coordinates in $\mathbf{e}$ and is denoted by $|\mathbf{e}|$.

1.2 Modern dual attacks.

The most recent dual attacks [CDMT22, MT23, CDMT24] essentially reduce decoding to an LPN problem, which is then solved with standard solvers.

LPN problem. An LPN problem is a problem where given access to an oracle which upon each call returns, $(\mathbf{a}, \langle \mathbf{a}, \mathbf{s} \rangle + e) \in \mathbb{F}_2^s \times \mathbb{F}_2$ where $\mathbf{a}$ is uniformly random in $\mathbb{F}_2^s$ and the noise e is taken as a Bernoulli with variable with bias $\varepsilon = \mathrm{bias}\,(e)$, where

$$\mathrm{bias}\,(X) \stackrel{\mathrm{def}}{=} \mathbb{P}\,(X = 0) - \mathbb{P}\,(X = 1)$$

and $\mathbf{s}$ is a fixed secret, the goal is to recover the secret $\mathbf{s}$ with as many calls to the oracle as wanted. To that extent, Shannon's first theorem states that it is sufficient that the number N of samples is of order $N \geqslant s/\varepsilon^2$ to be able to recover the secret $\mathbf{s}$ by finding the $\mathbf{x}$ maximizing of a certain score function (a.k.a maximum likelihood decoding) $F_{\mathcal{L}}$ defined as

$$F_{\mathcal{L}}\,(\mathbf{x}) \stackrel{\mathrm{def}}{=} \sum_{(\mathbf{a},b)\in\mathcal{L}} (-1)^{b-\langle \mathbf{a},\mathbf{x}\rangle} \qquad\qquad \forall \mathbf{x} \in \mathbb{F}_2^s$$

where $\mathcal{L}$ is a list of N queried LPN samples.

Reducing decoding to LPN. The reduction presented in [CDMT22] works as follows : by splitting the support $[\![1,\, n]\!]$ in two complementary parts $\mathscr{P}$ and $\mathscr{N}$ and by computing a dual vector $\mathbf{h}$ of low weight on the part $\mathscr{N}$, we get an LPN sample by computing the inner product of $\mathbf{h}$ and the noisy codeword $\mathbf{y} = \mathbf{c} + \mathbf{e}$:

$$\langle \mathbf{y}, \mathbf{h} \rangle = \langle \mathbf{a}, \mathbf{s} \rangle + e \quad \text{ where } \quad \begin{cases} \mathbf{s} & = \mathbf{e}_{\mathscr{P}} \\ \mathbf{a} & = \mathbf{h}_{\mathscr{P}} \\ e & = \langle \mathbf{e}_{\mathscr{N}}, \mathbf{h}_{\mathscr{N}} \rangle \end{cases}$$

where $\mathbf{e}_{\mathscr{P}} \in \mathbb{F}_2^{|\mathscr{P}|}$ (*i.e.* the vector $\mathbf{e}$ restricted to the coordinates given by $\mathscr{P}$) is the secret. The strategy is thus to compute many low-weight dual vectors on $\mathscr{N}$, each of them yielding an LPN sample $(\mathbf{h}_{\mathscr{P}},\ \langle \mathbf{y}, \mathbf{h} \rangle)$. The LPN problem is then solved using standard solvers. In RLPN this is done by computing *exhaustively* the score function for each entry $\mathbf{x} \in \mathbb{F}_2^{|\mathscr{P}|}$ and returning the $\mathbf{x}$ maximizing it.

Estimating the bias of the noise. Say, for the sake of the discussion, that $\mathbf{h}$ is taken uniformly at random in $\{\, \mathbf{h} \in \mathcal{C}^{\perp} \,:\, |\mathbf{h}_{\mathscr{N}}| = w\}$. It was shown in [CDMT22, Proposition 3.1] that, under some natural condition, $\mathrm{bias}\,(e)$ could be well be approximated by $\delta\,(|\mathbf{e}_{\mathscr{N}}|)$, the bias by forgetting about the code structure, namely

$$\delta\,(|\mathbf{e}_{\mathscr{N}}|) \stackrel{\mathrm{def}}{=} \mathrm{bias}\,(\langle \mathbf{e}_{\mathscr{N}}, \mathbf{g}_{\mathscr{N}} \rangle)$$

where $\mathbf{g}_{\mathscr{N}}$ is taken uniformly at random among weight of Hamming weight w in $\mathbb{F}_2^{|\mathscr{N}|}$. This is very convenient, as the latter bias can be expressed as a closed-form function of only the vector weights and lengths. This function involves Krawtchouk polynomials and is essentially decreasing as the weights of the vectors increase.

A first analysis with an LPN modelling. However, as noted in this original paper, the obtained samples *are not distributed as standard LPN samples*. This is due to the fact that the noises between each sample depend on each other because of common positions in the dual vectors $\mathbf{h}_{\mathscr{N}}$. To make the analysis tractable [CDMT22] used a simple *LPN modelling*

in which the variables were assumed to be independent.This basically allowed reusing the aforementioned information theoretic result stating that if the number N of LPN samples is such that

$$N \geqslant \frac{1}{\delta \left(|\mathbf{e}_{\mathscr{N}}| \right)^2}$$

then maximizing the score function would yield the secret $\mathbf{e}_{\mathscr{P}}$ with overwhelming probability. Some experimental discrepancies were noted, but it was conjectured that they were not problematic asymptotically.

The rise of the Poisson model. However, as shown shortly after by [MT23], this LPN modelling could not be used. This comes from the fact that the $\mathbf{h}_{\mathscr{N}}$ intervening in the noise have common positions. [MT23] showed that there exists some $\mathbf{x} \neq \mathbf{e}_{\mathscr{P}}$ such that their associated score function is bigger than the score of the secret $\mathbf{e}_{\mathscr{P}}$. This critically means that $\mathbf{e}_{\mathscr{P}}$ cannot be recovered by maximizing the score function, as was believed before.

To contravene this, [MT23] proposed a corrected variant of RLPN. The idea was, instead of returning the vector maximizing the score function, to consider a small set of potential candidates for $\mathbf{e}_{\mathscr{P}}$. A vector $\mathbf{x}$ would be considered as a candidate for $\mathbf{e}_{\mathscr{P}}$ if its associated score, $F_{\mathcal{L}}(\mathbf{x})$, was big enough, superior to a well-chosen threshold.

The algorithm would then test each candidate $\mathbf{x}$ for $\mathbf{e}_{\mathscr{P}}$ by solving a smaller decoding problem. This decoding problem would return the whole error $\mathbf{e}$ when $\mathbf{x} = \mathbf{e}_{\mathscr{P}}$ and fail otherwise. In particular, and importantly, testing a candidate is exponentially costly. As such, a key part of the analysis was to precisely estimate their number. This all boils down to understanding the tail distribution of the score function : given a fixed $\mathbf{x} \neq \mathbf{e}_{\mathscr{P}}$, what is the probability that $F_{\mathscr{H},\mathbf{y}}(\mathbf{x})$ is superior to the threshold that was chosen? Considering that the optimal parameters for RLPN in the regime of interests are such that $|\mathscr{P}| = \Omega(n)$ and as a result the space in which $\mathbf{x} \in \mathbb{F}_2^{|\mathscr{P}|}$ leaves is exponentially big, thus one would need exponential bounds, *i.e.* bounds that holds with probability $1 - 2^{-\Omega(n)}$ in order to say anything non-trivial about the size of the set of candidates. Sadly, [MT23] had to rely on some assumptions to obtain such an exponential bound.

Their technique relied on using the Poisson summation formula on the score function along with the model that when $\mathcal{C} + \mathbf{z}$ is a random coset of a random linear code of length n and dimension k then the number of codewords of weight i can be modeled as a Poisson variable of right expected value, $\binom{n}{i}/2^{n-k}$. This model is surprisingly accurate.

Under such a model, and equipped with this bound, they were able to show that the number of false candidates was polynomial and that consequently the cost of checking the candidates was completely dominated by the other costs of the algorithm (say by the part where we compute the dual vectors). All in all, this showed that even though the LPN model was not valid, the overall additional cost incurred by this structure was of polynomial nature. Later, [CDMT24] improved this RLPN algorithm by using an additional reduction from sparse LPN to plain LPN, and its analysis relied on a similar estimation of the number of false candidates. This estimation relied on a similar Poisson model, and it was shown that for the parameters of interest in the article, this number was exponential. However, again, it was sufficiently low so that the cost of checking these candidates never dominated the complexity of the other steps of the algorithm.

Dual attacks in lattice-based cryptography. The security of lattice-based cryptography relies on the hardness of the LWE problem. The first dual attack to solve LWE was

introduced by [AR05]. Similarly to coding theory, it is based on the computation of many short vectors in a related lattice. These attacks were improved successively, first [BDGL16] proposed to use the output list of a sieving algorithm to compute many short dual vectors in batch, then [Alb17, EJK20] used a splitting strategy that was then improved by [GJ21, MAT22], these two last work claimed to reduce the security of the NIST standard Kyber. However, [DP23] showed that the key independence assumptions (that can be seen somewhat as the LPN modelling in code-based cryptography) used in the analysis of these attacks were flawed. [DP23] showed that this led the score function, *i.e.* the key quantity underlying dual attacks, to be badly predicted. This started a controversy by casting a very serious doubt as to whether these recent dual attacks could really work as expected. Then, [PS24] used Fourier theory to fully prove a non-competitive variant of these attacks. Later, two concurrent works, one [CDMT24, Section 4], and the other, more in-depth [DP26], exhibited a model that accurately predicted the tail behaviour of the score function for a simpler dual attack. Then [CMST25] devised a variantof Matzov dual attack [MAT22] and analyzed it by generalizing the previous model. This work eventually showed that a competitive lattice-based dual attack could work as expected, as it surpassed Matzov attack in some regimes and claimed, in a certain computation model, to dent the security of Kyber. All in all, all these works analyze these lattice-based dual attacks by essentially understanding the fine-tail behaviour of the score function.

1.3 Contributions

- Our main contribution is to devise a variant of double-RLPN that we can fully prove without using any model whatsoever up to rate $R \leqslant 0.5$. We show that our variant has the same performance, up to polynomial factors, as the original double-RLPN algorithm. In practice, our result holds for higher rates, but then the result depends on the parameters of the algorithm. In any case, $R \leqslant 0.5$ already encompasses the rate regime where the best dual attacks are currently known to beat the best ISDs. Our technique departs from current techniques in code and lattice that reduce the analysis to understanding the tail behaviour of the score function. Here, our analysis only relies on the known rough second-order behaviour of the score function.
- Secondly, as we will recall later, the performance of our algorithm, as was the case for the doubleRLPN, heavily depends on having access to a good code along with an efficient decoder. We instantiate this code by choosing Cartesian product of a constant (instead of sublinear in the original proposal of [CDMT24]) number of random linear codes. We use a syndrome decoder based on blockwise error enumeration that was already used in [GJL14]. We show overall that our algorithm, when using this code, loses only a polynomial factor (instead of superpolynomial) compared to the ideal case where we would suppose that we have access to a random linear code that we could decode efficiently.

We will also quickly explain the similarities between lattice-based dual attack and code-based dual attack to argue why our approach could be adapted to prove lattice-based attacks such as [CMST25] without using any model.

1.4 Our technique

We give the idea of our provable variant by focusing on the simplest case of RLPN in this introduction.

The tool. Our goal is to make a variant of the attack whose proof relies only on known second-order behaviour on the score function that we give as follows (this is only a slight variation of [CDMT22, Proposition 3.1]).

Lemma 1.2 (Second-order behavior of the score function). *For any $k, t, w, n \in \mathbb{N}$ and any subset $\mathscr{N} \subset [\![1,\, n]\!]$ and any vector $\mathbf{e} \in \mathbb{F}_2^n$ we have with probability lower bounded by $1 - 1/n^2$ that*

$$F\left(\mathbf{e}_{\mathscr{P}}\right) \in \left[N\delta\left(\mathbf{e}_{\mathscr{N}}\right) \pm n^2\sqrt{N}\right]$$

where i) $\mathcal{C}$ is taken uniformly at random among binary linear codes of dimension k and length n that are of full rank on the positions given by the complementary of $\mathscr{N}$ and ii) $F\left(\mathbf{e}_{\mathscr{P}}\right) \stackrel{def}{=} \sum_{\mathbf{h}\in\mathscr{H}}(-1)^{\langle \mathbf{e}_{\mathscr{N}}, \mathbf{h}_{\mathscr{N}}\rangle}$ is the score function evaluated on the secret and $\mathscr{H} \stackrel{def}{=} \{\mathbf{h} \in \mathcal{C}^{\perp} : |\mathbf{h}_{\mathscr{N}}| = w \}$ is the set of dual vectors and where $N \stackrel{def}{=} \mathbb{E}\left(|\mathscr{H}|\right)$ and $\delta\left(|\mathbf{e}_{\mathscr{N}}|\right) \stackrel{def}{=} \mathrm{bias}\left(\langle \mathbf{e}_{\mathscr{N}}, \mathbf{g}_{\mathscr{N}}\rangle\right)$ where $\mathbf{g}_{\mathscr{N}}$ is taken uniformly at random in $\{\mathbf{r} \in \mathbb{F}_2^n \ : |\mathbf{r}| = w \}$.

This lemma is a direct application of Byeneme-Chebyshev inequality along with the fact that N is an upper bound on the variance of $F\left(\mathbf{e}_{\mathscr{P}}\right)$ and $N\delta\left(|\mathbf{e}_{\mathscr{N}}|\right)$ is the expected value of $F\left(\mathbf{e}_{\mathscr{P}}\right)$ with these quantities being functions of the parameters k, t, w, n and the weight of $\mathbf{e}_{\mathscr{N}}$ only. A similar proposition could be derived to show that for any fixed $\mathbf{x} \neq \mathbf{e}_{\mathscr{P}}$ we have with probability $1 - 1/n^2$ that $F\left(\mathbf{x}\right) = 0 + n^2\sqrt{N}$. This would be sufficient to show that one can distinguish the secret $\mathbf{e}_{\mathscr{P}}$ from a fixed $\mathbf{x} \neq \mathbf{e}_{\mathscr{P}}$ under the natural condition on the parameter that $N > 1/\delta\left(|\mathbf{e}_{\mathscr{N}}|\right)^2$. However, it would be completely insufficient to say that we could distinguish it from *all* of the $\mathbf{x}$'s.

The technique On a high level, we do as follows : instead of computing one LPN problem and an associated score function, we compute $|\mathscr{N}|+1$ LPN problems and an associated score function. This will allow us to make for each $\mathbf{x} \in \mathbb{F}_2^{|\mathscr{P}|}$ a guess $g\left(\mathbf{x}\right)$ for the value of $\mathbf{e}_{\mathscr{N}}$, in *polynomial* time. The two crucial point will be that i) a guess can be tested in *polynomial* and ii) we can show that on the secret $\mathbf{x} = \mathbf{e}_{\mathscr{P}}$ the guess is valid, namely $g\left(\mathbf{e}_{\mathscr{P}}\right) = \mathbf{e}_{\mathscr{N}}$ with good probability.

Base observation. Our base observation is that we can change the amount of noise of the LPN samples by flipping the coordinates of the received noisy codeword $\mathbf{y}$ that intervene in the noise of the LPN samples. More precisely, defining ξ_i as the null vector expect for its i'th coordinate, we can compute $\mathbf{y}^{[i]}$ as

$$\left(\mathbf{y}^{[i]}\right)_{\mathscr{P}} = \mathbf{y}_{\mathscr{P}} \quad \text{and} \quad \left(\mathbf{y}^{[i]}\right)_{\mathscr{N}} = \mathbf{y}_{\mathscr{N}} + \xi_i.$$

One can readily see that for a dual vector $\mathbf{h} \in \mathcal{C}$ of low weight on $\mathscr{N}$ we have the "flipped" LPN sample

$$\left\langle \mathbf{y}^{[i]}, \mathbf{h} \right\rangle = \langle \mathbf{e}_{\mathscr{P}}, \mathbf{h}_{\mathscr{N}}\rangle + \langle \mathbf{e}_{\mathscr{N}} + \xi_i, \mathbf{h}_{\mathscr{N}}\rangle$$

which is more or less noisy depending on the weight of $\mathbf{e}_{\mathscr{N}} + \xi_i$. Notably, denoting by $F_{\mathcal{L}(\mathscr{H},\mathbf{y})}$ the score function associated to the samples given by $\mathcal{L}\left(\mathscr{H}, \mathbf{y}\right) \stackrel{\text{def}}{=} \left(\left(\mathbf{h}_{\mathscr{P}}, \langle \mathbf{y}, \mathbf{h}\rangle\right)\right)_{\mathbf{h}\in\mathscr{H}}$ we have that

$$F_{\mathcal{L}(\mathscr{H},\mathbf{y})}(\mathbf{e}_{\mathscr{P}}) = \sum_{\mathbf{h}\in\mathscr{H}} (-1)^{\langle \mathbf{e}_{\mathscr{N}}, \mathbf{h}_{\mathscr{N}}\rangle},$$
$$F_{\mathcal{L}\left(\mathscr{H},\mathbf{y}^{[i]}\right)}(\mathbf{e}_{\mathscr{P}}) = \sum_{\mathbf{h}\in\mathscr{H}} (-1)^{\langle \mathbf{e}_{\mathscr{N}}+\xi_i, \mathbf{h}_{\mathscr{N}}\rangle}.$$

Consequently, we observe the following.

Observation 1.1.

$$\textit{If} \qquad (\mathbf{e}_{\mathscr{N}})_i = 1 \qquad \textit{we expect that} \qquad F_{\mathscr{H},\mathbf{y}^{[i]}}(\mathbf{e}_{\mathscr{P}}) > F_{\mathscr{H},\mathbf{y}}(\mathbf{e}_{\mathscr{P}}).$$

The algorithm. In our provable variant, we compute $F_{\mathcal{L}(\mathscr{H},\mathbf{y})}$ butalso compute the flipped scores $F_{\mathcal{L}\left(\mathscr{H},\mathbf{y}^{[i]}\right)}$ for $i \in [\![1, |\mathscr{N}|]\!]$. Computing these additional scores allows us to make easily for each $\mathbf{x} \in \mathbb{F}_2^{|\mathscr{P}|}$ a guess $g(\mathbf{x})$ for the value of $\mathbf{e}_{\mathscr{N}}$. This guess is given as follows:

$$g(\mathbf{x})_i = 1 \quad \text{If} \qquad F_{\mathcal{L}\left(\mathscr{H},\mathbf{y}^{[i]}\right)}(\mathbf{x}) > F_{\mathcal{L}(\mathscr{H},\mathbf{y})}(\mathbf{x}).$$

It is readily seen that when $\mathbf{x} = \mathbf{e}_{\mathscr{P}}$ we expect that the guess is right, namely that $g(\mathbf{x}) = \mathbf{e}_{\mathscr{N}}$. Now, we can test a guess in *polynomial* time by checking that $\mathbf{x}$ concatenated with the associated guess $g(\mathbf{x})$ is a solution to the original decoding problem (by making sure it is of weight t and that, when we remove $\mathbf{y}$, the result is in $\mathcal{C}$). As such, our algorithm simply goes through all $\mathbf{x} \in \mathbb{F}_2^{|\mathscr{P}|}$, makes a guess and checks this guess.

Proving our algorithm. Proving our algorithm only relies on proving that when $\mathbf{x} = \mathbf{e}_{\mathscr{P}}$ then the guess is good with high probability. It is clear here that the second-order concentration bounds given in Proposition 8.1 are sufficient to prove this. Indeed, for each position i we compare only two distributions, the one related to $\mathbf{e}_{\mathscr{N}_i}$ being 0 and the one for $\mathbf{e}_{\mathscr{N}_i}$ being 1. Of course, we must apply these bounds for each $i \in [\![1, n]\!]$, but this only incurs a polynomial loss compared to stronger bounds. We make no claim whatsoever regarding the other cases corresponding to $\mathbf{x} \neq \mathbf{e}_{\mathscr{P}}$ as they will be naturally discarded by our checking step.

Our variant has the same performance as the original algorithm. Last, we argue that in fact our algorithm has the same performance, up to polynomial factors, as the original RLPN algorithm. One could argue that we have somehow lost in performance since the distributions we are trying to distinguish now are closer than the original distributions in double-RLPN. This is essentially not the case. In RLPN the distributions being distinguished had either expected value $\mathbb{E}\left(F_{\mathcal{L}(\mathscr{H},\mathbf{y})}(\mathbf{e}_{\mathscr{P}})\right) = N\delta$ or expected value $\mathbb{E}\left(F_{\mathcal{L}(\mathscr{H},\mathbf{y})}(\mathbf{x})\right) = 0$ when $\mathbf{x} \neq \mathbf{e}_{\mathscr{P}}$. In our provable variant, the distributions that we compare have expected values respectively

$$\mathbb{E}\left(F_{\mathcal{L}(\mathscr{H},\mathbf{y})}(\mathbf{e}_{\mathscr{P}})\right) = N\delta\left(|\mathbf{e}_{\mathscr{N}}|\right)$$
$$\mathbb{E}\left(F_{\mathcal{L}\left(\mathscr{H},\mathbf{y}^{[i]}\right)}(\mathbf{e}_{\mathscr{P}})\right) = N\delta\left(|\mathbf{e}_{\mathscr{N}}| \pm 1\right)$$

and where we recall that δ was defined in Lemma 1.2. The key point is that we can show that in our regime of interest (*i.e.* in the non-oscillatory regime of Krawtchouk polynomials) then $\delta(|\mathbf{e}_{\mathscr{N}}| \pm 1) - \delta(|\mathbf{e}_{\mathscr{N}}|)$ is polynomially relatable to $\delta(|\mathbf{e}_{\mathscr{N}}|)$, namely we can show that

$$\delta(|\mathbf{e}_{\mathscr{N}}| \pm 1) - \delta(|\mathbf{e}_{\mathscr{N}}|) \geqslant \frac{1}{\mathrm{poly}(n)} \delta(|\mathbf{e}_{\mathscr{N}}|).$$

This shows, using Lemma 1.2, that our algorithm solves the decoding problem under a mild polynomial strengthening of the original condition, namely it requires that $N > \mathrm{poly}(n) / \delta(|\mathbf{e}_{\mathscr{N}}|)^2$.

2 Acknowledgements

The author would like to warmly thank Jean-Pierre Tillich for his proofreading and comments on the counterpart of this paper that I wrote as a part of my phd manuscript. The author is funded by pepr PQ-TLS. One part of this work was done in the Capsule team, Irisa, Rennes and the other in the Cosmiq team, Inria Paris.

3 Notation and Preliminaries

3.1 Notation

Set, vector and matrix notation. The set $[a,\, b]$ is closed set of reals between a and b. $[\![a, b]\!]$ indicates the closed integer interval between a and b. For two reals A and B, the set $[A \pm B]$ is $[A - B, A + B]$.

$\mathbb{F}_2$ is the binary field. $|E|$ is the cardinality of a finite set E. Vectors are indicated by lowercase bold letters $\mathbf{x}$ and matrices by uppercase bold letters $\mathbf{A}$. For a vector $\mathbf{x} = (x_i)_{1 \leqslant i \leqslant n}$ and $\mathscr{I} \subset [\![1,\, n]\!]$, $\mathbf{x}_{\mathscr{I}}$ is given by $\mathbf{x}_{\mathscr{I}} = (x_i)_{i \in \mathscr{I}}$ and $\mathbf{x}^{\intercal}$ is the transpose of $\mathbf{x}$ and $|\mathbf{x}|$ stands for the Hamming weight of $\mathbf{x}$. Given two vectors $\mathbf{x} \in \mathbb{F}_2^n$ and $\mathbf{y} \in \mathbb{F}_2^n$, their canonical inner product in $\mathbb{F}_2$ is denoted by $\langle \mathbf{x}, \mathbf{y} \rangle = \sum_{i=1}^{n} x_i y_i$. We denote by $(\mathbf{x} \,||\, \mathbf{y}) \in \mathbb{F}_2^{2n}$ the concatenation of $\mathbf{x}$ and $\mathbf{y}$. For a matrix $\mathbf{A} \in \mathbb{F}_2^{k \times n}$ and $\mathscr{I} \subset [\![1,\, n]\!]$, $\mathbf{A}_{\mathscr{I}}$ is the matrix $\mathbf{A}$ where we kept only the columns whose indices are in $\mathscr{I}$. $\mathrm{rank}(\mathbf{A})$ is the rank of $\mathbf{A}$. $\mathbf{I}_n$ is the identity matrix with n rows and columns. $\mathbf{0}_{k \times n} \in \mathbb{F}_2^{k \times n}$ is the all zero matrix. $\mathbf{0}_n \in \mathbb{F}_2^n$ is the null vector. $\mathcal{S}_w^n \stackrel{\mathrm{def}}{=} \{\mathbf{x} \in \mathbb{F}_2^n \,:\, |\mathbf{x}| = w\}$ is the Hamming sphere of weight w of $\mathbb{F}_2^n$. $\mathbf{1}_A$ is the indicator function of the set A, namely $\mathbf{1}_A(x) = 1$ if $x \in A$ otherwise 0.

Probability. We denote respectively by $\mathbb{E}(X)$ and $\mathbf{Var}(X)$ the expected value and variance of a random variable X. We will use Bienaymé–Chebyshev inequality given as follows.

Proposition 3.1 (Bienaymé–Chebyshev inequality). *For any random variable X and any $\alpha > 0$ we have*

$$\mathbb{P}(|X - \mathbb{E}(X)| > \alpha) \leqslant \frac{\mathbf{Var}(X)}{\alpha^2}.$$

We will also use the union bound.

Proposition 3.2 (Union bound). *Given N events $(E_i)_{i \in [\![1,\, N]\!]}$ we have that $\mathbb{P}\left(\bigcup_{i=1}^{N} E_i\right) \leqslant \sum_{i=1}^{n} \mathbb{P}(E_i)$.*

When $\mathcal{D}$ is a probability distribution we write that $\boldsymbol{X} \sim \mathcal{D}$ to specify that $\boldsymbol{X}$ is distributed according to $\mathcal{D}$. If A is a set, we denote by $\mathcal{U}(A)$ the uniform distribution over A. We denote by $\mathrm{Ber}(p)$ is the Bernoulli distribution of parameter p. By definition if $X \sim \mathrm{Ber}(p)$ then X take value in $\mathbb{F}_2$ and $\mathbb{P}(X=1) = p$. We define the bias of a binary random variable X as

$$\mathrm{bias}(X) \stackrel{\mathrm{def}}{=} \mathbb{P}(X=0) - \mathbb{P}(X=1).$$

In particular if $X \sim \mathrm{Ber}\left(\frac{1-\varepsilon}{2}\right)$ then $\mathrm{bias}(X) = \varepsilon$.

Lemma 3.3 (Pilling up lemma). *If* $X_1 \sim \mathrm{Ber}\left(\frac{1-\varepsilon_1}{2}\right)$ *and* $X_2 \sim \mathrm{Ber}\left(\frac{1-\varepsilon_2}{2}\right)$ *then* $\mathrm{bias}(X_1 + X_2) = \varepsilon_1\varepsilon_2$.

Fourier Transform. Let $f : \mathbb{F}_2^n \to \mathbb{R}$ be a function. We define its Fourier transform $\widehat{f} : \mathbb{F}_2^n \to \mathbb{R}$ as

$$\widehat{f}(\mathbf{x}) = \sum_{\mathbf{a} \in \mathbb{F}_2^n} f(\mathbf{a})(-1)^{\langle \mathbf{x}, \mathbf{a} \rangle} \quad \forall \mathbf{x} \in \mathbb{F}_2^n.$$

We call any algorithm computing this Fourier transform in time $\mathcal{O}(n2^n)$ a Fast Fourier transform.

Landau and asymptotic notation. For real valued functions defined over $\mathbb{R}$ or $\mathbb{N}$ we define $o()$, $\mathcal{O}()$, $\Omega()$, $\Theta()$, in the usual way. We write that $f = \omega(g)$ when f dominates g asymptotically that is when $\lim_{x \to \infty} \frac{|f(x)|}{g(x)} = \infty$. We use the less common notation $\widetilde{\mathcal{O}}()$, where $f = \widetilde{\mathcal{O}}(g)$ means that $f(x) = \mathcal{O}\left(x^k g(x)\right)$ for some constant k.

3.2 Linear codes and decoding problem

Definition 3.4 (Binary linear code). *A binary linear code $\mathcal{C}$ of length n and dimension k is a linear subspace of $\mathbb{F}_2^n$ of dimension k. We say that $\mathcal{C}$ is an $[n,k]$ linear code.*

We call $R = \frac{k}{n}$ the rate of the code. We denote by $\dim(\mathcal{C})$ the dimension of $\mathcal{C}$ as a linear space. We say that $\mathbf{G} \in \mathbb{F}_2^{k \times n}$ is a generator matrix of $\mathcal{C}$ if $\mathcal{C} = \{\mathbf{mG} \ : \ \mathbf{m} \in \mathbb{F}_2^k\}$ and that $\mathbf{H} \in \mathbb{F}_2^{(n-k) \times n}$ is a parity-check matrix of $\mathcal{C}$ if $\mathcal{C} = \{\mathbf{c} \in \mathbb{F}_2^n \ : \ \mathbf{Hc}^\intercal = \mathbf{0}\}$.

Definition 3.5 (Dual code). *For any binary linear code $\mathcal{C}$ of length we denote the dual code by $\mathcal{C}^\perp \stackrel{def}{=} \{\mathbf{h} \in \mathbb{F}_2^n : \langle \mathbf{c}, \mathbf{h} \rangle = 0, \ \forall \mathbf{c} \in \mathcal{C}\}$.*

We have that if $\mathcal{C}$ is an $[n,k]$-linear code then $\mathcal{C}^\perp$ is an $[n, n-k]$-linear code and that if $\mathbf{G}$ is a generator matrix of $\mathcal{C}$ then $\mathbf{G}$ is a parity-check matrix of $\mathcal{C}^\perp$. We will denote by $\mathcal{C}_\mathcal{I}$ the punctured code obtained from $\mathcal{C}$ by keeping only the positions in $\mathcal{I}$, *i.e.* :

$$\mathcal{C}_\mathcal{I} = \{\mathbf{c}_\mathcal{I} : \mathbf{c} \in \mathcal{C}\}.$$

Two of the most standard distributions on linear codes are given as follows.

Definition 3.6 (Distribution on linear codes). *We denote by $\mathcal{U}_\mathbf{G}(n, k)$ the distribution on linear codes obtained by taking a generator matrix of the code uniformly at random in $\mathbb{F}_2^{k \times n}$. We denote by $\mathcal{U}_\mathbf{H}(n, k)$ the distribution on linear codes obtained by taking a parity-check matrix of the code uniformly at random in $\mathbb{F}_2^{(n-k) \times n}$.*

In this work we will focus on designing an algorithm solving the following average binary decoding problem.

Definition 3.7 (Binary Decoding problem $\mathrm{DP}_{\mathbf{G}}(n,k,t)$). *Given $(\mathbf{G},\mathbf{y})$ where $\mathbf{G}$ is a matrix taken uniformly at random in $\mathbb{F}_2^{k\times n}$ and $\mathbf{y}=\mathbf{mG}+\mathbf{e}$ where $\mathbf{m}$ is taken uniformly at random in $\mathbb{F}_2^k$ and $\mathbf{e}$ is taken uniformly at random among vectors of $\mathbb{F}_2^n$ of Hamming weight t, the goal is to output an error vector $\mathbf{e}'$ of Hamming weight t such that $\mathbf{y}$.*

The so called Gilbert-Varshamov distance is an importance quantity which represents essentially the decoding distance where we expect a unique non-planted solution to the decoding problem.

Definition 3.8 (Gilbert-Varshamov distance). *Let $n,k\in\mathbb{N}$ be such that $k\leqslant n$. We define the Gilbert-Varshamov distance $d_{\mathrm{GV}}(n,\ k)$ as the largest integer such that*

$$\sum_{i=0}^{d_{\mathrm{GV}}(n,\ k)}\binom{n}{i}\leqslant 2^{n-k}.$$

The above could be proved using some slight variant of the following standard lemma.

Lemma 3.9 (Probability of belonging to a code). *Let $\mathcal{C}\sim\mathcal{U}_{\mathbf{H}}(n,\ k)$ and let $\mathbf{c}\in\mathbb{F}_2^n\setminus\{\ \mathbf{0}\}$ be a fixed vector. We have that*

$$\mathbb{P}\left(\mathbf{c}\in\mathcal{C}\right)=\frac{1}{2^{n-k}}.$$

If furthermore $\mathbf{d}\in\mathbb{F}_2^n\setminus\{\ \mathbf{0}\}$ is a fixed vector such that $\mathbf{d}\neq\mathbf{c}$ we have that

$$\mathbb{P}\left(\mathbf{c}\in\mathcal{C},\mathbf{d}\in\mathcal{C}\right)\leqslant\left(\frac{1}{2^{n-k}}\right)^2.$$

3.3 Lifting codewords

We recall here the standard fact that computing a vector $\mathbf{h}\in\mathcal{C}^{\perp}$ of small weight w on $\mathscr{N}$ reduces to computing a vector of weight w of $\left(\mathcal{C}^{\perp}\right)_{\mathscr{N}}$ as we can easily lift this vector onto $\mathcal{C}^{\perp}$.

Definition 3.10 (Information set and lifting). *A subset $\mathscr{I}$ of $[\![1,\ n]\!]$ is an information set of a linear code $\mathcal{D}$ if for any $\mathbf{x}\in\mathcal{D}_{\mathscr{I}}$ there exists a unique codeword $\mathbf{d}\in\mathcal{D}$ such that $\mathbf{d}_{\mathscr{I}}=\mathbf{x}$. We define*

$$\mathrm{Lift}\left(\mathcal{D},\ \mathscr{I},\ \mathbf{x}\right)\overset{def}{=}\mathbf{d}\ \textit{where}\ \mathbf{d}\ \textit{is the unique}\ \mathbf{d}\in\mathcal{D}\ \textit{such that}\ \mathbf{d}_{\mathscr{I}}=\mathbf{x}.$$

We call it the lift of $\mathbf{x}$ into $\mathcal{D}$.

This is possible only if $\mathscr{N}$ if an information set of $\mathcal{C}^{\perp}$, which is the case with high probability since the condition that $|\mathscr{N}|\overset{\text{def}}{=}n-s>n-k$ is naturally verified by the double-RLPN parameters. Checking if $\mathscr{N}$ is an information set of $\mathcal{C}^{\perp}$ can be done in polynomial time by checking that $\mathrm{rank}\left(\mathbf{G}_{\mathscr{P}}\right)=|\mathscr{P}|$ where $\mathbf{G}$ is a generator matrix of $\mathcal{C}$. Lifting $\mathbf{h}_{\mathscr{N}}\in\left(\mathcal{C}^{\perp}\right)_{\mathscr{N}}$ can be done in polynomial time by constructing a generator matrix $\mathbf{G}'$ of $\mathcal{C}$ of the form

$$\mathbf{G}'=\begin{bmatrix}\mathbf{I}_s & \mathbf{R}\\ \mathbf{0}_{s\times s} & \mathbf{A}\end{bmatrix}\tag{1}$$

and by defining $\mathbf{h}_{\mathscr{P}}\overset{\text{def}}{=}\mathbf{h}_{\mathscr{N}}\mathbf{R}^{\intercal}$.

3.4 Krawtchouk polynomial

We recall here some properties about Krawtchouk polynomial that will be useful in the article.

Definition 3.11. *(Krawtchouk polynomial) We define the Krawtchouk polynomial $K_w^{(n)}$ of degree w and of order n as $K_w^{(n)}(X) \stackrel{def}{=} \sum_{j=0}^{w} (-1)^j \binom{X}{j}\binom{n-X}{w-j}$.*

The following fact is well known, it gives an alternate expression of the Krawtchouk polynomial (see for instance [vL99, Lemma 5.3.1]) :

Lemma 3.12. *For any $\mathbf{a} \in \mathbb{F}_2^n$,*

$$K_w^{(n)}(|\mathbf{x}|) = \widehat{\mathbf{1}_{\mathcal{S}_t^n}}(\mathbf{x}) = \sum_{\mathbf{y}\in\mathbb{F}_2^n : |\mathbf{y}=w} (-1)^{\langle \mathbf{x},\mathbf{y}\rangle}. \tag{2}$$

Equipped with this lemma it is easy to see that Krawtchouk polynomials are related to the bias of the inner product of two random binary vectors.

Lemma 3.13 (Bias of an inner product). *Let $\mathbf{e} \in \mathcal{S}_t^n$ be a fixed vector and let $\mathbf{h}$ be taken uniformly at random in $\mathcal{S}_w^n$. We have that*

$$\mathrm{bias}(\langle \mathbf{e}, \mathbf{h}\rangle) = \delta_w^{(n)}(t)$$

where

$$\delta_w^{(n)}(t) \stackrel{def}{=} K_w^{(n)}(t) / \binom{n}{w}$$

3.5 Asymptotic expansion

To simplify our proofs we will often use the following standard asymptotic expansion of the binomial coefficient.

Proposition 3.14. *There exists a positive poly-bounded function f such that for any $n, t \in \mathbb{N}$ such that $t \leqslant n$ we have*

$$\frac{1}{f(n)} 2^{h(t/n)n} \leqslant \binom{n}{t} \leqslant f(n) 2^{h(t/n)n}$$

where $h(x) = -x\log_2(x) - (1-x)\log_2(1-x)$ is the binary entropy function. Furthermore, the binary entropy function is differentiable in $]0,\, 1[$.

We will also use this standard result about the asymptotic expansion of Krawtchouk polynomials.

Proposition 3.15 (About the asymptotic expansion of Krawtchouk polynomials). *Let us define $A \stackrel{def}{=} \{\, (\omega,\, \tau) \in [0,\, 1]^2 \,:\, \tau < 1/2 - \sqrt{\omega(1-\omega)}\}$. There exists a positive poly-bounded function f and a bivariate function $\kappa(\omega,\, \tau)$ that is differentiable on A and that is such that for any $w, t, n \in \mathbb{N}$ such that $t < \mathrm{Root}\left(K_w^{(n)}\right)$ we have*

$$\frac{1}{f(n)} 2^{\kappa(w/n,\, t/n)n} < K_w^{(n)}(t) < f(n) 2^{\kappa(w/n,\, t/n)n}.$$

where the delimitation of the root region is defined as

$$\mathrm{Root}\left(K_w^{(n)}\right) \stackrel{def}{=} n/2 - \sqrt{w(n-w)} \tag{3}$$

Proof. This is a direct corollary of [KS21, Section 2.2, point 6.] together with [KS21, Section 2.1.2].

4 Essential on double-RLPN

Our provable variant of double-RLPN reuses a major part of the original double-RLPN algorithm [CDMT24]. We recall that our goal here is to solve the following decoding problem : given an $[n, k]$-linear code $\mathcal{C}$ and a noisy codeword $\mathbf{y} = \mathbf{c} + \mathbf{e}$ where $\mathbf{c} \in \mathcal{C}$ and $\mathbf{e} \in \mathcal{S}_t^n$, the goal is to recover $\mathbf{e}$. The algorithm works as follows. We will detail step 2. and 3. as they will be used in our provable variant.

1. Choose at random $\mathscr{P}$ and $\mathscr{N}$ two complementary subsets of $[\![1,\, n]\!]$
2. Compute a list of LPN samples where the secret is related to $\mathbf{e}_{\mathscr{P}}$. We detail this step in Section 4.1.
3. Compute a score function associated with the LPN samples. It encodes how likely a vector is the secret of the LPN samples. We detail this step in Section 4.2.
4. Somehow recover $\mathbf{e}_{\mathscr{P}}$ using the values of the score function, then recover the rest of the error $\mathbf{e}_{\mathscr{N}}$. This is the step that required the use of the Poisson model to tightly estimate the number of candidates for the secret.

Those steps are iterated as a certain number N_{iter} of times until a bet on the weight of the error on the part $\mathscr{P}$ and $\mathscr{N}$ is verified, say $|\mathbf{e}_{\mathscr{N}}| = u$. The rationale for step 4. is that it recovers $\mathbf{e}$ when the bet is verified, otherwise, it fails. In Section 4.3 we recall the main quantities intervening in the analysis of the algorithm as they will appear in our main theorem that we state in Section 5.

4.1 Computing a list of LPN samples

In this section we suppose that two complementary subsets $\mathscr{P}$ and $\mathscr{N}$ of $[\![1,\, n]\!]$ such that

$$|\mathscr{P}| = s, \qquad \text{and} \quad |\mathscr{N}| = n - s.$$

are given and explain how double-RLPN computes some LPN samples whose secret is related to $\mathbf{e}_{\mathscr{P}}$.

The reduction made in [CDMT24]. The reduction starts from the base remark from [CDMT22] that a dual vectors $\mathbf{h}$ of small weight on $\mathscr{N}$, directly yield an LPN sample

$$\langle \mathbf{y}, \mathbf{h} \rangle = \langle \mathbf{c} + \mathbf{e}, \mathbf{h} \rangle = \langle \mathbf{e}_{\mathscr{P}}, \mathbf{h}_{\mathscr{P}} \rangle + \langle \mathbf{e}_{\mathscr{N}}, \mathbf{h}_{\mathscr{N}} \rangle .$$

Then, [CDMT24] further noticed that the dimension of the LPN samples could be reduced using a technique from [GJL14], at the cost of a mild increase of noise by using the sparsity of the secret $\mathbf{e}_{\mathscr{P}}$. The reduction works by considering an $[s, k_{\text{aux}}]$-linear auxiliary code $\mathcal{C}_{\text{aux}}$ and decoding $\mathbf{h}_{\mathscr{P}}$ onto $\mathcal{C}_{\text{aux}}$, *i.e.* , by finding $\mathbf{c}_{\text{aux}} \in \mathcal{C}_{\text{aux}}$ and an error $\mathbf{e}_{\text{aux}}$ of low weight t_{aux} such that

$$\mathbf{h}_{\mathscr{P}} = \mathbf{c}_{\text{aux}} + \mathbf{e}_{\text{aux}} \qquad\qquad |\mathbf{e}_{\text{aux}}| = t_{\text{aux}}.$$

Now considering $\mathbf{G}_{\text{aux}} \in \mathbb{F}_2^{k_{\text{aux}} \times s}$ a generator matrix of $\mathcal{C}_{\text{aux}}$, there exists a unique $\mathbf{m}_{\text{aux}} \in \mathbb{F}_2^{k_{\text{aux}}}$ such that $\mathbf{c}_{\text{aux}} = \mathbf{m}_{\text{aux}} \mathbf{G}_{\text{aux}}$. This allows to get the following LPN sample $(\mathbf{m}_{\text{aux}}, \langle \mathbf{y}, \mathbf{h} \rangle)$ where

$$\langle \mathbf{y}, \mathbf{h} \rangle = \langle \mathbf{a}, \mathbf{s} \rangle + e \quad \text{where} \quad \begin{cases} \mathbf{s} & = \mathbf{e}_{\mathscr{P}} \mathbf{G}_{\text{aux}}^{\intercal} \in \mathbb{F}_2^{k_{\text{aux}}} \\ \mathbf{a} & = \mathbf{m}_{\text{aux}} \\ e & = \langle \mathbf{e}_{\text{aux}}, \mathbf{e}_{\mathscr{P}} \rangle + \langle \mathbf{e}_{\mathscr{N}}, \mathbf{h}_{\mathscr{N}} \rangle . \end{cases} \tag{4}$$

The algorithm. The procedure computing the LPN samples, which we call LPN-Samples and describe in Algorithm 2, works as follows.

– Calling a procedureDecoded-Dual-Vectors$(\mathcal{C}, \mathscr{P}, \mathscr{N}, \mathbf{G}_{\text{aux}})$ that outputs a set $\mathscr{H}$ of decoded dual vectors

$$\mathscr{H} \subset \{\ (\mathbf{h}, \mathbf{e}_{\text{aux}}) \in \mathcal{C}^{\perp} \times \mathcal{S}^{s}_{t_{\text{aux}}}\ :\ \mathbf{h}_{\mathscr{P}} + \mathbf{e}_{\text{aux}} \in \mathcal{C}_{\text{aux}}\}.$$

We recall it in Algorithm 2.

– Create and output a list $\mathcal{L}(\mathbf{y}, \mathscr{H})$ of LPN samples as described in Notation 4.1. This can be done in polynomial time with basic linear algebra.

From now on we use the following notation.

Notation 4.1 *Given a vector $\mathbf{g} \in \mathbb{F}_2^n$ and when the set of positions $\mathscr{P}$ and $\mathscr{N}$ and the generator matrix $\mathbf{G}_{\text{aux}}$ are clear from context we denote by $\mathcal{L}(\mathbf{g}, \mathscr{H})$ the multiset containing for each $(\mathbf{h}, \mathbf{e}_{\text{aux}}) \in \mathscr{H}$ the corresponding sample $(\mathbf{m}_{\text{aux}}, \langle \mathbf{g}, \mathbf{h} \rangle)$ where $\mathbf{m}_{\text{aux}} \in \mathbb{F}_2^{k_{\text{aux}}}$ is the unique vector such that $\mathbf{m}_{\text{aux}}\mathbf{G}_{\text{aux}} + \mathbf{e}_{\text{aux}} = \mathbf{h}_{\mathscr{P}}$.*

Algorithm 1

Name: LPN-Samples$(\mathbf{y}, \mathcal{C}, \mathscr{P}, \mathscr{N}, \mathbf{G}_{\text{aux}})$
1: $\mathscr{H} \leftarrow$ Decoded-Dual-Vectors$(\mathcal{C}, \mathscr{P}, \mathscr{N}, \mathbf{G}_{\text{aux}})$
2: **return** $\mathcal{L}(\mathbf{y}, \mathscr{H})$

The procedure computing the decoded dual vectors. We describe here the procedure Decoded-dual-vectors. First, note that by enforcing that $\mathscr{N}$ is an information set of $\mathcal{C}^{\perp}$ (which is essentially always the case for reasonable parameters verifying $|\mathscr{P}| \leqslant k$) we ensure that computing the low weight dual vectors reduces to computing low-weight $\mathbf{h}_{\mathscr{N}} \in \left(\mathcal{C}^{\perp}\right)_{\mathscr{N}}$. Indeed, as detailed in Section 3.3, one can uniquely lift $\mathbf{h}_{\mathscr{N}}$ onto a vector $\mathbf{h}$ of $\mathcal{C}^{\perp}$.

To work, the procedure Decoded-dual-vectors uses i) a procedure Compute-Small-Codeword$(\mathcal{D},\ w)$ that returns a subset of $\{\ \mathbf{d} \in \mathcal{D}\ :\ |\mathbf{d}| = w\}$ and ii) a procedure Decode-Auxilary$(\mathbf{G}_{\text{aux}}, \mathbf{a}, t_{\text{aux}})$ that returns a subset of $\{\ \mathbf{e}_{\text{aux}} \in \mathcal{S}^{s}_{t_{\text{aux}}}\ :\ \mathbf{a} + \mathbf{e}_{\text{aux}} \in \mathcal{C}_{\text{aux}}\}$ and which will be called with $\mathbf{a} = \mathbf{h}_{\mathscr{P}}$. The algorithm is described in Algorithm 2.

Algorithm 2

Name: Decoded-Dual-Vectors$(\mathcal{C}, \mathscr{P}, \mathscr{N}, \mathbf{G}_{\text{aux}})$
1: Continue if $\mathscr{N}$ is an information set of $\mathcal{C}^{\perp}$
2: $\mathscr{W}_{\mathscr{N}} \leftarrow$ Compute-Small-Codeword$(\left(\mathcal{C}^{\perp}\right)_{\mathscr{N}},\ w)$ ▷ *Returns a subset of* $\{\ \mathbf{h} \in \mathcal{C}^{\perp}\ :\ |\mathbf{h}_{\mathscr{N}}| = w\}$
3: $\mathscr{W} \leftarrow \{\ \mathbf{h} \in \mathcal{C}^{\perp}\ :\ \mathbf{h} = \text{Lift}\left(\mathcal{C}^{\perp},\ \mathscr{N},\ \mathbf{h}_{\mathscr{N}}\right)\}$ ▷ *The lifting operation is defined in Section 3.3*
4: $\mathscr{H} \leftarrow \emptyset$
5: **for** $\mathbf{h} \in \mathscr{W}$ **do**
6: $\quad \mathcal{E} \leftarrow$ Decode-Auxilary$(\mathcal{C}_{\text{aux}}, \mathbf{h}_{\mathscr{P}})$ ▷ *Returns a set of error of small weight* $\mathbf{e}_{\text{aux}} \in \mathcal{S}^{s}_{t_{\text{aux}}}$ *s.t* $\mathbf{h}_{\mathscr{P}} - \mathbf{e}_{\text{aux}} \in \mathcal{C}_{\text{aux}}$
7: $\quad$ **for** $\mathbf{e}_{\text{aux}} \in \mathcal{E}$ **do**
8: $\quad\quad \mathscr{H}$.append$((\mathbf{h},\ \mathbf{e}_{\text{aux}}))$ ▷ *The set of decoded dual vectors*

Proposition 4.1 (Complexity for producing the LPN samples). *The complexity of a call to* LPN-Samples$(\mathbf{y}, \mathcal{C}, \mathscr{P}, \mathscr{P}, \mathbf{G}_{\text{aux}})$ *is given by*

$$\text{poly}(n)\,(T_{\text{eq}} + N\ T_{\text{dec}})$$

where $N \stackrel{def}{=} |\mathscr{W}|$ *is the number of vectors returns by* Compute-Small-Codeword *and* T_{eq} *and* T_{dec} *are the time complexity of the* Compute-Small-Codeword *and* Decode-Auxilary *procedure respectively.*

4.2 Computing the score function

In double-RLPN solving the LPN problem, *i.e.* recovering the secret $\mathbf{s} \stackrel{\text{def}}{=} \mathbf{e}_{\mathscr{P}}\mathbf{G}_{\text{aux}}^{\intercal}$, is done by computing exhaustively a score function $F_{\mathcal{L}}$ that encodes how likely a vector $\mathbf{z}$ is the secret of the LPN problem.

Definition 4.2 (Score function). *Let* $\mathcal{L}$ *be a list of LPN samples of the form* $(\mathbf{a}, b) \in \mathbb{F}_2^{k_{\text{aux}}} \times \mathbb{F}_2$*. For all* $\mathbf{z} \in \mathbb{F}_2^{k_{\text{aux}}}$ *we define the score function as*

$$F_{\mathcal{L}}(\mathbf{z}) \stackrel{def}{=} \sum_{(\mathbf{a},b)\in\mathcal{L}} (-1)^{b-\langle \mathbf{a},\mathbf{z}\rangle}.$$

double-RLPN Computes this score function using a Fast Fourier Transform-based approach, which was already used in RLPN and more generally, which was introduced by [LF06] to seed-up LPN solvers. We refer to the previous article for more details. The procedure is described in Score-function$(\mathcal{L})$ and is detailed in Algorithm 3.

Algorithm 3

Name: Score-function$(\mathcal{L})$

1: **for** $\mathbf{a} \in \mathbb{F}_2^{k_{\text{aux}}}$ **do** ▷ *Compute LPN score function with FFT*
2: $\quad f_{\mathcal{L}}(\mathbf{r}) \leftarrow \sum_{(\mathbf{a},)\in\mathcal{L}\,:\,\mathbf{a}=\mathbf{r}} (-1)^b$
3: $F_{\mathcal{L}} \leftarrow \text{FFT}(f_{\mathcal{L}})$ ▷ FFT(f) *is a Fast Fourier transform algorithm that outputs the Fourier Transform* $\widehat{f}$ *of* f*.*
4: **return** $F_{\mathcal{L}}$

Proposition 4.3. *The procedure* Score-function$(\mathcal{L})$ *takes as input a list* $\mathcal{L}$ *of LPN samples* $(\mathbf{a}, b) \in \mathbb{F}_2^{k_{\text{aux}}} \times \mathbb{F}_2$ *and returns* $F_{\mathcal{L}}$ *in time and memory respectively*

$$\textbf{\textit{Time}} = \text{poly}(n)\left(|\mathcal{L}| + 2^{k_{\text{aux}}}\right), \qquad \textbf{\textit{Memory}} = \text{poly}(n)\left(|\mathcal{L}| + 2^{k_{\text{aux}}}\right)$$

4.3 Key quantity of the analysis of double-RLPN

We recall here the basic quantities intervening in the analysis of double-RLPN and that of our provable variant. This section allows the reader to interpret the quantities that will appear in the main theorem of this paper and that we state in the next section.

The analysis of double-RLPN was done when $\mathcal{C}$ is chosen by taking its generator matrix uniformly at random in $\mathbb{F}_2^{k\times n}$ and by supposing that the set $\mathscr{W}$ of computed dual vectors is

the whole set $\{ \mathbf{h} \in \mathcal{C}^\perp : |\mathbf{h}_{\mathscr{N}}| = w\}$. So that the quantities are well-defined, we condition all the probabilities by the event that $\mathscr{N}$ is an information set of $\mathcal{C}^\perp$. To simplify the analysis, the code $\mathcal{C}_{\text{aux}}$ was taken as a uniformly random $[s, k_{\text{aux}}]$-linear codes equipped with a genie-aided list decoder, namely by supposing that the $\text{DECODE}(\mathcal{C}_{\text{aux}}, \mathbf{h}_{\mathscr{P}}, t_{\text{aux}})$ returns the set $\{ \mathbf{e}_{\text{aux}} \in \mathcal{S}^s_{t_{\text{aux}}} : \mathbf{e}_{\text{aux}} - \mathbf{h}_{\mathscr{P}} \in \mathcal{C}_{\text{aux}}\}$. All in all this means that the set of decoded dual vectors $\mathscr{H}$ is comprised of all possible couples, namely

$$\mathscr{H} = \{ (\mathbf{h}, \mathbf{e}_{\text{aux}}) \in \mathcal{C}^\perp \times \mathcal{S}^s_{t_{\text{aux}}} : |\mathbf{h}_{\mathscr{N}}| = w \text{ and } \mathbf{h}_{\mathscr{P}} + \mathbf{e}_{\text{aux}} \in \mathcal{C}_{\text{aux}}\}.$$

There are two relevant quantities to the analysis : the average number of LPN samples and the bias of the noise of the samples.

Number of available LPN samples. First, one could show that the expected number of samples is

$$\begin{aligned}\mathbb{E}(|\mathcal{L}|) &= \mathbb{E}(|\mathscr{H}|) \\ &= \frac{\binom{n-s}{w}}{2^{k-s}} \frac{\binom{s}{t_{\text{aux}}}}{2^{s-k_{\text{aux}}}} \\ &= \frac{\binom{n-s}{w}\binom{s}{t_{\text{aux}}}}{2^{k-k_{\text{aux}}}}.\end{aligned}$$

The reader interested in a rigorous proof can see this statement as a direct corollary of the later stated Proposition 8.1. More roughly, this equality comes from the fact that i) by construction, the expected number of dual vectors of $\mathcal{C}$ weight w on $\mathscr{N}$ is given by $\mathbb{E}\left(\left(\mathcal{C}^\perp\right)_{\mathscr{N}} \bigcap \mathcal{S}^{n-s}_w\right) = \binom{n-s}{w}/2^{k-s}$ and ii) the expected number of error returned by the auxiliary list decoder for each $\mathbf{h}_{\mathscr{P}}$ is $\mathbb{E}\left((\mathcal{C}_{\text{aux}} + \mathbf{h}_{\mathscr{P}}) \bigcap \mathcal{S}^s_{t_{\text{aux}}}\right) = \binom{s}{t_{\text{aux}}}/2^{s-k_{\text{aux}}}$.

Bias of the LPN samples. The second key quantity is the bias of the noise $\langle \mathbf{e}_{\mathscr{N}}, \mathbf{h}_{\mathscr{N}}\rangle + \langle \mathbf{e}_{\mathscr{P}}, \mathbf{e}_{\text{aux}}\rangle$ of the LPN samples. This quantity obviously depends on the weight of the vectors. We recall that by construction

$$|\mathbf{h}_{\mathscr{N}}| = w, \quad |\mathbf{e}_{\text{aux}}| = t_{\text{aux}}$$

and that a bet is made on the weight of the error on each subpart, namely that

$$|\mathbf{e}_{\mathscr{N}}| = u, \quad |\mathbf{e}_{\mathscr{P}}| = t - u.$$

Forgetting about the fact that all these vectors come from a decoding problem, we can easily estimate the bias of this noise.

Lemma 4.4. *Bias of the noise by forgetting about the code structure. Let $\mathscr{P}$ and $\mathscr{N}$ two complementary subset of $[\![1, n]\!]$ of size s and $n-s$ respectively. Let $\mathbf{e} \in \mathcal{S}^n_t$ such that $|\mathbf{e}_{\mathscr{P}}| = u$ and $|\mathbf{e}_{\mathscr{N}}| = t - u$. Then*

$$\text{bias}\left(\langle \mathbf{e}_{\mathscr{N}}, \mathbf{h}_{\mathscr{N}}\rangle + \langle \mathbf{e}_{\mathscr{P}}, \mathbf{e}_{\text{aux}}\rangle\right) = \delta^{(n-s)}_w(u)\, \delta^{(s)}_{t_{\text{aux}}}(t-u)$$

where $\mathbf{h}_{\mathscr{N}}$ is taken uniformly at random in $\mathcal{S}^{n-s}_w$ and $\mathbf{e}_{\text{aux}}$ is taken uniformly at random in $\mathcal{S}^s_{t_{\text{aux}}}$.

Proof. This is a direct consequence of the Pilling-Up lemma and Lemma 3.13.

Of course, one cannot completely forget this code structure but ultimately [CDMT24, Proposition 2] showed with a second-order technique that under some conditions, the quantity $\delta_w^{(n-s)}(u)\,\delta_{t_{\text{aux}}}^{(s)}(t-u)$ was with good probability a good approximation of the bias of the noise of the LPN samples.

Rationale behind the attack. Under the flawed [MT23, CDMT24] LPN modelling, one would expect to require that the number N of LPN samples is superior to n/ε^2 in order to be able to recover the secret of an LPN problem with a bias of noise ε. In our case this would roughly be that

$$\frac{\binom{n-s}{w}\binom{s}{t_{\text{aux}}}}{2^{k-k_{\text{aux}}}} > \frac{n}{\left(\delta_w^{(n-s)}(u)\,\delta_{t_{\text{aux}}}^{(s)}(t-u)\right)^2}$$

to recover the secret $\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\text{aux}}^{\intercal}$.

5 Results and main theorem

In this section, we state the main result of this paper. We state a theorem giving the performance of our provable variant of double-RLPN. We will describe in detail this provable variant in the next Section 6. For simplicity, we state, as was the case for double-RLPN, our theorem in the case where we suppose that we have access to a procedure that computes all the vectors of $\left(\mathcal{C}^{\perp}\right)_{\mathscr{N}}$ of weight w. To make the quantities appearing in our theorem more intelligible, the reader can refer to the previous Section 4.3. We mark with a "*" the new constraints that did not appear in the original paper [CDMT24, Proposition 9].

Theorem 5.1. *There exists a positive poly-bounded function f such that for any $k, t, s, k_{\text{aux}}, t_{\text{aux}}, w, u \in \mathbb{N}$ implicit functions of a parameter $n \in \mathbb{N}$ and any procedure* Compute-small-codewords *that are such that*

1. *(Computing all the whole set of dual vectors $\{\mathbf{h} \in \mathcal{C}^{\perp} \,:\, |\mathbf{h}_{\mathscr{N}}| = w\}$)*

$$\mathbb{P}\left(\textsc{Compute-small-codewords}(\mathcal{D}) \neq \mathcal{D} \bigcap \mathcal{S}_w^{n-s}\right) \in o(1)\,,$$

   where $\mathcal{D}$ is taken by choosing its parity-check matrix uniformly at random in $\mathbb{F}_2^{(k-s)\times(n-s)}$.
2. *(Main constraint that we have enough dual vectors)*

$$\frac{\binom{n-s}{w}\binom{s}{t_{\text{aux}}}}{2^{k-k_{\text{aux}}}} \in \Omega\left(\frac{f(n^{s/k_{\text{aux}}})}{\left(\delta_w^{(n-s)}(u)\,\delta_{t_{\text{aux}}}^{(s)}(t-u)\right)^2}\right),$$

3. *(Decoding the auxiliary code below Gilbert-Varshamov) $\binom{s}{t_{\text{aux}}}/2^{s-k_{\text{aux}}} \in \mathcal{O}(1)$,*
4. **(Linear scaling of the parameters) $s/k_{\text{aux}} \in \mathcal{O}(1)$,*
5. **($\mathscr{N}$ is an information set of the code $\mathcal{C}$) $n-s-k \in \omega(1)$,*
6. **(Small technical constraints) $\text{Root}\left(K_w^{(n-s)}\right) - u \in \Omega(n-s)$ and $\text{Root}\left(K_{t_{\text{aux}}}^{(s)}\right) - (t-u) \in \Omega(s)$ and $k-s \in \omega(1)$ and $n-k \in \omega(1)$*

then there exists an algorithm solving $\mathrm{DP}_{\mathbf{G}}(n,k,t)$ *with probability* $1-o(1)$ *in time and memory*

$$\boldsymbol{Time} = \widetilde{\mathcal{O}}\left(\frac{\binom{n}{t}}{\binom{s}{t-u}\binom{n-s}{u}}\left(T_{\mathrm{eq}}+2^{k_{\mathrm{aux}}}\right)\right), \qquad \boldsymbol{Memory} = \widetilde{\mathcal{O}}\left(\left(M_{\mathrm{eq}}+2^{k_{\mathrm{aux}}}\right)\right)$$

where T_{eq} *and* M_{eq} *are respectively time and memory complexity of one call to the procedure* Compute-small-codewords$(\mathcal{D})$ *and where we recall that* $\delta_w^{(n)}(t)$ *is defined in Lemma 3.13 and* $\mathrm{Root}\left(K_w^{(n)}\right)$ *is defined in Eq.* (3)*. In particular, the said algorithm is the one defined later in Definition 7.4 by taking* $b \stackrel{def}{=} \lceil s/k_{\mathrm{aux}} \rceil$ *along with a good choice for* N_{iter}*.*

Next, we make a comparison of [CDMT24, Proposition 9] giving the performance of double-RLPN with the theorem that we have just stated.

- The proposition giving the performance of the algorithm in [CDMT24, Proposition 9] requires the use of the Poisson model [CDMT24, Model 1] whereas our theorem do not require any model.
- [CDMT24, Proposition 9] the complexity is in $2^{k_{\mathrm{aux}}} + T_{\mathrm{eq}} 2^{o(n)} + T_{\mathrm{check}}$ where T_{check} is an additional term for checking false candidates and the $2^{o(n)}$ is the cost of one call to the auxiliary decoder, we do not have this last term because of the auxiliary code that we use.
- Our provable variant will require that $\mathscr{N}$ is an information set of $\mathcal{C}$. So that this happens with good probability we had to add the constraint on the parameters that $n-s-k \in \omega(1)$ where we recall that $n-s = |\mathscr{N}|$. However, because $s<k$ we have that this is unconditionally true for codes of rate $R \stackrel{\mathrm{def}}{=} k/n$ smaller than 0.5. We believe we could remove this constraint but at the cost of a slightly more complex algorithm.
- The constraint that $s/k_{\mathrm{aux}} \in \mathcal{O}(1)$ comes from the choice we made for the auxiliary code $\mathcal{C}_{\mathrm{aux}}$ in Section 7 : it is the condition required for the code and its decoder to be optimal up to polynomial factors. In practice for the parameter regime we are interested in this article, namely $k=\Theta(n)$ and $t=\Theta(n)$, the optimal values of the parameters (*i.e.* matching the main constraint and minimizing the complexity) always verify this condition. In particular, this is the case for all the given parameters of [CDMT24]. We believe that this condition could be removed by using Polar codes as suggested in [Car20, CDMT24], but then the proof would be much more difficult.
- In [CDMT24, Proposition 9], the main constraint needed for the proof under the Poisson model was that

$$\frac{\binom{n-s}{w}\binom{s}{t_{\mathrm{aux}}}}{2^{k-k_{\mathrm{aux}}}} \in \Omega\left(\frac{n^8}{\left(\delta_w^{(n-s)}(u)\,\delta_{t_{\mathrm{aux}}}^{(s)}(t-u)\right)^2}\right). \tag{5}$$

  Here we have some other poly-bounded function $f(n^b)$ instead of n^8. The point is that, regardless of the polynomial, it is easy to create from a set of parameters verifying the constraint given by Eq. (5) a new set of parameters verifying this new constraint by increasing w only slightly by a $\mathcal{O}(\log_2(n))$ term. This could rigorously be shown by computing the derivative in w of the asymptotic expansion of $\frac{\binom{n-s}{w}\binom{s}{t_{\mathrm{aux}}}}{2^{k-k_{\mathrm{aux}}}}\left(\delta_w^{(n-s)}(u)\,\delta_{t_{\mathrm{aux}}}^{(s)}(t-u)\right)^2$.

– We added some small additional constraints (Point 6. in the theorem) to make our statement rigorous. In practice, those are always verified for non-degenerate parameters. And, in particular, they are verified for all the asymptotic parameter datasets given in [CDMT24].

In particular, any complexity claims about double-RLPN made using [CDMT24, Proposition 9] under the Poisson model are provably achieved (by the algorithm that we describe next), up to polynomial factors, when the rate of the code $R \stackrel{\text{def}}{=} k/n$ is smaller than 0.5.

6 Fully provable double-RLPN

Let us now describe our provable variant of double-RLPN. Here we are given a code $\mathcal{C}$ and some noisy codeword $\mathbf{y} = \mathbf{c} + \mathbf{e}$ where $\mathbf{c} \in \mathcal{C}$ and $\mathbf{e} \in \mathcal{S}_n^t$ and our goal is to recover $\mathbf{e}$.

Our algorithm reuses the main loop of double-RLPN by choosing at random two complementary subsets $\mathscr{P}$ and $\mathscr{N}$. Then we will call multiple times the main procedure (see Section 4) of double-RLPN to build a GUESSING-$\mathbf{e}_{\mathscr{N}}(\mathcal{C}, \mathbf{y}, \mathscr{P}, \mathscr{N})$ procedure in Section 6.1 that returns a set $\mathcal{G}$ containing many vectors, hopefully one of which is equal to $\mathbf{e}_{\mathscr{N}}$. We then build a checking procedure CHECKING-$\mathbf{e}_{\mathscr{N}}(\mathcal{G},\ \mathcal{C}, \mathbf{y}, \mathscr{P}, \mathscr{N}, t)$ in Section 6.2 which will check each vector for $\mathbf{e}_{\mathscr{N}}$ and return the whole error if that is the case.

Algorithm 4 Provable-double-RLPN algorithm

Name: PROVABLE-DOUBLE-RLPN$(\mathcal{C}, \mathbf{y}, t)$
Input: $\mathcal{C}$ a linear code of length n, $\mathbf{y} = \mathbf{c} + \mathbf{e} \in \mathbb{F}_2^n$ with $\mathbf{c} \in \mathcal{C}$ and $|\mathbf{e}| = t$
Parameter: s, w, u and N_{iter}

while $i = 1 \dots N_{\text{iter}}$ **do**
  $\mathscr{P} \xleftarrow{\$} \{\mathscr{P} \subset [\![1,\ n]\!] \ :\ |\mathscr{P}| = s\}$ ▷ *Hope that* $|\mathbf{e}_{\mathscr{N}}| = u$
  $\mathscr{N} \leftarrow [\![1,\ n]\!] \setminus \mathscr{P}$
  $\mathcal{G} \leftarrow$GUESSING-$\mathbf{e}_{\mathscr{N}}(\mathcal{C}, \mathbf{y}, \mathscr{P}, \mathscr{N})$
  $\mathbf{e} \leftarrow$ CHECKING-$\mathbf{e}_{\mathscr{N}}(\mathcal{G},\ \mathcal{C}, \mathbf{y}, \mathscr{P}, \mathscr{N}, t)$
  if $\mathbf{e} \neq \perp$ **then**
    return $\mathbf{e}$

6.1 Guessing step

Here we will start with $\mathcal{F}$ a family of $[s, k_{\text{aux}}]$-linear code and we will By reusing the reduction used in double-RLPN we notice that given an auxiliary code $\mathcal{C}_{\text{aux}}$ with generator matrix $\mathbf{G}_{\text{aux}}$ and $\mathscr{H}$ a set of decoded dual vectors

$$\mathscr{H} \subset \{\ (\mathbf{h}, \mathbf{e}_{\text{aux}}) \in \mathcal{C}^{\perp} \times \mathcal{S}_{t_{\text{aux}}}^{s}\ :\ \mathbf{h}_{\mathscr{P}} + \mathbf{e}_{\text{aux}} \in \mathcal{C}_{\text{aux}}\}$$

then we recall that the set of considered LPN sample $\mathcal{L}(\mathbf{y}, \mathscr{H})$ contains for each $(\mathbf{h}, \mathbf{e}_{\text{aux}}) \in \mathcal{C}_{\text{aux}}$ the sample

$$(\mathbf{m}_{\text{aux}}, \langle \mathbf{y}, \mathbf{h} \rangle) = \left(\mathbf{m}_{\text{aux}}, \left\langle \underbrace{\mathbf{e}_{\text{aux}} \mathbf{G}_{\text{aux}}^{\intercal}}_{\text{secret}}, \mathbf{m}_{\text{aux}} \right\rangle + \underbrace{\langle \mathbf{e}_{\mathscr{P}}, \mathbf{e}_{\text{aux}} \rangle + \langle \mathbf{e}_{\mathscr{N}}, \mathbf{h}_{\mathscr{N}} \rangle}_{\text{noise}} \right)$$

where $\mathbf{m}_{\text{aux}} \in \mathbb{F}_2^{k_{\text{aux}}}$ is the unique vector such that $\mathbf{m}_{\text{aux}} \mathbf{G}_{\text{aux}} + \mathbf{e}_{\text{aux}} = \mathbf{h}_{\mathscr{P}}$.

Here, our strategy is, for each $\mathbf{z} \in \mathbb{F}_2^{k_{\text{aux}}}$, to make a guess on the value of $\mathbf{e}_{\mathscr{N}}$. We make this guess bit by bit. We will exploit the fact that flipping the positions of the noisy codeword $\mathbf{y} = \mathbf{c} + \mathbf{e}$ is exactly flipping the positions of the error vector $\mathbf{e}$. As such, we expect that flipping the i'th bit of $\mathbf{y}$ in $\mathscr{N}$ as described in the following definition changes the noise of the LPN sample (and hence the score function).

Definition 6.1. *For any $i \in [\![1,\, n]\!]$ we denote by $\mathbf{y}^{[i]} \in \mathbb{F}_2^n$ the vector defined as $\left(\mathbf{y}^{[i]}\right)_{\mathscr{P}} \stackrel{def}{=} \mathbf{y}_{\mathscr{P}}$ and $\left(\mathbf{y}^{[i]}\right)_{\mathscr{N}} \stackrel{def}{=} \mathbf{y}_{\mathscr{N}} + \xi_i$ where $\xi_i \in \mathbb{F}_2^{n-s}$ is the vector which is zero everywhere except on its i'th coordinate.*

We make use of the following rewriting of the score function.

Lemma 6.2 (Score function evaluated on the secret). *When $\mathbf{y} = \mathbf{c} + \mathbf{e}$ with $\mathbf{c} \in \mathcal{C}$ we have that*

$$F_{\mathcal{L}(\mathbf{y},\mathscr{H})}\left(\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\text{aux}}^{\intercal}\right) = \sum_{(\mathbf{h},\, \mathbf{e}_{\text{aux}}) \in \mathscr{H}} (-1)^{\langle \mathbf{e}_{\mathscr{N}}, \mathbf{h}_{\mathscr{N}} \rangle + \langle \mathbf{e}_{\mathscr{P}}, \mathbf{e}_{\text{aux}} \rangle}.$$

Proof. This follows from a direct application of the definition of the score function F in Definition 4.2 and the list of LPN samples $\mathcal{L}(\mathbf{y}, \mathscr{H})$ in Notation 4.1.

Clearly, we have that,

$$F_{\mathcal{L}\left(\mathscr{H},\mathbf{y}^{[i]}\right)}\left(\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\text{aux}}^{\intercal}\right) = \sum_{(\mathbf{h},\, \mathbf{e}_{\text{aux}}) \in \mathscr{H}} (-1)^{\langle \mathbf{e}_{\mathscr{N}} + \xi_i, \mathbf{h}_{\mathscr{N}} \rangle + \langle \mathbf{e}_{\mathscr{P}}, \mathbf{e}_{\text{aux}} \rangle}.$$

And thus we expected that $F_{\mathscr{H},\mathbf{y}^{[i]}}\left(\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\text{aux}}^{\intercal}\right) > F_{\mathscr{H},\mathbf{y}}\left(\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\text{aux}}^{\intercal}\right)$ if we flipped an erroneous position, namely if $\left(\mathbf{e}_{\mathscr{N}}\right)_i = 1$. This is our decision rationale for our guess on $\mathbf{e}_{\mathscr{N}}$.

Algorithm We compute for each $\mathbf{z} \in \mathbb{F}_2^{k_{\text{aux}}}$ and associated guess for $\mathbf{e}_{\mathscr{N}}$, call it $G(\mathbf{z}) \in \mathbb{F}_2^{n-s}$ as described in the following definition.

Definition 6.3. *For each $\mathbf{x} \in \mathbb{F}_2^{k_{\text{aux}}}$ we denote by $G(\mathbf{z})$ the vector of $\mathbb{F}_2^{n-s}$ whose i'th coordinate is equal to*

$$G(\mathbf{z})_i \stackrel{def}{=} \begin{cases} 1 & \text{if} \quad F_{\mathscr{H},\mathbf{y}^{[i]}}(\mathbf{z}) > F_{\mathscr{H},\mathbf{y}}(\mathbf{z}) \\ 0 & \text{else.} \end{cases}.$$

We call $G(\mathbf{z})$ the guess for $\mathbf{e}_{\mathscr{N}}$ related to $\mathbf{z}$.

We then store these guesses in a set $\mathcal{G} = \{(\mathbf{z},\, G(\mathbf{z})) \,:\, \mathbf{z} \in \mathbb{F}_2^{k_{\text{aux}}}\}$ and outputs it.

Algorithm 5

Name: GUESSING-$\mathbf{e}_{\mathscr{N}}(\mathcal{C}, \mathbf{y}, \mathscr{P}, \mathscr{N})$
Input: $\mathcal{C}, \mathbf{y}, \mathscr{P}, \mathscr{N}, \mathcal{F}$

$\mathcal{C}_{\text{aux}} \xleftarrow{\$} \mathcal{F}$ ▷ *A code $\mathcal{C}_{\text{aux}}$ is drawn uniformly at random in $\mathcal{F}$*
$\mathbf{G}_{\text{aux}} \leftarrow$ A generator matrix of $\mathcal{C}_{\text{aux}}$
$\mathscr{H} \leftarrow$ DECODED-DUAL-VECTOR$(\mathcal{C},\ \mathscr{P},\ \mathscr{N},\ \mathcal{F})$
$\mathcal{L} \leftarrow \mathcal{L}(\mathbf{y}, \mathscr{H})$ ▷ *The list of LPN samples as defined in Notation 4.1*
$F_{\mathcal{L}(\mathbf{y},\mathscr{H})} \leftarrow$ SCORE-FUNCTION$(\mathcal{L})$ ▷ *Reference value of the score function*
while $i = 1 \dots n - s$ **do**
    $\mathbf{y}^{[i]} \leftarrow \mathbf{y} + \xi_{\mathscr{N}_i}$ ▷ *This flips the i'th bit of $\mathbf{e}_{\mathscr{N}}$*
    $\mathcal{L}^{(i)} \leftarrow \mathcal{L}\left(\mathbf{y}^{[i]}, \mathscr{H}\right)$ ▷ *The list of flipped LPN samples as defined in Notation 4.1*
    $F_{\mathcal{L}\left(\mathbf{y}^{[i]},\mathscr{H}\right)} \leftarrow$ SCORE-FUNCTION$(\mathcal{L}^{(i)})$ ▷ *Comparative score function*
Compute $G(\mathbf{z})$ (Definition 6.3) for all $\mathbf{z} \in \mathbb{F}_2^{k_{\text{aux}}}$
return $\{(\mathbf{z},\ G(\mathbf{z}))\ :\ \mathbf{z} \in \mathbb{F}_2^{k_{\text{aux}}}\}$

6.2 Checking step

Here we want to check each guess for $\mathbf{e}_{\mathscr{N}}$. It is important that each guess is checked in polynomial time as we range over the whole secret space $\mathbb{F}_2^{k_{\text{aux}}}$.

The procedure. Note that if $\mathscr{N}$ is an information set of the code $\mathcal{C}$ then we can easily check a guess for $\mathbf{e}_{\mathscr{N}}$. Indeed, a guess $\mathbf{z}$ for $\mathbf{e}_{\mathscr{N}}$ can be verified in polynomial time by simply computing the unique codeword $\mathbf{c}$ of $\mathcal{C}$ which is such that $\mathbf{c}_{\mathscr{N}} = \mathbf{y}_{\mathscr{N}} - \mathbf{z}$ and then check that $\mathbf{y} - \mathbf{c}$ is of right Hamming weight t (the weight of the error). Importantly, denoting by $s \stackrel{\text{def}}{=} |\mathscr{P}|$, the setting where $\mathscr{N}$ is an information set with good probability, is when

$$n - s \geqslant k$$

which are trivially verified by all our parameters when the rate R of the code $\mathcal{C}$ is smaller than 0.5 as $s \leqslant k$. This is more than enough to account for the interesting parameters for which double-RLPN beats the ISDs. The checking algorithm is described in Algorithm 6.

Algorithm 6

Name: CHECKING-$\mathbf{e}_{\mathscr{N}}(\mathcal{G}, \mathcal{C}, \mathbf{y}, \mathscr{P}, \mathscr{N})$
Input: $\mathcal{G}, \mathcal{C}, \mathbf{y}, \mathscr{P}, \mathscr{N}$
Parameter: t

while $\mathbf{e}_{\mathscr{N}}^{(\mathbf{z})} \in \mathcal{G}$ **do**
    $\mathbf{c}_{\mathscr{N}} \leftarrow \mathbf{y}_{\mathscr{N}} - \mathbf{e}_{\mathscr{N}}^{(\mathbf{z})}$
    $\mathbf{c} \leftarrow \text{Lift}(\mathcal{C},\ \mathscr{N},\ \mathbf{c}_{\mathscr{N}})$
    if $|\mathbf{y} - \mathbf{c}| = t$ **then**
        $\mathbf{e} \leftarrow \mathbf{y} - \mathbf{c}$
        return $\mathbf{e}$

It is readily seen that we have that if the guess for $\mathbf{e}_{\mathscr{N}}$ related to the secret $\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\text{aux}}^{\intercal}$ is good, namely if $G(\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\text{aux}}^{\intercal}) = \mathbf{e}_{\mathscr{N}}$ and if $\mathscr{N}$ contains an information set of $\mathcal{C}$ then CHECKING-$\mathbf{e}_{\mathscr{N}}$ outputs $\mathbf{e}$.

6.3 Discussion

Note that in the case of double-RLPN the secret of the LPN samples is not $\mathbf{e}_{\mathscr{P}}$ but rather some linear combination of $\mathbf{e}_{\mathscr{P}}$, say $\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\text{aux}}^{\intercal}$ where $\mathbf{G}_{\text{aux}}$ is the generator matrix of the code used in the reduction from sparse − LPN to plain − LPN. Consequently, given $\mathbf{z}$ a candidate for $\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\text{aux}}^{\intercal}$ and an associated guess for $\mathbf{e}_{\mathscr{N}}$, we cannot easily verify if this couple is indeed the solution to our decoding problem as we could have done if we had access to $\mathbf{e}_{\mathscr{P}}$. Notably, trying to recover $\mathbf{e}_{\mathscr{P}}$ from $\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\text{aux}}^{\intercal}$ would be exponentially harmful in here as we observe that in practice our double-RLPN optimal parameters are such that this compression of $\mathbf{e}_{\mathscr{P}}$ is extremely lossy, namely the Hamming sphere in which $\mathbf{e}_{\mathscr{P}}$ lives is exponentially larger than this arrival space. We believe there are workarounds for this issue, but we would rather use the previous strategy to keep the algorithm simple.

7 Instantiation with a juxtaposition codes

Goal. Now, let us instantiate our algorithm with a specific family of $[s, k_{\text{aux}}]$-linear code $\mathcal{F}$ that are used in the reduction from sparse LPN to plain LPN : the auxiliary code $\mathcal{C}_{\text{aux}}$ is drawn uniformly at random in $\mathcal{F}$. The codes and its decoder must be as efficient as possible : given $\mathbf{h}_{\mathscr{P}}$ one wants that the decoder returns a codeword $\mathbf{c}_{\text{aux}} \in \mathcal{C}_{\text{aux}}$ at small distance t_{aux} possible while having a small decoding time the decoding distance, *i.e.* the weight of $\mathbf{h}_{\mathscr{P}} - \mathbf{c}_{\text{aux}}$, crucially intervene in the noise of the generated LPN samples. Information theory says that the best we could hope for is finding an auxiliary codeword at the Gilbert-Varshamov distance $d_{\text{GV}}(s, k_{\text{aux}})$ for a proportion $1 - o(1)$ of the words of the space $\mathbb{F}_2^s$.

Codes/Decoder used in [CDMT24]. Forgetting about the question of actually having a decoder running in polynomial time, this could be achieved with a random code. In fact, in double-RLPN the analysis was carried out when $\mathcal{C}_{\text{aux}}$ is taken uniformly at random among $[s, k_{\text{aux}}]$-linear codes and it was supposed to be equipped with an ideal genie-aided decoder that finds all the codeword at distance t_{aux} and where each codeword found in time poly (n). In [CDMT24,] it was also proposed to use a Cartesian product (what we call juxtaposition codes) of $b = s/\log_2(s)$ smaller random linear codes $\mathcal{C}_{\text{aux}} = \mathcal{C}^{(1)} \times \mathcal{C}^{(2)} \times \cdots \times \mathcal{C}^{(b)}$, each of length $s/\log_2(s)$ and dimension $k_{\text{aux}}/\log_2(s)$.

Definition 7.1 (Juxtaposition code). *We define the set of juxtaposition codes with b blocks, and of length n and dimension k, namely $\mathfrak{C}^{\text{juxt}}[b,\ n,\ k]$, as the set of linear codes $\mathcal{C}$ such that there exists b linear codes, $(\mathcal{C}^{(i)})_{i \in [\![1,\ b]\!]}$ that are such that for every $i \in [\![1,\ b]\!]$, $\mathcal{C}^{(i)}$ is an $[n^{(i)}, k^{(i)}]$-linear code and such that*

$$\mathcal{C} = \mathcal{C}^{(1)} \times \mathcal{C}^{(2)} \times \cdots \times \mathcal{C}^{(b)}$$

where we denote implicitly (in b), for each integer $v \in \mathbb{N}$ its i'th part as:

$$v^{(i)} \stackrel{def}{=} \begin{cases} \lfloor v/b \rfloor + 1 & \text{if } i \leqslant (v \mod b) \\ \lfloor v/b \rfloor & \text{else} \end{cases} \tag{6}$$

Definition 7.2. *When the context is clear we will implicitly denote the i'th part of a vector $\mathbf{x} \in \mathbb{F}_2^n$ relative to the support given by n by $\mathbf{x}^{(i)} \stackrel{def}{=} \mathbf{x}_{\mathscr{I}}$ where $\mathscr{I} \stackrel{def}{=} [\![\sum_{j=1}^{i-1} n^{(j)}, \sum_{j=1}^{i} n^{(j)}]\!]$. In the same manner, given $\mathcal{C} \in \mathfrak{C}^{\text{juxt}}[b,\ n,\ k]$ we denote by $\mathcal{C}^{(i)}$ its i'th constituent code.*

Decoding $\mathbf{h}_{\mathscr{P}}$ is done by enumerating the codewords of each code and decoding independently on each part. It was shown overall that this incurs only a loss of order $2^{o(n)}$ in double-RLPN compared to the ideal case where a random code with a genie-aided decoder is used.

The decoder we use here. Here we use a decoder that was used by [GJL14] based on independent blockwise enumeration. The base observation that even if $\mathcal{C}_{\text{aux}}$ is a random $[s, k_{\text{aux}}]$-linear code, it can be decoded at distance t_{aux} in polynomial amortized time as long as number total number of calls, to the decoders is superior to the number of admissible errors of good weight $\mathbf{e}_{\text{aux}} \in \mathcal{S}^s_{t_{\text{aux}}}$, namely $\binom{t_{\text{aux}}}{s}$. In this case one can simply create a syndrome table $\{ (\mathbf{H}_{\text{aux}}\mathbf{e}_{\text{aux}},\ \mathbf{e}_{\text{aux}}) \ :\ \mathbf{e}_{\text{aux}} \in \mathcal{S}^s_{t_{\text{aux}}}\}$ once and for all and, when one wants to decode $\mathbf{a} \in \mathbb{F}_2^s$ onto $\mathcal{C}_{\text{aux}}$ at distance t_{aux}, the decoders look up the table and returns the $\mathbf{e}_{\text{aux}}$ that are such that $\mathbf{H}_{\text{aux}}\mathbf{a} = \mathbf{H}_{\text{aux}}\mathbf{e}_{\text{aux}}$. This idea can be extended by considering a juxtaposition code with b block and decoding each code independently at distance t_{aux}/b with this error enumeration technique. Basically the decoder returns the following set of errors.

Definition 7.3 (Set of admissible errors). *Let $b, n, k, t \in \mathbb{N}$. Let $\mathcal{C} \in \mathfrak{C}^{\text{juxt}}[b,\ n,\ k]$ be a juxtaposition code of length n and dimension k. Let $\mathbf{y} \in \mathbb{F}_2^n$, we define the set of admissible errors as*

$$Dec^{\text{juxt}}(\mathcal{C}, \mathbf{y}, t) \stackrel{def}{=} \{\mathbf{e} \in \mathcal{S}^n_t \ :\ \left|\mathbf{e}^{(i)}\right| = t^{(i)} \text{ and } \mathbf{y}^{(i)} - \mathbf{e}^{(i)} \in \mathcal{C}^{(i)},\ \forall i \in [\![1,\ b]\!]\}.$$

We have in particular that

$$Dec^{\text{juxt}}(\mathcal{C}, \mathbf{y}, t) \subset \{\mathbf{e} \in \mathcal{S}^n_t \ :\ \mathbf{e} + \mathbf{y} \in \mathcal{C}\}.$$

This is particularly useful when one has to decode an exponential number of vectors, say $2^{\lambda s}$ with $\lambda > 0$. The point being that there exists a constant b such that the cost of computing the syndrome table is $\text{poly}(n)\binom{s/b}{t_{\text{aux}}/b} = \text{poly}(n)\sqrt[b]{\binom{s}{t_{\text{aux}}}}$ is smaller than $2^{\lambda s}$. The fact that b is constant allows one to argue that

$$\mathbb{E}\left(\left|\text{Dec}^{\text{juxt}}(\mathcal{C}, \mathbf{y}, t)\right|\right) = \tilde{\Omega}\left(\frac{\binom{t_{\text{aux}}}{s}}{2^{s-k_{\text{aux}}}}\right).$$

All in all, here is our provable-double-RLPN algorithm instantiated with this family of code and this decoder.

Definition 7.4 (Provable-DoubleRLPN with juxtaposition codes). *We define an instantiation of Algorithm 4 where we have an additionnal parameter $b \in \mathbb{N}$ and where:*

- *The family $\mathcal{F} \subset \mathfrak{C}[s, k_{\text{aux}}]$ of auxiliary codes is defined as*

$$\mathcal{F} = \mathfrak{C}^{\text{juxt}}[b,\ s,\ k_{\text{aux}}]$$

- *After having drawn the code $\mathcal{C}_{\text{aux}}$ from $\mathcal{F}$, compute and store for each $i \in [\![1, b]\!]$ an hash table $T^{(i)}$ indexed by syndromes, namely $T^{(i)}[\mathbf{s}] = \{\mathbf{e}^{(i)}_{\text{aux}} \in \mathcal{S}^{s^{(i)}}_{t^{(i)}_{\text{aux}}} \ :\ {\mathbf{H}_{\text{aux}}}^{(i)}\mathbf{e}^{(i)}_{\text{aux}} = \mathbf{s}^{(i)}\ \}$ where ${\mathbf{H}_{\text{aux}}}^{(i)}$ is a parity-check matrix of $\mathcal{C}^{(i)}_{\text{aux}}$.*

– *A call to* $\textsc{Decode}(\mathcal{C}_{\mathrm{aux}}, \mathbf{a},\ t_{\mathrm{aux}})$ *returns* $Dec^{\mathrm{juxt}}(\mathcal{C}, \mathbf{a}, t_{\mathrm{aux}})$ *by looking up the previous syndrome tables.*

Importantly, our regime of interests here will be when $R, t = \Theta(n)$, which corresponds to the regime in which the results of [CDMT24] were given. In this case, the optimal parameters of double-RLPN are such that $s, k_{\mathrm{aux}}, t_{\mathrm{aux}} = \Theta(s)$ and, in particular, the complexity of an iteration of double-RLPN is exponential in n and at least equal to the complexity of the FFT given by $k_{\mathrm{aux}}\, 2^{k_{\mathrm{aux}}}$. Basically, this means that there exists a constant b such that $\sqrt[b]{\binom{s}{t_{\mathrm{aux}}}} < 2^{k_{\mathrm{aux}}}$, making the syndrome table creation step negligible compared to the cost of the FFT.

We directly have the following complexity result.

Proposition 7.5 (Complexity of fully provable double-RLPN with juxtaposition codes). *Let n and let $k, t, s, k_{\mathrm{aux}}, t_{\mathrm{aux}}, N_{\mathrm{iter}}, b$ be implicit functions of n. Suppose that $b = \lceil s/k_{\mathrm{aux}} \rceil$ and that $\binom{s}{t_{\mathrm{aux}}}/2^{s-k_{\mathrm{aux}}} \in \mathcal{O}(1)$. Then, given an instance of $\mathrm{DP}_{\mathbf{G}}(n, k, t)$, the expected time and memory complexity of Definition 7.4 is given by*

$$\textbf{\textit{Time}} = \widetilde{\mathcal{O}}\big(N_{\mathrm{iter}}\left(T_{\mathrm{eq}} + 2^{k_{\mathrm{aux}}}\right)\big), \qquad \textbf{\textit{Memory}} = \widetilde{\mathcal{O}}\big(\left(M_{\mathrm{eq}} + 2^{k_{\mathrm{aux}}}\right)\big)$$

where T_{eq} and M_{eq} are respectively the expected time and memory complexity of one call to Compute-Small-Codewords.

8 Analysis

In this section we prove the main Theorem 5.1 by analyzing our provable variant of double-RLPN instantiated with juxtaposition code. Forgetting about the other technical details that we will deal with later in the rigorous proof, in essence, proving our main theorem, mainly relies on proving that, we have with probability $1 - o(1)$ that when $\mathbf{x}$ is the secret, namely $\mathbf{x} = \mathbf{e}_{\mathscr{P}}\mathbf{G}_{\mathrm{aux}}^{\intercal}$, then the guess $G(\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\mathrm{aux}}^{\intercal})$ is equal to $\mathbf{e}_{\mathscr{N}}$. By construction of the guess, see Definition 6.3, it is sufficient to prove that with probability $1 - o(1/n)$ we have that

$$F_{\mathcal{L}(\mathbf{y},\mathscr{H})}(\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\mathrm{aux}}^{\intercal}) > F_{\mathcal{L}\left(\mathbf{y}^{[i]},\mathscr{H}\right)}(\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\mathrm{aux}}^{\intercal}) \qquad \text{if } (\mathbf{e}_{\mathscr{N}})_i = 0, \tag{7}$$

$$F_{\mathcal{L}(\mathbf{y},\mathscr{H})}(\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\mathrm{aux}}^{\intercal}) < F_{\mathcal{L}\left(\mathbf{y}^{[i]},\mathscr{H}\right)}(\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\mathrm{aux}}^{\intercal}) \qquad \text{if } (\mathbf{e}_{\mathscr{N}})_i = 1. \tag{8}$$

The result could then be concluded with a union bound on $i \in [\![1,\, n-s]\!]$. Our proof mainly relies on the two following propositions. The first is the known second-order behavior of the score function, which was already devised in [CDMT24, Proposition 2] but that we i) generalize by stating when a juxtaposition code is used as an auxiliary code and ii) by deleting some unnecessary constraints by changing the proof.

Proposition 8.1. *There exists a positive poly-bounded function f_0 such that for any $t, k, s, k_{\mathrm{aux}}, t_{\mathrm{aux}}, w, u, b \in \mathbb{N}$ implicit functions of n such that $\frac{\prod_{j=1}^{b}\binom{s^{(j)}}{(j)}_{t_{\mathrm{aux}}}}{2^{s-k_{\mathrm{aux}}}}/2^{s-k_{\mathrm{aux}}} \in \mathcal{O}(1)$ then with probability $1 - \mathcal{O}\big(1/n^2\big)$ we have that*

$$F_{\mathcal{L}(\mathbf{y},\mathscr{H})}(\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\mathrm{aux}}^{\intercal}) \in [N\delta \pm f_0(n^b)\sqrt{N}]$$

where

$$N \stackrel{def}{=} N_{\text{eq}} N_{\text{aux}}, \qquad N_{\text{eq}} \stackrel{def}{=} \frac{\binom{n-s}{w}}{2^{k-s}}, \qquad N_{\text{aux}} \stackrel{def}{=} \frac{\prod_{j=1}^{b} \binom{s^{(j)}}{t_{\text{aux}}^{(j)}}}{2^{s-k_{\text{aux}}}}$$

and

$$\delta \stackrel{def}{=} \delta_w^{(n-s)}\left(|\mathbf{e}_{\mathscr{N}}|\right) \prod_{j=1}^{b} \delta_{t_{\text{aux}}^{(j)}}^{(s^{(j)})}\left(\left|\mathbf{e}_{\mathscr{P}}^{(i)}\right|\right)$$

moreover $\mathbb{E}\left(F_{\mathcal{L}(\mathbf{y},\mathscr{H})}\left(\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\text{aux}}^{\intercal}\right)\right) = N\delta$ *and* $N = \mathbb{E}\left(|\mathscr{H}|\right)$*. The distributions and quantities considered are as follows.*

- $\mathscr{N}$ *and* $\mathscr{P}$ *are two fixed complementary subsets of* $[\![1,\, n]\!]$ *of size* s *and* $n-s$ *respectively.*
- $\mathcal{C}$ *is chosen by taking its generator matrix* $\mathbf{G} \in \mathbb{F}_2^{k\times n}$ *uniformly at random in* $\{\ \mathbf{G} \in \mathbb{F}_2^{k\times n}\ :\ \text{rank}\left(\mathbf{G}_{\mathscr{P}}\right) = s\}$ *and* $\mathcal{C}_{\text{aux}}$ *is chosen uniformly at random in* $\mathfrak{C}^{\text{juxt}}\left[b,\, s,\, k_{\text{aux}}\right]$*, the set of juxtaposition codes with* b *blocks, length* s *and dimension* k_{aux}*.*
- $\mathbf{y} = \mathbf{c} + \mathbf{e}$ *where* $\mathbf{c} \in \mathcal{C}$ *and* $\mathbf{e} \in \mathcal{S}_t^n$ *is a fixed vector.*
- *The set* $\mathscr{H}$ *of decoded dual vectors is*

$$\mathscr{H} \stackrel{def}{=} \{(\mathbf{h}, \mathbf{e}_{\text{aux}}) \in \mathcal{C}^{\perp} \times \mathcal{S}_s^{t_{\text{aux}}}\ :\ |\mathbf{h}_{\mathscr{N}}| = w,\ \forall i \in [\![1,\, b]\!],\ \left|\mathbf{e}_{\text{aux}}^{(i)}\right| = {t_{\text{aux}}}^{(i)}\}$$

  and where we recall that the power notation $\mathbf{x}^{(i)}$ *was defined in Definition 7.2.*
- *Where* $\mathcal{L}\left(\mathbf{y}, \mathscr{H}\right)$ *is defined in Notation 4.1 and where we recall that the score function* $F_{\mathcal{L}(\mathbf{y},\mathscr{H})}$ *is defined in Definition 4.2.*

This proposition is proved in Section 11.1.

Remark 8.2. This proposition is only a slight variant of [CDMT24, Proposition 2].

The other main part of the proof of our theorem relies on the following lemma that state essentially that the expected value of $F_{\mathcal{L}(\mathbf{y},\mathscr{H})}\left(\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\text{aux}}^{\intercal}\right)$ and that of $F_{\mathcal{L}\left(\mathbf{y}^{(i)},\mathscr{H}\right)}\left(\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\text{aux}}^{\intercal}\right)$ are essentially as far as the expected value of $F_{\mathcal{L}(\mathbf{y},\mathscr{H})}\left(\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\text{aux}}^{\intercal}\right)$ and 0.

Lemma 8.3. *There exists a positive poly-bounded function* f_1 *such that for any* $s, w, u \in \mathbb{N}$ *implicit functions of* $n \in \mathbb{N}$ *such that* $\text{Root}\left(K_w^{(n-s)}\right) - u \in \Omega\left(n-s\right)$ *we have that*

$$\delta_w^{(n-s)}\left(u-1\right) - \delta_w^{(n-s)}\left(u\right) \geqslant \frac{1}{f_1(n)}\delta_w^{(n-s)}\left(u\right),$$
$$\delta_w^{(n-s)}\left(u\right) - \delta_w^{(n-s)}\left(u+1\right) \geqslant \frac{1}{f_1(n)}\delta_w^{(n-s)}\left(u\right).$$

This lemma is proved in Section 11.2.

8.1 Rest of the proof.

Recall that our algorithm is iterated a number N_{iter} of times. In RLPN this was done to ensure that there existed at least one iteration such that $|\mathbf{e}_{\mathscr{N}}| = u$ where u is a parameter. Clearly, from Proposition 8.1 one can see that, because we used juxtaposition codes with b block, the distribution of the score function evaluated on the secret, $F_{\mathscr{H},\mathbf{y}}\left(\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\text{aux}}^{\intercal}\right)$, finely depends on the weight of $\mathbf{e}$ on each subpart, namely on $|\mathbf{e}_{\mathscr{N}}|$ and $\left|\mathbf{e}_{\mathscr{P}}^{(i)}\right|$ for each $i \in [\![1,\, b]\!]$.

We will make the additional, finer bet that $\left|\mathbf{e}^{(i)}_{\mathscr{P}}\right|$ is of typical weight roughly $(t-u)/b$. More precisely we will make the bet that

$$|\mathbf{e}_{\mathscr{N}}| = u \bigwedge_{i=1}^{b} \left|\mathbf{e}^{(i)}_{\mathscr{P}}\right| = (t-u)^{(i)} .$$

First, because b is constant, essentially this bet is verified with the same probability, up to polynomial factors, as the simpler bet that $|\mathbf{e}_{\mathscr{N}}| = w$. Namely, we have that

Lemma 8.4. *There exists a positive poly-bounded function g such that for any t, s, b implicit functions of n we have that*

$$\mathbb{P}\left(|\mathbf{e}_{\mathscr{N}}| = u \bigwedge_{i=1}^{b} \left|\mathbf{e}^{(i)}_{\mathscr{P}}\right| = (t-u)^{(i)}\right) \geqslant \frac{1}{g(n^b)} \frac{\binom{n-s}{u}\binom{s}{t-u}}{\binom{n}{t}}$$

where $\mathscr{N}$ and $\mathscr{P}$ are any fixed complementary subsets of $[\![1,\, n]\!]$ and $\mathbf{e}$ is taken uniformly at random in $\mathcal{S}^n_t$.

The proof is straightforward using Proposition 3.14. Moreover, using again the fact b is constant one can show easily that all the quantities appearing in Proposition 8.1, like N and the product of the biases, polynomially relates to the ideal case $b = 1$. More precisely we have the following.

Proposition 8.5 (Relating the quantities to the case $b = 1$). *There exists a positive poly-bounded function f_2 such that for any $k, t, s, u, t_{\mathrm{aux}}, b \in \mathbb{N}$ implicit functions of $n \in \mathbb{N}$ such that*

$$\mathrm{Root}\left(K^{(n-s)}_{w}\right) - u \in \Omega\left(n-s\right) \quad \text{and} \quad \forall i \in [\![1,\, b]\!]\ \mathrm{Root}\left(K^{(s)}_{t_{\mathrm{aux}}}\right) - (t-u) \in \Omega\left(s\right)$$

then we have that

$$\prod_{j=1}^{b} \delta^{(s^{(j)})}_{t^{(j)}_{\mathrm{aux}}}\left(\left|(t-u)^{(i)}\right|\right) \in \Omega\left(\frac{1}{f_2(n^b)} \delta^{(s)}_{t_{\mathrm{aux}}}\left(t-u\right)\right),$$

$$\frac{\prod_{j=1}^{b}\binom{s^{(j)}}{t^{(j)}_{\mathrm{aux}}}}{2^{s-k_{\mathrm{aux}}}} \in \Omega\left(\frac{1}{f_2(n^b)} \frac{\binom{s}{t_{\mathrm{aux}}}}{2^{s-k_{\mathrm{aux}}}}\right).$$

The proof is straightforward using Proposition 3.14 and Proposition 3.15. Putting together all these propositions and lemmas yields our main Theorem 5.1. For the sake of completeness, we provide a detailed proof of Theorem 5.1 in Appendix 11.3.

9 Discussion

9.1 Variants of our provable attack

In this paper, we focused on the simplest (in terms of presentation) provable variant of double-RLPN but there are a lot of other variants that could be considered to minimize the polynomial overhead. Basically, the only step that would change in these variants is how we guess the i'th coordinate of $\mathbf{e}_{\mathscr{N}}$. Instead of having a bit-flipping approach one could rather

compute dual vectors $\mathbf{h}$ such that $\left|(\mathbf{h}_{\mathscr{N}})_{\setminus i}\right| = w$, where $\mathbf{x}_{\setminus i}$ is defined as the vectors $\mathbf{x}$ where we deleted the i'th coordinate. We could then leverage the fact that the noise $\langle \mathbf{e}_{\mathscr{N}}, \mathbf{h}_{\mathscr{N}} \rangle$ related to the sample $\langle \mathbf{y}, \mathbf{h} \rangle$ is either biased toward 0 if $(\mathbf{e}_{\mathscr{N}})_i = 0$ and is non-biased else. As such, we would guess that $(\mathbf{e}_{\mathscr{N}})_i$ is equal to 1 if the score function associated with the samples $\langle \mathbf{y}, \mathbf{h} \rangle$ is superior to a certain well-chosen threshold. One could also devise a variant where it is the coordinates of the codeword $\mathbf{c}$ that are guessed. In a practical scenario, one only has to guess k coordinate of the error or the codeword.

9.2 For proving lattice-based dual attacks

We argue that both lattice- and code-based attacks can be interpreted in the same framework and are very close. In particular, we disagree with the statements in [PS24, Introduction, Appendix A] about them being different. Showing the similarities also allows to argue simply that we could adapt our strategy to prove lattice-based dual attacks.

Let us focus on the attack of [MAT22, CMST25], a different argument could be held for [GJ21]. The target in these papers is to solve the following "sparse"-secret LWE where the secret and the error follow the same distribution, as it is the case for Kyber. The problem is as follows given $(\mathbf{A}, \mathbf{s} = \mathbf{s}\mathbf{A} + \mathbf{e}')$ where $\mathbf{A}$ is taken uniformly in $\mathbb{Z}_q^{k\times(n-k)}$ and where the secret $\mathbf{s} \in \mathbb{Z}_q^k$ and $\mathbf{e}' \in \mathbb{Z}_q^{n-k}$ are two vectors whose coordinate are i.i.d distirbuted according to given distribution $\mathcal{D}$ that is biased toward the small values of $\mathbb{Z}_q$, the goal is to recover $\mathbf{s}$. It is standard that this problem is equivalent to the following problem : given $(\mathbf{G}, \mathbf{y} = \mathbf{m}\mathbf{G} + \mathbf{e})$ where $\mathbf{G}$ is taken uniformly among matrices of $\mathbb{Z}_q^{k\times n}$ that are of rank k and $\mathbf{m}$ is taken uniformly in $\mathbb{F}_2^k$ and the coordinate of $\mathbf{e}$ are taken i.i.d according to $\mathcal{D}$. Slightly rewriting both attacks [MAT22, CMST25], one can see that they are exactly computing two complementary subsets $\mathscr{P}$ and $\mathscr{N}$ and computing vectors $\mathbf{h}$ in the dual of the module generated by $\mathbf{G}$ that are small on the part $\mathscr{N}$. Each of these vectors gives the following LWE sample

$$\langle \mathbf{y}, \mathbf{h} \rangle = \langle \mathbf{h}_{\mathscr{P}}, \mathbf{e}_{\mathscr{P}} \rangle + \langle \mathbf{e}_{\mathscr{N}}, \mathbf{h}_{\mathscr{N}} \rangle$$

where the secret is given by $\mathbf{e}_{\mathscr{P}}$. In [MAT22] this LWE problem is solved by using a modulus switching technique and an FFT and in [CMST25] with a combination of a decoding technique and an FFT. All in all, the main difference with code-based dual attacks lies in the fact that the dual vectors $\mathbf{h}$ are computed using lattice tools, this is usually done by embedding the module into a lattice $\Lambda = \{\mathbf{m}\mathbf{G} : \mathbf{m} \in \mathbb{Z}_q^k\} + q\mathbb{Z}^n$.

10 Conclusion

In this work we have presented a variant of the most recent code-based dual attack, double-RLPN, but that we can fully prove without using any heuristics up to rate $R \leqslant 0.5$ and that has the same performances, up to polynomial factors, as the original algorithm. We also argued why we believe this proof strategy could also be adapted to lattice-based dual attacks.

References

AAB+22a. Carlos Aguilar Melchor, Nicolas Aragon, Paulo Barreto, Slim Bettaieb, Loïc Bidoux, Olivier Blazy, Jean-Christophe Deneuville, Philippe Gaborit, Santosh Ghosh, Shay

Gueron, Tim Güneysu, Rafael Misoczki, Edoardo Persichetti, Jan Richter-Brockmann, Nicolas Sendrier, Jean-Pierre Tillich, Valentin Vasseur, and Gilles Zémor. BIKE. Round 4 Submission to the NIST Post-Quantum Cryptography Call, v. 5.1, October 2022.

AAB+22b. Carlos Aguilar Melchor, Nicolas Aragon, Slim Bettaieb, Loïc Bidoux, Olivier Blazy, Jurjen Bos, Jean-Christophe Deneuville, Arnaud Dion, Philippe Gaborit, Jérôme Lacan, Edoardo Persichetti, Jean-Marc Robert, Pascal Véron, Gilles Zémor, and Jurjen Bos. HQC. Round 4 Submission to the NIST Post-Quantum Cryptography Call, October 2022. `https://pqc-hqc.org/`.

ABC+22. Martin Albrecht, Daniel J. Bernstein, Tung Chou, Carlos Cid, Jan Gilcher, Tanja Lange, Varun Maram, Ingo von Maurich, Rafael Mizoczki, Ruben Niederhagen, Edoardo Persichetti, Kenneth Paterson, Christiane Peters, Peter Schwabe, Nicolas Sendrier, Jakub Szefer, Cen Jung Tjhai, Martin Tomlinson, and Wang Wen. Classic McEliece (merger of Classic McEliece and NTS-KEM). `https://classic.mceliece.org`, November 2022. Fourth round finalist of the NIST post-quantum cryptography call.

Alb17. Martin R. Albrecht. On dual lattice attacks against small-secret LWE and parameter choices in HElib and SEAL. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *Advances in Cryptology - EUROCRYPT 2017 - 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Paris, France, April 30 - May 4, 2017, Proceedings, Part II*, volume 10211 of *Lecture Notes in Computer Science*, pages 103–129, 2017.

AMBB+24. Carlos Aguilar Melchor, Slim Bettaieb, Loïc Bidoux, Thibauld Feneuil, Philippe Gaborit, Nicolas Gama, Shay Gueron, James Howe, Andreas Hülsing, David Joseph, Antoine Joux, Mukul Kulkarni, Edoardo Persichetti, Tovohery Randrianarisoa, Matthieu Rivain, and Dongze Yue. SDitH. Round 2 Additional Signatures to the NIST Post-Quantum Cryptography: Digital Signature Schemes Call, May 2024.

AR05. Dorit Aharonov and Oded Regev. Lattice problems in NP cap coNP. *J. ACM*, 52(5):749–765, 2005.

BDGL16. Anja Becker, Léo Ducas, Nicolas Gama, and Thijs Laarhoven. New directions in nearest neighbor searching with applications to lattice sieving. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '16, page 10–24, USA, 2016. Society for Industrial and Applied Mathematics.

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.

BM18. Leif Both and Alexander May. Decoding linear codes with high error rate and its impact for LPN security. In Tanja Lange and Rainer Steinwandt, editors, *Post-Quantum Cryptography 2018*, volume 10786 of *LNCS*, pages 25–46, Fort Lauderdale, FL, USA, April 2018. Springer.

Car20. Kevin Carrier. *Recherche de presque-collisions pour le décodage et la reconnaissance de codes correcteurs*. Theses, Sorbonne Université, June 2020.

CDMT22. Kevin Carrier, Thomas Debris-Alazard, Charles Meyer-Hilfiger, and Jean-Pierre Tillich. Statistical decoding 2.0: Reducing decoding to LPN. In *Advances in Cryptology - ASIACRYPT 2022*, LNCS. Springer, 2022.

CDMT24. Kévin Carrier, Thomas Debris-Alazard, Charles Meyer-Hilfiger, and Jean-Pierre Tillich. Reduction from sparse LPN to LPN, dual attack 3.0. In Marc Joye and Gregor Leander, editors, *Advances in Cryptology - EUROCRYPT 2024 - 43rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zurich, Switzerland, May 26-30, 2024, Proceedings, Part VI*, volume 14656 of *LNCS*, pages 286–315. Springer, 2024. Artifact available at `https://artifacts.iacr.org/eurocrypt/2024/a10/`.

CMST25. Kevin Carrier, Charles Meyer-Hilfiger, Yixin Shen, and Jean-Pierre Tillich. Assessing the impact of a variant of matzov's dual attack on kyber. Cryptology ePrint Archive, Paper 2022/1750, 2025. `https://eprint.iacr.org/2022/1750`. Fourth version of the 2022 preprint.

DP23. Léo Ducas and Ludo N. Pulles. Does the dual-sieve attack on learning with errors even work? In Helena Handschuh and Anna Lysyanskaya, editors, *Advances in Cryptology - CRYPTO 2023*, volume 14083 of *LNCS*, pages 37–69, Santa Barbara, CA, USA, August 2023. Springer.

DP26. Léo Ducas and Ludo N. Pulles. Accurate score prediction for dual-sieve attacks. *Journal of cryptology*, 39(9), 2026.

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

Dum89. Il'ya Dumer. Two decoding algorithms for linear codes. *Probl. Inf. Transm.*, 25(1):17–23, 1989.

EJK20. Thomas Espitau, Antoine Joux, and Natalia Kharchenko. On a dual/hybrid approach to small secret LWE - A dual/enumeration technique for learning with errors and application to security estimates of FHE schemes. In Karthikeyan Bhargavan, Elisabeth Oswald, and Manoj Prabhakaran, editors, *Progress in Cryptology - INDOCRYPT 2020 - 21st International Conference on Cryptology in India, Bangalore, India, December 13-16, 2020, Proceedings*, volume 12578 of *Lecture Notes in Computer Science*, pages 440–462. Springer, 2020.

GJ21. Qian Guo and Thomas Johansson. Faster dual lattice attacks for solving LWE with applications to CRYSTALS. In Mehdi Tibouchi and Huaxiong Wang, editors, *Advances in Cryptology - ASIACRYPT 2021 - 27th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 6-10, 2021, Proceedings, Part IV*, volume 13093 of *Lecture Notes in Computer Science*, pages 33–62. Springer, 2021.

GJL14. Qian Guo, Thomas Johansson, and Carl Löndahl. Solving LPN using covering codes. In *Advances in Cryptology - ASIACRYPT 2014*, volume 8873 of *LNCS*, pages 1–20. Springer, 2014.

Jab01. Abdulrahman Al Jabri. A statistical decoding algorithm for general linear block codes. In Bahram Honary, editor, *Cryptography and coding. Proceedings of the 8th IMA International Conference*, volume 2260 of *LNCS*, pages 1–8, Cirencester, UK, December 2001. Springer.

KS21. Naomi Kirshner and Alex Samorodnitsky. A moment ratio bound for polynomials and some extremal properties of krawchouk polynomials and hamming spheres. *IEEE Trans. Inform. Theory*, 67(6):3509–3541, 2021.

LF06. Éric Levieil and Pierre-Alain Fouque. An improved LPN algorithm. In *Proceedings of the 5th international conference on Security and Cryptography for Networks*, volume 4116 of *LNCS*, pages 348–359. Springer, 2006.

MAT22. MATZOV. Report on the Security of LWE: Improved Dual Lattice Attack, April 2022.

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.

MT23. Charles Meyer-Hilfiger and Jean-Pierre Tillich. Rigorous foundations for dual attacks in coding theory. In *Theory of Cryptography Conference, TCC 2023*, volume 14372 of *LNCS*, pages 3–32. Springer Verlag, December 2023.

Ove06. Raphael Overbeck. Statistical decoding revisited. In Reihaneh Safavi-Naini Lynn Batten, editor, *Information security and privacy : 11th Australasian conference, ACISP 2006*, volume 4058 of *LNCS*, pages 283–294. Springer, 2006.

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

PS24. Amaury Pouly and Yixin Shen. Provable dual attacks on learning with errors. In Marc Joye and Gregor Leander, editors, *Advances in Cryptology - EUROCRYPT 2024 - 43rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zurich, Switzerland, May 26-30, 2024, Proceedings, Part VI*, volume 14656 of *LNCS*, pages 256–285. Springer, 2024.

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.

vL99. Jacobus Hendricus van Lint. *Introduction to coding theory*. Graduate texts in mathematics. Springer, 3rd edition edition, 1999.

11 Appendices

11.1 Proof of the second-order concentration bound

The goal of this section is to prove Proposition 8.1 that we recall here.

Proposition 8.1. *There exists a positive poly-bounded function f_0 such that for any $t, k, s, k_{\text{aux}}, t_{\text{aux}}, w, u, b \in \mathbb{N}$ implicit functions of n such that $\frac{\prod_{j=1}^{b}\binom{s^{(j)}}{t_{\text{aux}}^{(j)}}}{2^{s-k_{\text{aux}}}}/2^{s-k_{\text{aux}}} \in \mathcal{O}(1)$ then with probability $1-\mathcal{O}\left(1/n^2\right)$ we have that*

$$F_{\mathcal{L}(\mathbf{y},\mathscr{H})}\left(\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\text{aux}}^{\intercal}\right) \in [N\delta \pm f_0(n^b)\sqrt{N}]$$

where

$$N \stackrel{def}{=} N_{\text{eq}}N_{\text{aux}}, \qquad N_{\text{eq}} \stackrel{def}{=} \frac{\binom{n-s}{w}}{2^{k-s}}, \qquad N_{\text{aux}} \stackrel{def}{=} \frac{\prod_{j=1}^{b}\binom{s^{(j)}}{t_{\text{aux}}^{(j)}}}{2^{s-k_{\text{aux}}}}$$

and

$$\delta \stackrel{def}{=} \delta_w^{(n-s)}\left(|\mathbf{e}_{\mathscr{N}}|\right)\prod_{j=1}^{b}\delta_{t_{\text{aux}}^{(j)}}^{(s^{(j)})}\left(\left|\mathbf{e}_{\mathscr{P}}^{(i)}\right|\right)$$

moreover $\mathbb{E}\left(F_{\mathcal{L}(\mathbf{y},\mathscr{H})}\left(\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\text{aux}}^{\intercal}\right)\right) = N\delta$ and $N = \mathbb{E}\left(|\mathscr{H}|\right)$. The distributions and quantities considered are as follows.

- *$\mathscr{N}$ and $\mathscr{P}$ are two fixed complementary subsets of $[\![1, n]\!]$ of size s and $n-s$ respectively.*
- *$\mathcal{C}$ is chosen by taking its generator matrix $\mathbf{G} \in \mathbb{F}_2^{k\times n}$ uniformly at random in $\{\ \mathbf{G} \in \mathbb{F}_2^{k\times n}\ :\ \text{rank}\left(\mathbf{G}_{\mathscr{P}}\right) = s\}$ and $\mathcal{C}_{\text{aux}}$ is chosen uniformly at random in $\mathfrak{C}^{\text{juxt}}[b,\ s,\ k_{\text{aux}}]$, the set of juxtaposition codes with b blocks, length s and dimension k_{aux}.*
- *$\mathbf{y} = \mathbf{c} + \mathbf{e}$ where $\mathbf{c} \in \mathcal{C}$ and $\mathbf{e} \in \mathcal{S}_t^n$ is a fixed vector.*
- *The set $\mathscr{H}$ of decoded dual vectors is*

$$\mathscr{H} \stackrel{def}{=} \{(\mathbf{h}, \mathbf{e}_{\text{aux}}) \in \mathcal{C}^{\perp} \times \mathcal{S}_s^{t_{\text{aux}}}\ :\ |\mathbf{h}_{\mathscr{N}}| = w,\ \forall i \in [\![1,\ b]\!],\ \left|\mathbf{e}_{\text{aux}}^{(i)}\right| = {t_{\text{aux}}}^{(i)}\}$$

  and where we recall that the power notation $\mathbf{x}^{(i)}$ was defined in Definition 7.2.
- *Where $\mathcal{L}\left(\mathbf{y}, \mathscr{H}\right)$ is defined in Notation 4.1 and where we recall that the score function $F_{\mathcal{L}(\mathbf{y},\mathscr{H})}$ is defined in Definition 4.2.*

The main technical lemma that we will use here is the following giving the first two moments of the score function.

Lemma 11.1. *(Main technical lemma.) For any $k, t, s, u, w, k_{\text{aux}}, t_{\text{aux}}, b \in \mathbb{N}$ are implicit functions of $n \in \mathbb{N}$ we have that*

$$\mathbb{E}\left(F_{\mathcal{L}(\mathbf{y},\mathscr{H})}\left(\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\text{aux}}^{\intercal}\right)\right) = N\,\delta_w^{(n-s)}\left(|\mathbf{e}_{\mathscr{N}}|\right)\prod_{j=1}^{b}\delta_{t_{\text{aux}}^{(j)}}^{(s^{(j)})}\left(\left|\mathbf{e}_{\mathscr{P}}^{(i)}\right|\right)$$
$$\mathbf{Var}\left(F_{\mathcal{L}(\mathbf{y},\mathscr{H})}\left(\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\text{aux}}^{\intercal}\right)\right) = \mathcal{O}\big(n^{b+1}\,N\,\max\left(1,\,N_{\text{aux}}\right)\big)$$

Remark 11.2. We believe that the term n^{b+1} appearing in the variance is much more reasonable and is rather some $\mathcal{O}\big(2^b\big)$ in the constant rate regime.

Proof (Proof of Proposition 8.1). This is directly obtained by using Byenemé-Chebyshev inequality along with the moments given in Lemma 11.1.

Proving the second-order moments. Let us now prove Lemma 11.1. We first give a lemma giving the distribution of some related quantities that will appear in the proof of the second-order lemma.

Lemma 11.3 (Distribution of some related quantities). *We reuse the notation of Proposition 8.1 and define $\mathbf{R}$ as the unique matrix such that $\mathbf{h}_{\mathscr{P}} = \mathbf{h}_{\mathscr{N}}\mathbf{R}^{\top}$ for all $\mathbf{h} \in \mathcal{C}^{\perp}$. We have that*

$$\mathbf{R} \sim \mathcal{U}\left(\mathbb{F}_2^{s\times(n-s)}\right), \tag{9}$$
$$\left(\mathcal{C}^{\perp}\right)_{\mathscr{N}} \sim \mathcal{U}_{\mathbf{H}}\left(n-s,\,n-k\right), \tag{10}$$
$$\mathbf{R} \text{ and } \mathcal{C}^{\mathscr{N}} \text{ are independent.} \tag{11}$$

Proof. The second property is straightforward and the first and third can be shown by observing the way $\mathbf{R}$ is constructed in Eq. (1) along with the fact that by assumption in Proposition 8.1 the code $\mathcal{C}_{\mathscr{N}}$ is taken by choosing a generator matrix $\mathbf{G}_{\mathscr{N}}$ uniformly at random in $\mathbb{F}_2^{k\times(n-s)}$.

This allows us to prove this intermediate technical lemma on the moments of our variables.

Lemma 11.4. *We have that*

$$\mathbb{E}\left(F_{\mathscr{H},\mathbf{y}}\left(\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\text{aux}}^{\intercal}\right)\right) = N\,\delta_w^{(n-s)}\left(|\mathbf{e}_{\mathscr{N}}|\right)\prod_{j=1}^{b}\delta_{t_{\text{aux}}^{(j)}}^{(s^{(j)})}\left(\left|\mathbf{e}_{\mathscr{P}}^{(i)}\right|\right),$$
$$\mathbf{Var}\left(F_{\mathscr{H},\mathbf{y}}\left(\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\text{aux}}^{\intercal}\right)\right) \leqslant N\left[1+\sum_{\mathbf{c}\in\{0,1\}^b\,:\,\mathbf{c}\neq\mathbf{0}}\prod_{i=1}^{b}\left(\frac{\binom{s^{(i)}}{t_{\text{aux}}^{(i)}}}{2^{s^{(i)}-t_{\text{aux}}^{(i)}}}\right)^{c_i}\right]$$

Proof. Let us first compute the expected value. We will show that

$$\mathbb{E}\left(F_{\mathscr{H},\mathbf{y}}\left(\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\text{aux}}^{\intercal}\right)\right) = N\;\delta_w^{(n-s)}\left(|\mathbf{e}_{\mathscr{N}}|\right)\prod_{j=1}^{b}\delta_{t_{\text{aux}}^{(j)}}^{(s^{(j)})}\left(\left|\mathbf{e}_{\mathscr{P}}^{(i)}\right|\right).$$

Rewriting the score function we have that

$$F\left(\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\text{aux}}^{\intercal}\right)$$
$$= \sum_{\mathbf{h}_{\mathscr{N}}\in\mathcal{S}_w^{n-s}} \sum_{\substack{\mathbf{e}_{\text{aux}}^{(1)}\in\mathcal{S}_{t_{\text{aux}}^{(1)}}^{s^{(1)}}}} \cdots \sum_{\substack{\mathbf{e}_{\text{aux}}^{(b)}\in\mathcal{S}_{t_{\text{aux}}^{(b)}}^{s^{(b)}}}} (-1)^{\langle \mathbf{e}_{\mathscr{N}}, \mathbf{h}_{\mathscr{N}}\rangle} \left[\prod_{i=1}^{b}(-1)^{\left\langle \mathbf{e}_{\mathscr{P}}^{(i)}, \mathbf{e}_{\text{aux}}^{(i)}\right\rangle}\right] \mathbf{1}_{\mathbf{h}_{\mathscr{N}}\in\left(\mathcal{C}^{\perp}\right)_{\mathscr{N}}} \prod_{i=1}^{b} \mathbf{1}_{\mathbf{e}_{\text{aux}}^{(i)}+\left(\mathbf{h}_{\mathscr{N}}\mathbf{R}^{\intercal}\right)^{(i)}\in\mathcal{C}_{\text{aux}}^{(i)}} \tag{12}$$

By linearity of the expected value we have that

$$\mathbb{E}\left(F_{\mathscr{H},\mathbf{y}}\left(\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\text{aux}}^{\intercal}\right)\right) = \sum_{\mathbf{h}_{\mathscr{N}}\in\mathcal{S}_w^{n-s}} \sum_{\substack{\mathbf{e}_{\text{aux}}^{(1)}\in\mathcal{S}_{t_{\text{aux}}^{(1)}}^{s^{(1)}}}} \cdots \sum_{\substack{\mathbf{e}_{\text{aux}}^{(b)}\in\mathcal{S}_{t_{\text{aux}}^{(b)}}^{s^{(b)}}}}$$
$$\left[(-1)^{\langle \mathbf{e}_{\mathscr{N}}, \mathbf{h}_{\mathscr{N}}\rangle} \prod_{i=1}^{b}(-1)^{\left\langle \mathbf{e}_{\mathscr{P}}^{(i)}, \mathbf{e}_{\text{aux}}^{(i)}\right\rangle}\right] \left[\mathbb{E}\left(\mathbf{1}_{\mathbf{h}_{\mathscr{N}}\in\left(\mathcal{C}^{\perp}\right)_{\mathscr{N}}} \prod_{i=1}^{b} \mathbf{1}_{\mathbf{e}_{\text{aux}}^{(i)}+\left(\mathbf{h}_{\mathscr{N}}\mathbf{R}^{\intercal}\right)^{(i)}\in\mathcal{C}_{\text{aux}}^{(i)}}\right)\right] \tag{13}$$

Now from the independence of the indicator variable we get that

$$\begin{aligned}\mathbb{E}\left(\mathbf{1}_{\mathbf{h}_{\mathscr{N}}\in\left(\mathcal{C}^{\perp}\right)_{\mathscr{N}}} \prod_{i=1}^{b} \mathbf{1}_{\mathbf{e}_{\text{aux}}^{(i)}+\left(\mathbf{h}_{\mathscr{N}}\mathbf{R}^{\intercal}\right)^{(i)}\in\mathcal{C}_{\text{aux}}^{(i)}}\right) &= \mathbb{E}\left(\mathbf{1}_{\mathbf{h}_{\mathscr{N}}\in\left(\mathcal{C}^{\perp}\right)_{\mathscr{N}}}\right) \prod_{i=1}^{b} \mathbb{E}\left(\mathbf{1}_{\mathbf{e}_{\text{aux}}^{(i)}+\left(\mathbf{h}_{\mathscr{N}}\mathbf{R}^{\intercal}\right)^{(i)}\in\mathcal{C}_{\text{aux}}^{(i)}}\right) \\ &= \mathbb{P}\left(\mathbf{h}_{\mathscr{N}} \in \left(\mathcal{C}^{\perp}\right)_{\mathscr{N}}\right) \prod_{i=1}^{b} \mathbb{P}\left(\mathbf{e}_{\text{aux}}^{(i)} + \left(\mathbf{h}_{\mathscr{N}}\mathbf{R}^{\intercal}\right)^{(i)} \in \mathcal{C}_{\text{aux}}^{(i)}\right).\end{aligned} \tag{14}$$

From Lemma 11.3 we have that $\left(\mathcal{C}^{\perp}\right)_{\mathscr{N}} \sim \mathcal{U}_{\mathbf{H}}\left(n-s,\ n-k\right)$ thus using Lemma 3.9 we get

$$\mathbb{P}\left(\mathbf{h}_{\mathscr{N}} \in \left(\mathcal{C}^{\perp}\right)_{\mathscr{N}}\right) = \frac{1}{2^{k-s}}.$$

From Lemma 11.3 we have that for each $i \in [\![1,\ b]\!]$, $\mathbf{e}_{\text{aux}}^{(i)} + \left(\mathbf{h}_{\mathscr{N}}\mathbf{R}^{\intercal}\right)^{(i)} \sim \mathcal{U}\left(\mathbb{F}_2^{s^{(i)}}\right)$, thus as $\mathcal{C}_{\text{aux}}^{(i)}$ is an $[s^{(i)}, k_{\text{aux}}^{(i)}]-$linear code we get that

$$\mathbb{P}\left(\mathbf{e}_{\text{aux}}^{(i)} + \left(\mathbf{h}_{\mathscr{N}}\mathbf{R}^{\intercal}\right)^{(i)} \in \mathcal{C}_{\text{aux}}^{(i)}\right) = \frac{1}{2^{s^{(i)}-k_{\text{aux}}^{(i)}}}. \tag{15}$$

Plugging these last equation into Eq. (14) yield that

$$\begin{aligned}\mathbb{E}\left(\mathbf{1}_{\mathbf{h}_{\mathscr{N}}\in\left(\mathcal{C}^{\perp}\right)_{\mathscr{N}}} \prod_{i=1}^{b} \mathbf{1}_{\mathbf{e}_{\text{aux}}^{(i)}+\left(\mathbf{h}_{\mathscr{N}}\mathbf{R}^{\intercal}\right)^{(i)}\in\mathcal{C}_{\text{aux}}^{(i)}}\right) &= \frac{1}{2^{k-s}} \prod_{i=1}^{b} \frac{1}{2^{s^{(i)}-k_{\text{aux}}^{(i)}}} \\ &= \frac{1}{2^{k-k_{\text{aux}}}}\end{aligned} \tag{16}$$

where in the last equation we used the fact that by definition of the i'th part of a vector in Eq. (6) we have that

$$\sum_{i=1}^{b} s^{(i)} = s, \qquad \sum_{i=1}^{b} k_{\text{aux}}^{(i)} = k_{\text{aux}}.$$

Finally, plugging this last equality back into Eq. (13) gives that

$$\begin{aligned}
\mathbb{E}\left(F\left(\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\text{aux}}^{\intercal}\right)\right) &= \frac{1}{2^{k-k_{\text{aux}}}} \sum_{\mathbf{h}_{\mathscr{N}} \in \mathcal{S}_w^{n-s}} \sum_{\substack{\mathbf{e}_{\text{aux}}^{(1)} \in \mathcal{S}_{t_{\text{aux}}^{(1)}}^{s^{(1)}}}} \cdots \sum_{\substack{\mathbf{e}_{\text{aux}}^{(b)} \in \mathcal{S}_{t_{\text{aux}}^{(b)}}^{s^{(b)}}}} (-1)^{\langle \mathbf{e}_{\mathscr{N}}, \mathbf{h}_{\mathscr{N}} \rangle} \prod_{i=1}^{b} (-1)^{\left\langle \mathbf{e}_{\mathscr{P}}^{(i)}, \mathbf{e}_{\text{aux}}^{(i)} \right\rangle} \\
&= \frac{1}{2^{k-k_{\text{aux}}}} K_w^{(n-s)}\left(|\mathbf{e}_{\mathscr{N}}|\right) \prod_{i=1}^{b} K_{t_{\text{aux}}^{(i)}}^{(s^{(i)})}\left(\left|\mathbf{e}_{\mathscr{P}}^{(i)}\right|\right) \\
&= \frac{\binom{n-s}{w} \prod_{i=1}^{b} \binom{s^{(i)}}{t_{\text{aux}}^{(i)}}}{2^{k-k_{\text{aux}}}} \frac{K_w^{(n-s)}\left(|\mathbf{e}_{\mathscr{N}}|\right) \prod_{i=1}^{b} K_{t_{\text{aux}}^{(i)}}^{(s^{(i)})}\left(\left|\mathbf{e}_{\mathscr{P}}^{(i)}\right|\right)}{\binom{n-s}{w} \prod_{i=1}^{b} \binom{s^{(i)}}{t_{\text{aux}}^{(i)}}} \\
&= N \delta_w^{(n-s)}\left(|\mathbf{e}_{\mathscr{N}}|\right) \prod_{i=1}^{b} \delta_{t_{\text{aux}}^{(i)}}^{(s^{(i)})}\left(\left|\mathbf{e}_{\mathscr{P}}^{(i)}\right|\right)
\end{aligned}$$

where in the last lines we used the definition of δ in Lemma 3.13.

Let us now compute the variance of $F\left(\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\text{aux}}^{\intercal}\right)$. Starting again from Equation (12) and denoting by $A \in \{-1, 1\}$ the value:

$$A\left(\mathbf{h}_{\mathscr{N}},\ \mathbf{e}_{\text{aux}}^{(1)},\ \ldots,\ \mathbf{e}_{\text{aux}}^{(b)}\right) \stackrel{\text{def}}{=} (-1)^{\langle \mathbf{e}_{\mathscr{N}}, \mathbf{h}_{\mathscr{N}} \rangle} \left[\prod_{i=1}^{b} (-1)^{\left\langle \mathbf{e}_{\mathscr{P}}^{(i)}, \mathbf{e}_{\text{aux}}^{(i)} \right\rangle}\right]$$

we have that

$$F\left(\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\text{aux}}^{\intercal}\right) = \sum_{\mathbf{h}_{\mathscr{N}} \in \mathcal{S}_w^{n-s}} \sum_{\left(\mathbf{e}_{\text{aux}}^{(i)} \in \mathcal{S}_{t_{\text{aux}}^{(i)}}^{s^{(i)}}\right)_{i \in [\![1,\, b]\!]}} A\left(\mathbf{h}_{\mathscr{N}},\ \mathbf{e}_{\text{aux}}^{(1)},\ \ldots,\ \mathbf{e}_{\text{aux}}^{(b)}\right) \mathbf{1}_{\mathbf{h}_{\mathscr{N}} \in (\mathcal{C}^{\perp})_{\mathscr{N}}} \prod_{i=1}^{b} \mathbf{1}_{\mathbf{e}_{\text{aux}}^{(i)} + (\mathbf{h}_{\mathscr{N}} \mathbf{R}^{\intercal})^{(i)} \in \mathcal{C}_{\text{aux}}^{(i)}}.$$

Now we use the fact that we can upper bound the variance of $\sum_i A_i \mathbf{X}_i$ where $\mathbf{X}_i$ are some random variables the $A_i \in \{-1, 1\}$ are some fixed coefficient by upper bounding the covariance as

$$\begin{aligned}
Cov\left(A_i\, X_i, A_j\, X_j\right) &= A_i A_j Cov\left(X_i, X_j\right) \\
&\leqslant \left|Cov\left(X_i, X_j\right)\right|.
\end{aligned}$$

This observation allows us to write that

$$\mathbf{Var}\left(F\left(\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\text{aux}}^{\intercal}\right)\right) \leqslant V + C \tag{17}$$

where

$$V \stackrel{\text{def}}{=} \sum_{\mathbf{h}_{\mathscr{N}} \in \mathcal{S}_w^{n-s}} \sum_{\left(\mathbf{e}_{\text{aux}}^{(i)} \in \mathcal{S}_{t_{\text{aux}}^{(i)}}^{s^{(i)}}\right)_{i \in [\![1,\, b]\!]}} \mathbf{Var}\left(\mathbf{1}_{\mathbf{h}_{\mathscr{N}} \in (\mathcal{C}^{\perp})_{\mathscr{N}}} \prod_{i=1}^{b} \mathbf{1}_{\mathbf{e}_{\text{aux}}^{(i)} + (\mathbf{h}_{\mathscr{N}} \mathbf{R}^{\intercal})^{(i)} \in \mathcal{C}_{\text{aux}}^{(i)}}\right) \tag{18}$$

and

$$\begin{aligned}
C \stackrel{\text{def}}{=} \sum_{\mathbf{h}_{\mathscr{N}} \in \mathcal{S}_w^{n-s}} \sum_{\mathbf{g}_{\mathscr{N}} \in \mathcal{S}_w^{n-s}} \sum_{\left(\mathbf{e}_{\text{aux}}^{(i)} \in \mathcal{S}_{t_{\text{aux}}^{(i)}}^{s^{(i)}}\right)_{i \in [\![1,\, b]\!]}} \sum_{\left(\mathbf{z}_{\text{aux}}^{(i)} \in \mathcal{S}_{t_{\text{aux}}^{(i)}}^{s^{(i)}}\right)_{i \in [\![1,\, b]\!]}} \\
\mathbf{1}_{(\mathbf{h}_{\mathscr{N}},\, \mathbf{e}_{\text{aux}}) \neq (\mathbf{g}_{\mathscr{N}},\, \mathbf{z}_{\text{aux}})} \left|\mathbf{C}\left(\mathbf{h}_{\mathscr{N}},\ \mathbf{g}_{\mathscr{N}},\ \mathbf{e}_{\text{aux}},\ \mathbf{z}_{\text{aux}}\right)\right|
\end{aligned} \tag{19}$$

where

$$\mathbf{C}\left(\mathbf{h}_{\mathscr{N}},\ \mathbf{g}_{\mathscr{N}},\ \mathbf{e}_{\text{aux}},\ \mathbf{z}_{\text{aux}}\right) \stackrel{\text{def}}{=} \mathbf{Cov}\left(\mathbf{1}_{\mathbf{h}_{\mathscr{N}}\in(\mathcal{C}^{\perp})_{\mathscr{N}}}\prod_{i=1}^{b}\mathbf{1}_{\mathbf{e}_{\text{aux}}^{(i)}+(\mathbf{h}_{\mathscr{N}}\mathbf{R}^{\intercal})^{(i)}\in\mathcal{C}_{\text{aux}}^{(i)}},\ \mathbf{1}_{\mathbf{g}_{\mathscr{N}}\in(\mathcal{C}^{\perp})_{\mathscr{N}}}\prod_{i=1}^{b}\mathbf{1}_{\mathbf{z}_{\text{aux}}^{(i)}+(\mathbf{h}_{\mathscr{N}}\mathbf{R}^{\intercal})^{(i)}\in\mathcal{C}_{\text{aux}}^{(i)}}\right). \tag{20}$$

Let us first compute the term V. As V is the variance of a Bernoulli distribution, we can upper bound it by the expected value of this Bernoulli:

$$\begin{aligned}\mathbf{Var}\left(\mathbf{1}_{(\mathbf{h}_{\mathscr{N}}\mathbf{R}^{\intercal},\mathbf{h}_{\mathscr{N}})\in\mathscr{W}}\prod_{i=1}^{b}\mathbf{1}_{\mathbf{e}_{\text{aux}}^{(i)}+(\mathbf{h}_{\mathscr{N}}\mathbf{R}^{\intercal})^{(i)}\in\mathcal{C}_{\text{aux}}^{(i)}}\right) &\leqslant \mathbb{E}\left(\mathbf{1}_{(\mathbf{h}_{\mathscr{N}}\mathbf{R}^{\intercal},\mathbf{h}_{\mathscr{N}})\in\mathscr{W}}\prod_{i=1}^{b}\mathbf{1}_{\mathbf{e}_{\text{aux}}^{(i)}+(\mathbf{h}_{\mathscr{N}}\mathbf{R}^{\intercal})^{(i)}\in\mathcal{C}_{\text{aux}}^{(i)}}\right)\\ &= \frac{1}{2^{k-k_{\text{aux}}}} \qquad (Eq.\ (16))\end{aligned}$$

And thus, plugging this last equation in Eq. (18) we get

$$\begin{aligned}V &\leqslant \sum_{\mathbf{h}_{\mathscr{N}}\in\mathcal{S}_{w}^{n-s}}\ \sum_{\left(\mathbf{e}_{\text{aux}}^{(i)}\in\mathcal{S}_{t_{\text{aux}}^{(i)}}^{s^{(i)}}\right)_{i\in[\![1,\ b]\!]}} \frac{1}{2^{k-k_{\text{aux}}}}\\ &= \frac{\binom{n-s}{w}\prod_{i=1}^{b}\binom{s^{(i)}}{t_{\text{aux}}^{(i)}}}{2^{k-k_{\text{aux}}}}\\ &= N.\end{aligned}$$

Let us now compute the covariance terms C by first rewriting $\mathbf{C}\left(\mathbf{h}_{\mathscr{N}},\ \mathbf{g}_{\mathscr{N}},\ \mathbf{e}_{\text{aux}},\ \mathbf{z}_{\text{aux}}\right)$. We have

$$\begin{aligned}&\mathbf{C}\left(\mathbf{h}_{\mathscr{N}},\ \mathbf{g}_{\mathscr{N}},\ \mathbf{e}_{\text{aux}},\ \mathbf{z}_{\text{aux}}\right) = \mathbb{E}\left(\mathbf{1}_{\mathbf{h}_{\mathscr{N}}\in(\mathcal{C}^{\perp})_{\mathscr{N}}}\prod_{i=1}^{b}\mathbf{1}_{\mathbf{e}_{\text{aux}}^{(i)}+(\mathbf{h}_{\mathscr{N}}\mathbf{R}^{\intercal})^{(i)}\in\mathcal{C}_{\text{aux}}^{(i)}}\mathbf{1}_{\mathbf{g}_{\mathscr{N}}\in(\mathcal{C}^{\perp})_{\mathscr{N}}}\prod_{i=1}^{b}\mathbf{1}_{\mathbf{z}_{\text{aux}}^{(i)}+(\mathbf{g}_{\mathscr{N}}\mathbf{R}^{\intercal})^{(i)}\in\mathcal{C}_{\text{aux}}^{(i)}}\right)\\ &-\mathbb{E}\left(\mathbf{1}_{\mathbf{h}_{\mathscr{N}}\in(\mathcal{C}^{\perp})_{\mathscr{N}}},\ \prod_{i=1}^{b}\mathbf{1}_{\mathbf{e}_{\text{aux}}^{(i)}+(\mathbf{g}_{\mathscr{N}}\mathbf{R}^{\intercal})^{(i)}\in\mathcal{C}_{\text{aux}}^{(i)}}\right)\mathbb{E}\left(\mathbf{1}_{\mathbf{g}_{\mathscr{N}}\in(\mathcal{C}^{\perp})_{\mathscr{N}}},\ \prod_{i=1}^{b}\mathbf{1}_{\mathbf{z}_{\text{aux}}^{(i)}+(\mathbf{g}_{\mathscr{N}}\mathbf{R}^{\intercal})^{(i)}\in\mathcal{C}_{\text{aux}}^{(i)}}\right)\end{aligned}$$

Thus

$$\begin{aligned}&\mathbf{C}\left(\mathbf{h}_{\mathscr{N}},\ \mathbf{g}_{\mathscr{N}},\ \mathbf{e}_{\text{aux}},\ \mathbf{z}_{\text{aux}}\right) =\\ &= \mathbb{E}\left(\mathbf{1}_{\mathbf{h}_{\mathscr{N}}\in(\mathcal{C}^{\perp})_{\mathscr{N}}}\mathbf{1}_{\mathbf{g}_{\mathscr{N}}\in(\mathcal{C}^{\perp})_{\mathscr{N}}}\prod_{i=1}^{b}\mathbf{1}_{\mathbf{e}_{\text{aux}}^{(i)}+(\mathbf{h}_{\mathscr{N}}\mathbf{R}^{\intercal})^{(i)}\in\mathcal{C}_{\text{aux}}^{(i)}}\mathbf{1}_{\mathbf{z}_{\text{aux}}^{(i)}+(\mathbf{g}_{\mathscr{N}}\mathbf{R}^{\intercal})^{(i)}\in\mathcal{C}_{\text{aux}}^{(i)}}\right) - \left(\frac{1}{2^{k-k_{\text{aux}}}}\right)^{2}\\ &= \mathbb{P}\left(\mathbf{h}_{\mathscr{N}},\ \mathbf{g}_{\mathscr{N}}\in\left(\mathcal{C}^{\perp}\right)_{\mathscr{N}}\right)\prod_{i=1}^{b}\mathbb{P}\left(\mathbf{e}_{\text{aux}}^{(i)}+(\mathbf{h}_{\mathscr{N}}\mathbf{R}^{\intercal})^{(i)}\in\mathcal{C}_{\text{aux}}^{(i)},\ \mathbf{z}_{\text{aux}}^{(i)}+(\mathbf{g}_{\mathscr{N}}\mathbf{R}^{\intercal})^{(i)}\in\mathcal{C}_{\text{aux}}^{(i)}\right) - \left(\frac{1}{2^{k-k_{\text{aux}}}}\right)^{2}\end{aligned} \tag{21}$$

where in the last line we used the independence of the variables.

1. Case $\mathbf{h}_{\mathscr{N}}\neq\mathbf{g}_{\mathscr{N}}$. Suppose here that $\mathbf{h}_{\mathscr{N}}\neq\mathbf{h}_{\mathscr{N}}$ Let us first compute $\mathbf{C}\left(\mathbf{h}_{\mathscr{N}},\ \mathbf{g}_{\mathscr{N}},\ \mathbf{e}_{\text{aux}},\ \mathbf{z}_{\text{aux}}\right)$. First, as $\left(\mathcal{C}^{\perp}\right)_{\mathscr{N}}\sim\mathcal{U}_{\mathbf{H}}\left(n-s,\ n-k\right)$ we get from Lemma 3.9 that

$$\mathbb{P}\left(\mathbf{h}_{\mathscr{N}},\ \mathbf{g}_{\mathscr{N}}\in\left(\mathcal{C}^{\perp}\right)_{\mathscr{N}}\right) = \left(\frac{1}{2^{k-s}}\right)^{2}$$

Moreover, regardless of the values of $\mathbf{e}_{\text{aux}}^{(i)}$ and $\mathbf{z}_{\text{aux}}^{(i)}$ we have that $\mathbf{e}_{\text{aux}}^{(i)} + (\mathbf{g}_{\mathscr{N}}\mathbf{R}^{\intercal})^{(i)}$ and $\mathbf{z}_{\text{aux}}^{(i)} + (\mathbf{g}_{\mathscr{N}}\mathbf{R}^{\intercal})^{(i)}$ are independent and uniformly distributed which yields that

$$\text{if} \quad \mathbf{h}_{\mathscr{N}} \neq \mathbf{g}_{\mathscr{N}} \quad \text{then} \quad \mathbb{P}\left(\mathbf{e}_{\text{aux}}^{(i)} + (\mathbf{h}_{\mathscr{N}}\mathbf{R}^{\intercal})^{(i)} \in \mathcal{C}_{\text{aux}}^{(i)},\ \mathbf{z}_{\text{aux}}^{(i)} + (\mathbf{g}_{\mathscr{N}}\mathbf{R}^{\intercal})^{(i)} \in \mathcal{C}_{\text{aux}}^{(i)}\right) = \left(\frac{1}{2^{s^{(i)}-k_{\text{aux}}^{(i)}}}\right)^2.$$

Plugging this last equality back into $\mathbf{C}$ gives that

$$\textbf{If} \quad \mathbf{h}_{\mathscr{N}} \neq \mathbf{g}_{\mathscr{N}} \ \textbf{then} \quad \mathbf{C}\left(\mathbf{h}_{\mathscr{N}},\ \mathbf{g}_{\mathscr{N}},\ \mathbf{e}_{\text{aux}},\ \mathbf{z}_{\text{aux}}\right) = 0.$$

Using this fact in Eq. (19) gives that

$$C = \sum_{\mathbf{h}_{\mathscr{N}} \in \mathcal{S}_w^{n-s}} \sum_{\left(\mathbf{e}_{\text{aux}}^{(i)} \in \mathcal{S}_{t_{\text{aux}}^{(i)}}^{s^{(i)}}\right)_{i \in [\![1,\ b]\!]}} \sum_{\left(\mathbf{z}_{\text{aux}}^{(i)} \in \mathcal{S}_{t_{\text{aux}}^{(i)}}^{s^{(i)}}\right)_{i \in [\![1,\ b]\!]}} \mathbf{1}_{\mathbf{e}_{\text{aux}} \neq \mathbf{z}_{\text{aux}}} \left|\mathbf{C}\left(\mathbf{h}_{\mathscr{N}},\ \mathbf{h}_{\mathscr{N}},\ \mathbf{e}_{\text{aux}},\ \mathbf{z}_{\text{aux}}\right)\right| \tag{22}$$

2. Case $\mathbf{h}_{\mathscr{N}} = \mathbf{g}_{\mathscr{N}}$. Let us now compute $\left|\mathbf{C}\left(\mathbf{h}_{\mathscr{N}},\ \mathbf{h}_{\mathscr{N}},\ \mathbf{e}_{\text{aux}},\ \mathbf{z}_{\text{aux}}\right)\right|$. Recall that from Eq. (21) we have that

$$\begin{aligned}
&\mathbf{C}\left(\mathbf{h}_{\mathscr{N}},\ \mathbf{h}_{\mathscr{N}},\ \mathbf{e}_{\text{aux}},\ \mathbf{z}_{\text{aux}}\right) = \\
&= \mathbb{P}\left(\mathbf{h}_{\mathscr{N}} \in \left(\mathcal{C}^{\perp}\right)_{\mathscr{N}}\right) \prod_{i=1}^{b} \mathbb{P}\left(\mathbf{e}_{\text{aux}}^{(i)} + (\mathbf{h}_{\mathscr{N}}\mathbf{R}^{\intercal})^{(i)} \in \mathcal{C}_{\text{aux}}^{(i)},\ \mathbf{z}_{\text{aux}}^{(i)} + (\mathbf{g}_{\mathscr{N}}\mathbf{R}^{\intercal})^{(i)} \in \mathcal{C}_{\text{aux}}^{(i)}\right) - \left(\frac{1}{2^{k-k_{\text{aux}}}}\right)^2.
\end{aligned}$$

But as we have that

$$\begin{aligned}
&\mathbb{P}\left(\mathbf{h}_{\mathscr{N}} \in \left(\mathcal{C}^{\perp}\right)_{\mathscr{N}}\right) \prod_{i=1}^{b} \mathbb{P}\left(\mathbf{e}_{\text{aux}}^{(i)} + (\mathbf{h}_{\mathscr{N}}\mathbf{R}^{\intercal})^{(i)} \in \mathcal{C}_{\text{aux}}^{(i)},\ \mathbf{z}_{\text{aux}}^{(i)} + (\mathbf{h}_{\mathscr{N}}\mathbf{R}^{\intercal})^{(i)} \in \mathcal{C}_{\text{aux}}^{(i)}\right) \\
&\leqslant \frac{1}{2^{k-s}} \prod_{i=1}^{b} \left(\frac{1}{2^{s^{(i)}-t_{\text{aux}}^{(i)}}}\right)^{1+\mathbf{1}_{\mathbf{e}_{\text{aux}}^{(i)} \neq \mathbf{z}_{\text{aux}}^{(i)}}} \\
&= \frac{1}{2^{k-k_{\text{aux}}}} \prod_{i=1}^{b} \left(\frac{1}{2^{s^{(i)}-t_{\text{aux}}^{(i)}}}\right)^{\mathbf{1}_{\mathbf{e}_{\text{aux}}^{(i)} \neq \mathbf{z}_{\text{aux}}^{(i)}}}
\end{aligned}$$

This yield that

$$\left|\mathbf{C}\left(\mathbf{h}_{\mathscr{N}},\ \mathbf{h}_{\mathscr{N}},\ \mathbf{e}_{\text{aux}},\ \mathbf{z}_{\text{aux}}\right)\right| = \mathcal{O}\left(\frac{1}{2^{k-k_{\text{aux}}}} \prod_{i=1}^{b} \left(\frac{1}{2^{s^{(i)}-t_{\text{aux}}^{(i)}}}\right)^{\mathbf{1}_{\mathbf{e}_{\text{aux}}^{(i)} \neq \mathbf{z}_{\text{aux}}^{(i)}}}\right).$$

Finally,

$$\begin{aligned}
C &= \mathcal{O}\left(\sum_{\mathbf{h}_{\mathscr{N}} \in \mathcal{S}_w^{n-s}} \sum_{\left(\mathbf{e}_{\text{aux}}^{(i)} \in \mathcal{S}_{t_{\text{aux}}^{(i)}}^{s^{(i)}}\right)_{i \in [\![1,\ b]\!]}} \sum_{\left(\mathbf{z}_{\text{aux}}^{(i)} \in \mathcal{S}_{t_{\text{aux}}^{(i)}}^{s^{(i)}}\right)_{i \in [\![1,\ b]\!]}} \mathbf{1}_{\mathbf{e}_{\text{aux}} \neq \mathbf{z}_{\text{aux}}} \frac{1}{2^{k-k_{\text{aux}}}} \prod_{i=1}^{b} \left(\frac{1}{2^{s^{(i)}-t_{\text{aux}}^{(i)}}}\right)^{\mathbf{1}_{\mathbf{e}_{\text{aux}}^{(i)} \neq \mathbf{z}_{\text{aux}}^{(i)}}}\right) \\
&= \mathcal{O}\left(\frac{\binom{n-s}{w} \prod_{i=1}^{b} \binom{s^{(i)}}{t_{\text{aux}}^{(i)}}}{2^{k-k_{\text{aux}}}} \sum_{\mathbf{c} \in \{0,1\}^b\,:\,\mathbf{c} \neq \mathbf{0}} \prod_{i=1}^{b} \left(\frac{\binom{s^{(i)}}{t_{\text{aux}}^{(i)}}}{2^{s^{(i)}-t_{\text{aux}}^{(i)}}}\right)^{c_i}\right) \\
&= \mathcal{O}\left(N \sum_{\mathbf{c} \in \{0,1\}^b\,:\,\mathbf{c} \neq \mathbf{0}} \prod_{i=1}^{b} \left(\frac{\binom{s^{(i)}}{t_{\text{aux}}^{(i)}}}{2^{s^{(i)}-t_{\text{aux}}^{(i)}}}\right)^{c_i}\right).
\end{aligned}$$

Plugging this last upper bound for C along with the expression of V in Eq. (18) in Eq. (17) allows writing

$$\mathbf{Var}\left(F_{\mathscr{H},\mathbf{y}}\left(\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\text{aux}}^{\intercal}\right)\right) \leqslant N\left(1+\mathcal{O}\left(\sum_{\mathbf{c}\in\{0,1\}^b\,:\,\mathbf{c}\neq\mathbf{0}}\prod_{i=1}^{b}\left(\frac{\binom{s^{(i)}}{t_{\text{aux}}^{(i)}}}{2^{s^{(i)}-k_{\text{aux}}^{(i)}}}\right)^{c_i}\right)\right)$$

We can now prove our main lemma.

Proof (Proof of Lemma 11.1). The expected values are already given by Lemma 11.4, we only have to show that

$$\mathbf{Var}\left(F\left(\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\text{aux}}^{\intercal}\right)\right) = \mathcal{O}\big(N\, n^{b+1}\ \max\left(1,\ N_{\text{aux}}\right)\big)\,.$$

Recall that from Lemma 11.4, we have that

$$\mathbf{Var}\left(F\left(\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\text{aux}}^{\intercal}\right)\right) \leqslant N\left[1+\sum_{\mathbf{c}\in\{0,1\}^b\,:\,\mathbf{c}\neq\mathbf{0}}\prod_{i=1}^{b}\left(\frac{\binom{s^{(i)}}{t_{\text{aux}}^{(i)}}}{2^{s^{(i)}-t_{\text{aux}}^{(i)}}}\right)^{c_i}\right]. \tag{23}$$

The result will come from the fact that, as for any i, j we have that

$$\begin{aligned} s^{(i)} &= s^{(j)} \pm 1,\\ k_{\text{aux}}^{(i)} &= k_{\text{aux}}^{(j)} \pm 1,\\ t_{\text{aux}}^{(i)} &= t_{\text{aux}}^{(i)} \pm 1, \end{aligned}$$

we can write that

$$\begin{aligned} \frac{\binom{s^{(i)}}{t_{\text{aux}}^{(i)}}}{2^{s^{(i)}-k_{\text{aux}}^{(i)}}} &= \mathcal{O}\left(\frac{s^{(i)}+1}{t_{\text{aux}}^{(i)}+1}\frac{\binom{s^{(j)}}{t_{\text{aux}}^{(j)}}}{2^{s^{(j)}-k_{\text{aux}}^{(j)}}}\right) &(24)\\ &= \mathcal{O}(n)\,\frac{\binom{s^{(j)}}{t_{\text{aux}}^{(j)}}}{2^{s^{(j)}-k_{\text{aux}}^{(j)}}} &(25) \end{aligned}$$

The previous equation yields that, regardless of $\mathbf{c}\in\{0,1\}^b$ in Eq. (23) we have that

$$\begin{aligned} \prod_{i=1}^{b}\left(\frac{\binom{s^{(i)}}{t_{\text{aux}}^{(i)}}}{2^{s^{(i)}-t_{\text{aux}}^{(i)}}}\right)^{c_i} &= \mathcal{O}\left(n^b\ \max\left(1,\ \frac{\prod_{i=1}^{b}\binom{s^{(i)}}{t_{\text{aux}}^{(i)}}}{2^{s-k_{\text{aux}}}}\right)\right)\\ &= \mathcal{O}\big(n^b\ \max\left(1,\ N_{\text{aux}}\right)\big)\,. \end{aligned}$$

Plugging this last equation in Eq. (23) we get:

$$\begin{aligned} \mathbf{Var}\left(F\left(\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\text{aux}}^{\intercal}\right)\right) &\leqslant N\left[1+\sum_{\mathbf{c}\in\{0,1\}^b\,:\,\mathbf{c}\neq\mathbf{0}}\mathcal{O}\big(n^b\ \max\left(1,\ N_{\text{aux}}\right)\big)\right]\\ &\leqslant N\left[1+\ \mathcal{O}\big(n^{b+1}\ \max\left(1,\ N_{\text{aux}}\right)\big)\right]\\ &= \mathcal{O}\big(N\, n^{b+1}\ \max\left(1,\ N_{\text{aux}}\right)\big)\,. \end{aligned}$$

Which is our desired result.

11.2 Proof of the lemma comparing the biases

In this section we prove Lemma 8.3 and which we recall here.

Lemma 8.3. *There exists a positive poly-bounded function f_1 such that for any $s, w, u \in \mathbb{N}$ implicit functions of $n \in \mathbb{N}$ such that* $\mathrm{Root}\left(K_w^{(n-s)}\right) - u \in \Omega\left(n-s\right)$ *we have that*

$$\delta_w^{(n-s)}\left(u-1\right) - \delta_w^{(n-s)}\left(u\right) \geqslant \frac{1}{f_1(n)}\delta_w^{(n-s)}\left(u\right),$$
$$\delta_w^{(n-s)}\left(u\right) - \delta_w^{(n-s)}\left(u+1\right) \geqslant \frac{1}{f_1(n)}\delta_w^{(n-s)}\left(u\right).$$

To prove it we use the following recurrence relations.

Lemma 11.5 (Difference of bias). *Let $n, w, t \in \mathbb{N}$, we have that*

$$\delta_w^{(n)}\left(t-1\right) - \delta_w^{(n)}\left(t\right) = 2\frac{w}{n}\delta_{w-1}^{(n-1)}\left(t-1\right).$$

Proof. Recall that by definition in Lemma 3.13

$$\delta_w^{(n)}\left(t\right) = \frac{K_w^{(n)}\left(t\right)}{\binom{n}{w}}.$$

First let us show the following recurrence relations

$$K_w^{(n)}\left(t-1\right) = K_w^{(n-1)}\left(t-1\right) + K_{w-1}^{(n-1)}\left(t-1\right), \tag{26}$$
$$K_w^{(n)}\left(t\right) = K_w^{(n-1)}\left(t-1\right) - K_{w-1}^{(n-1)}\left(t-1\right). \tag{27}$$

By Lemma 3.12 for any $\mathbf{x} \in \mathcal{S}_v^n$ we have

$$\begin{aligned} K_w^{(n)}\left(v\right) &= K_w^{(n)}\left(\mathbf{x}\right) \\ &= \sum_{\mathbf{h}\in\mathcal{S}_w^n} (-1)^{\langle \mathbf{h},\mathbf{x}\rangle}. \end{aligned}$$

By taking any $\mathbf{x}' \in \mathcal{S}_{t-1}^{n-1}$ and constructing $\mathbf{x} = (0 \,||\, \mathbf{x}') \in \mathcal{S}_{t-1}^n$ we get, by decomposing the previous sum on the values of $\mathbf{h}$ on the first position, that

$$K_w^{(n)}\left(t-1\right) = K_w^{(n-1)}\left(t-1\right) + K_{w-1}^{(n-1)}\left(t-1\right).$$

Thus, Eq. (26) is showed. In the same manner, taking this time $\mathbf{x} = (1 \,||\, \mathbf{x}') \in \mathcal{S}_t^n$ we get

$$K_w^{(n)}\left(t\right) = K_w^{(n-1)}\left(t-1\right) - K_{w-1}^{(n-1)}\left(t-1\right).$$

Thus, Eq. (27) is showed. Now, using the fact that

$$\binom{n}{w} = \frac{n}{w}\binom{n-1}{w-1}$$

and these recurrence relations we get

$$\begin{aligned} \delta_w^{(n)}\left(t-1\right) - \delta_w^{(n)}\left(t-1\right) &= \frac{2\,K_{w-1}^{(n-1)}\left(t-1\right)}{\binom{n}{w}} \\ &= 2\,\frac{w}{n}\frac{K_{w-1}^{(n-1)}\left(t-1\right)}{\binom{n-1}{w-1}}. \end{aligned}$$

The proof of Lemma 8.3 is now straightforward.

Proof (Proof of Lemma 8.3). Using the previous recurrence relations we have that

$$\delta_w^{(n-s)}(u-1) - \delta_w^{(n-s)}(u) = \delta_{w-1}^{(n-s-1)}(u-1).$$

Now, using Proposition 3.15, there exists a bivariate function κ such that

$$\delta_{w-1}^{(n-s-1)}(u-1) = \tilde{\Theta}\left(2^{\kappa((w-1)/(n-s-1),(u-1)/(n-s-1))n}\right).$$

By assumption in the lemma, we have that $u < \text{Root}\left(K_w^{(n-s)}\right)$, this allows to show that the couple $((w-1)/(n-s-1),(u-1)/(n-s-1))$ belongs to $A \stackrel{\text{def}}{=} \{(\omega,\tau) \in [0,\,1]^2 \,:\, \tau < 1/2 - \sqrt{\omega(1-\omega)}\,\}$. Using the fact, from Proposition 3.15 that κ is differentiable on A along with Taylor's theorem allows to conclude that $\delta_{w-1}^{(n-s-1)}(u-1) = \tilde{\Omega}\left(\delta_w^{(n-s)}(u)\right)$.

11.3 Proof of the main theorem

We prove here the main theorem Theorem 5.1.

Proof (Proof of the main Theorem 5.1). In Theorem 5.1 we choose the positive poly-bounded function f appearing as $f(n) = (f_0(n)f_1(n))^2 f_2(n)^3$ where f_0, f_1 and f_2 are defined in Proposition 8.1, Lemma 8.3 and Proposition 8.5 respectively. Let us now consider parameters $k, t, s, w, u, s, k_{\text{aux}}, t_{\text{aux}}, b$ satifying the conditions of the theorem. Now, about the choice for N_{iter}, by using Lemma 8.4 we can show that there exists N_{iter} that is such that i) $N_{\text{iter}} = \widetilde{\mathcal{O}}\left(\frac{\binom{n}{t}}{\binom{s}{t-u}\binom{n-s}{u}}\right)$ and ii) there exists with probability $1 - o(1)$ an iteration of the algorithm that is such that the bet on the error is valid, namely such that

$$|\mathbf{e}_{\mathscr{N}}| = u \bigwedge_{i=1}^{b} \left|\mathbf{e}_{\mathscr{P}}^{(i)}\right| = (t-u)^{(i)}. \tag{28}$$

Let us consider such a value for N_{iter} and suppose we are in one of the iteration where the bet is valid. It is easy to convince oneself that the conditions in the theorem are sufficient to ensure that the conditions to apply Proposition 8.1 are met with probability $1 - o(1)$. We recall that those conditions to apply Proposition 8.1 are that $\dim(\mathcal{C}_{\mathscr{P}}) = s$ and that $\mathscr{H}$ is equal to the full set of decoded dual vectors. We suppose that this iteration is such that those conditions are met. Let $i \in [\![1,\, n-s]\!]$ and let us now prove that

$$\mathbb{P}\left(G\left(\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\text{aux}}^{\intercal}\right)_i = (\mathbf{e}_{\mathscr{N}})_i\right) = 1 - o(1/n). \tag{29}$$

Note that proving it would directly prove our theorem. Indeed we then we can easily prove that

$$\mathbb{P}\left(G\left(\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\text{aux}}^{\intercal}\right) = \mathbf{e}_{\mathscr{N}}\right) = 1 - o(1)$$

by using the union bound. In turn, we can show that $\mathscr{N}$ is an information set of the code $\mathcal{C}$ with probability $1 - o(1)$ by using the condition that $n - s - k \in \omega(1)$ in the theorem. This means that with probability $1 - o(1)$ the procedure TESTING-$\mathbf{e}_{\mathscr{N}}$ returns the error $\mathbf{e}$.

Let us now prove Eq. (29). Suppose, without loss of generality that $(\mathbf{e}_{\mathscr{N}})_i = 1$, the proof in the other case is similar. Recall that by construction $G(\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\text{aux}}^{\intercal})_i = 1$ if and only if $F_{\mathscr{H},\mathbf{y}}(\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\text{aux}}^{\intercal}) < F_{\mathscr{H},\mathbf{y}^{[i]}}(\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\text{aux}}^{\intercal})$. Thus to prove our result we only have to prove that

$$\mathbb{P}\left(F_{\mathscr{H},\mathbf{y}}(\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\text{aux}}^{\intercal}) < F_{\mathscr{H},\mathbf{y}^{[i]}}(\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\text{aux}}^{\intercal})\right) = 1 - o(1).$$

We prove it using Proposition 8.1. We have that with probability $1 - \mathcal{O}\left(\frac{1}{n^2}\right)$

$$F_{\mathbf{y},\mathscr{H}}(\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\text{aux}}^{\intercal}) \in \left[E(0) \pm f_1(n^b)\sqrt{N}\right],$$
$$F_{\mathbf{y}^{[i]},\mathscr{H}}(\mathbf{e}_{\mathscr{P}}\mathbf{G}_{\text{aux}}^{\intercal}) \in \left[E(1) \pm f_1(n^b)\sqrt{N}\right]$$

where

$$E(x) \stackrel{\text{def}}{=} N\delta_w^{(n-s)}(u+x)\prod_{j=1}^{b}\delta_{t_{\text{aux}}^{(j)}}^{(s^{(j)})}\left((t-u)^{(j)}\right), \qquad N \stackrel{\text{def}}{=} \frac{\binom{n-s}{w}\prod_{j=1}^{b}\binom{s^{(j)}}{t_{\text{aux}}^{(j)}}}{2^{k-k_{\text{aux}}}}.$$

This allows saying that we make the good guess for each coordinate with probability $1 - o(1/n)$ as long as

$$E(-1) - E(0) > f_0(n^b)\sqrt{N} \tag{30}$$

Using successively Lemma 8.3 and Proposition 8.5 we get

$$\begin{aligned} E(-1) - E(0) &\geqslant \frac{1}{f_1(n^b)} N\delta_w^{(n-s)}(u)\prod_{j=1}^{b}\delta_{t_{\text{aux}}^{(j)}}^{(s^{(j)})}\left((t-u)^{(j)}\right) \\ &\geqslant \frac{1}{f_1(n)f_2(n)} N\delta_w^{(n-s)}(u)\,\delta_{t_{\text{aux}}}^{(s)}(t-u) \end{aligned}$$

Plugging this into Eq. (30), this means that we make the right decision with probability $1 - o(1)$ if

$$\frac{1}{f_1(n^b)f_2(n^b)} N\delta_w^{(n-s)}(u)\,\delta_{t_{\text{aux}}}^{(s)}(t-u) \geqslant f_0(n^b)\sqrt{N}$$

This is equivalent to asking that

$$N \geqslant \frac{\left(f_0(n^b)f_1(n^b)f_2(n^b)\right)^2}{\left(\delta_w^{(n-s)}(u)\,\delta_{t_{\text{aux}}}^{(s)}(t-u)\right)^2}.$$

But, from Proposition 8.5 we have that

$$N \geqslant \frac{1}{f_2(n^b)}\frac{\binom{s}{t_{\text{aux}}}}{2^{s-k_{\text{aux}}}}.$$

Thus replacing N in the previous equation we get that our condition to make the right decision with probability $1 - o(1)$ is

$$\frac{\binom{n-s}{w}\binom{s}{t_{\text{aux}}}}{2^{s-k_{\text{aux}}}} \geqslant \frac{\left(f_0(n^b)f_1(n^b)\right)^2 f_2(n^b)^3}{\left(\delta_w^{(n-s)}(u)\,\delta_{t_{\text{aux}}}^{(s)}(t-u)\right)^2}.$$

which is exactly the condition of the theorem.