Dual Support Decomposition in the Head: Shorter Signatures from Rank SD and MinRank

Loïc Bidoux[1], Thibauld Feneuil[2], Philippe Gaborit[3], Romaric Neveu[3], and Matthieu Rivain[2]

[1] Technology Innovation Institute, Abu Dhabi, UAE
[2] CryptoExperts, Paris, France
[3] University of Limoges, Limoges, France

Abstract. The MPC-in-the-Head (MPCitH) paradigm is widely used for building post-quantum signature schemes, as it provides a versatile way to design proofs of knowledge based on hard problems. Over the years, the MPCitH landscape has changed significantly, with the most recent improvements coming from *VOLE-in-the-Head* (VOLEitH) and *Threshold-Computation-in-the-Head* (TCitH).

While a straightforward application of these frameworks already improve the existing MPCitH-based signatures, we show in this work that we can adapt the arithmetic constraints representing the underlying security assumptions (here called the *modeling*) to achieve smaller sizes using these new techniques. More precisely, we explore existing modelings for the rank syndrome decoding (RSD) and MinRank problems and we introduce a new modeling, named *dual support decomposition*, which achieves better sizes with the VOLEitH and TCitH frameworks by minimizing the size of the witnesses. While this modeling is naturally more efficient than the other ones for a large set of parameters, we show that it is possible to go even further and explore new areas of parameters. With these new modeling and parameters, we obtain low-size witnesses which drastically reduces the size of the "arithmetic part" of the signature.

We apply the TCitH and VOLEitH frameworks to our new modeling for both RSD and MinRank and compare our results to the NIST candidates RYDE, MiRitH, and MIRA (MPCitH-based schemes from RSD and MinRank). We also note that recent techniques optimizing the sizes of GGM trees are applicable to our schemes and further reduce the signature sizes by a few hundred bytes. We obtain signature sizes below 3.5 kB for 128 bits of security with $N = 256$ parties (a.k.a. leaves in the GGM trees) and going as low as ≈ 2.8 kB with $N = 2048$, for both RSD and MinRank. This represents an improvement of more than 2 kB compared to the original submissions to the 2023 NIST call for additional signatures.

1 Introduction

The MPC-in-the-Head (MPCitH) paradigm is a popular framework to build post-quantum signatures. After sharing the secret key, the signer emulates "in his head" an MPC protocol and commits each party's view independently. He then reveals the views of a pseudo-random subset of parties, where this subset is given by the hash digest of the commitments (in the setting of the Fiat-Shamir heuristic). By the privacy of the MPC protocol, nothing is revealed about the secret key, which implies the zero-knowledge property. On the other hand, a malicious signer needs to cheat for at least one party, which shall be discovered by the verifier with high probability, hence ensuring the unforgeability property.

In the new NIST call for additional post-quantum signatures [33], many submissions rely on the MPCitH paradigm applied on a large range of security assumptions. Three MPCitH candidates fall in the rank-based cryptography category:

- RYDE [4], for which the security relies on the hardness of solving the rank syndrome decoding problem;
- MIRA [5] and MiRitH [1], for which the security relies on the hardness of solving the MinRank problem (MIRA and MiRitH rely on the same security assumption, but use different modelings and MPC protocols).

Recently, new techniques of MPC-in-the-Head have been proposed:

- the VOLE-in-the-Head (VOLEitH) framework [12] released in Summer 2023;[4]
- the TC-in-the-Head (TCitH) framework [21] released in Autumn 2023.[5]

As shown in [21] a simple application of these frameworks leads to shorter and faster signature schemes compared to those submitted to the NIST call (for similar underlying security assumption).

For MPCitH-based schemes (including those based on VOLEitH and TCitH), the signatures are composed of two parts, a "symmetric part" made of seeds and hash digests and an "arithmetic part" composed of the open party views and broadcast shares of the MPC protocol. While for a given security level the symmetric part is of rather fixed size (for the considered MPCitH framework), the arithmetic part depends on the modeling of the used security assumption and the associated MPC protocol. In the traditional broadcast-based MPCitH framework (*i.e.* the MPCitH framework widely used before VOLEitH and TCitH), to minimize the signature size, the designers had minimize the *sum of the sizes of the MPC input and of the broadcasted values* while considering only *linear* multiparty computation. With the VOLEitH and TCitH frameworks, the game rules have changed. These frameworks enable quadratic (or higher degree) multiparty computation, which implies that minimizing the signature size is achieved only by minimizing the MPC protocol input (i.e., the witness of the modeling).

In rank-based cryptography, several modelings for the rank syndrome decoding problem and the MinRank problem have been proposed. The first one

[4] While VOLEitH has not been introduced as an MPCitH technique, [21] showed that it can be considered as such.

[5] The original version of the TCitH framework was released in Autumn 2022 [22] (and published at Asiacrypt 2023), we refer here to the improved version of the TCitH framework [21].

is derived from [37] and consists in working with a permuted version and an additively-masked version of the secret. The best scheme relying on it is proposed in [15]. The second modeling is based on q-polynomials and is first used in such a context in [19]. The third modeling consists in writing the low-rank object as the product of two small matrices and is first used in such a context in [2] and [19]. The last modeling relies on Kipnis-Shamir technique, initially proposed for the cryptanalysis of the MinRank problem [29] and used to build a scheme in [1]. We sum up the different techniques to handle the rank metric in Table 1.

Problem	**Permuted Secret**	**q-Polynomial Evaluation (q-pol)**	**Matrix Rank Decomposition (MRD)**	**Kipnis Shamir (KS)**	**Dual Support Decomposition (DSD)**
RSD	BG23 [15]	RYDE [4, 19]	Fen24 [19]	-	This work
MinRank	-	MIRA [5, 19]	Fen24 [19]	MiRitH [1]	This work

Table 1: Techniques used in MPCitH-based signatures for RSD and MinRank.

In this work, we explore modelings for the rank syndrome decoding problem and the MinRank problems to identify the best option with the new VOLEitH and TCitH techniques. We show that the shortest signatures with RSD and MinRank are obtained thanks to the *dual support decomposition* modeling, which consists in finding a basis $(e_1, \dots, e_r)$ and coefficients $c_{1,1}, \dots, c_{n,r}$ such that

$$y = Hx \quad \text{and} \quad \forall i,\ x_i = \sum_{j=1}^{r} c_{i,r} \cdot e_j.$$

While this modeling is quite natural for the rank syndrome decoding problem, it requires to work with a dual matrix of a code in the MinRank problem: we need to consider the syndrome decoding problem for matrix codes. In fact, the MinRank problem being the message decoding problem for such codes, we define in this work the MinRank Syndrome problem, which is a equivalent variant of the MinRank problem which has not been previously used in cryptosystems. Working in the dual has the advantage to remove the encoded message from the witness of the code-based problem, leading to a shorter witness. With the dual support decomposition modeling, the witness size (and thus the signature size) is independent of the code dimension, thanks to the definition of the syndrome version of the MinRank problem. This enables us to optimize the parameters by taking codes of larger dimensions.

We then apply the TCitH and VOLEitH frameworks on the optimal modeling, yielding new signature schemes with smaller sizes as summarized in Table 2.

We also put the signature sizes of the NIST candidates based on the same security assumptions (namely RYDE, MIRA and MiRitH) in the column "MPCitH" and their signature sizes when performing a straightforward application of VOLEitH and TCitH. We observe that the difference in signature sizes between VOLEitH and TCitH tends to disappear while increasing the parameter N, i.e., the number of leaves in GGM seed trees used for the commitment (a.k.a. the number of parties in standard MPCitH schemes). Since these two frameworks are faster than previous MPCitH schemes, it becomes natural to consider larger values of N. We obtain signature sizes down to 3.7 kB for TCitH with $N = 256$ leaves, and down to 2.9 kB for VOLEitH and TCitH with $N = 2048$ leaves (more details are given in Tables 7 and 10). The ranges of sizes reported in Table 2 correspond to a parameter N ranging between 256 and 2048. Let us note that new generic optimizations for MPCitH-based signatures have been proposed in [11] very recently. We applied these optimisations to our new signature schemes, enabling us to save an additional few hundred bytes. The obtained sizes are reported in Table 2 with the label "optimized".

Security Assumption	Scheme	MPCitH	VOLEitH	TCitH
Rank SD	RYDE (q-pol)	5 956 B	4 133–4 720 B	4 274–5 281 B
	Our scheme (DSD), optimized	-	2 851–3 450 B	2 937–3 708 B
MinRank	MIRA (q-pol)	5 640 B	4 170–4 770 B	4 314–5 340 B
	MiRitH-Ia (KS)	5 665 B	3 762–4 226 B	3 873–4 694 B
	MiRitH-Ib (KS)	6 298 B	4 110–4 690 B	4 250–5 245 B
	Our scheme (DSD), optimized	-	2 813–3 396 B	2 896–3 640 B

Table 2: Comparison of our schemes based on dual support decomposition (DSD) with the NIST candidates based on the same security assumptions. The sizes in the column "MPCitH" are given when using seed trees with 256 leaves, while the size range in columns "VOLEitH" and "TCitH" are given when using seed trees with between 256 and 2048 leaves.

Paper organization. The paper is organized as follows: In Section 2, we introduce the necessary background on the rank metric and sharing schemes. We present the existing attacks against RSD and MinRank in Section 3. We explore the possible modelings for rank-based cryptography in Section 4. We recall the TCitH and VOLEitH frameworks in Section 5 and we apply these frameworks to the dual support decomposition modeling to obtain new signature schemes in Section 6.

2 Preliminaries

2.1 Notations

We denote by $\mathbb{F}_q$ the finite field of size q. The set of vectors with n coordinates in $\mathbb{F}_q$ is referred as $\mathbb{F}_q^n$, the set of matrices with m rows and n columns in $\mathbb{F}_q$ is referred as $\mathbb{F}_q^{m\times n}$. We use lowercase bold letters to represent vectors and uppercase bold letters for matrices ($\boldsymbol{E} \in \mathbb{F}_q^{m\times n}$, $\boldsymbol{x} \in \mathbb{F}_q^k$, $x \in \mathbb{F}_q$). The subset of integers from 1 to n is represented with $[1, n]$. If S is a set, we write $x \overset{\$}{\longleftarrow} S$ the uniform sampling of a random element x in S. We note the $\mathbb{F}_q$-linear subspace of $\mathbb{F}_{q^m}$ generated by $(x_1, \dots, x_n) \in \mathbb{F}_{q^m}^n$ as $\langle x_1, \dots, x_n\rangle$. Let us define the gaussian coefficient $\begin{bmatrix} m \\ r \end{bmatrix}_q = \prod_{i=0}^{r-1} \frac{q^m - q^i}{q^r - q^i} \approx q^{r(m-r)}$, it corresponds to the number of different dimension-r $\mathbb{F}_q$-linear subspaces of $\mathbb{F}_{q^m}$.

2.2 Secret Sharing

A threshold secret sharing scheme is a method to share a value v into a sharing $[\![v]\!] := ([\![v]\!]_1, \dots, [\![v]\!]_N)$ such that v can be reconstructed from any $\ell{+}1$ shares while no information is revealed on the secret from the knowledge of ℓ shares. We note by $[\![x]\!]_i$ the i^{th} share of $[\![x]\!]$ (*i.e.* the share of the i^{th} party). We can also note $[\![x]\!]_I$ where I is a set of indices, to denote all the shares of the parties in the set I.

Let us define Shamir's secret sharing scheme [36], since the frameworks we will consider rely on it. Let ℓ and N two integers such that $1 \le \ell \le N$. Let $e, \omega_1, \dots, \omega_N$ be $N+1$ distinct elements of $\mathbb{F} \cup \{\infty\}$. To share a value $v \in \mathbb{F}$ using Shamir's secret sharing scheme, one should

1. sample ℓ randoms values $r_1, \dots, r_\ell$ of $\mathbb{F}$;
2. compute the polynomial P by interpolation such that

$$P(e) = v \quad \text{and} \quad \forall i \in [1, \ell], P(\omega_i) = r_i;$$

3. build the N shares $[\![v]\!]_1, \dots, [\![v]\!]_N$ as

$$\forall i \in [1, N],\ [\![v]\!]_i := P(\omega_i).$$

To recover the secret value from $\ell + 1$ shares, we re-compute the polynomial P by interpolation and we just deduce $P(e)$. Let us stress that $P(\infty)$ refers to the leading coefficient of the polynomial P. The most classical choice is to set e to zero but we may consider alternative choices depending on the context (and in particular $e = \infty$).

We define the *degree* of a Shamir's secret sharing as the degree of the underlying polynomial. A sharing generated using the above process is of degree ℓ. The sum of a d_1-degree sharing and a d_2-degree sharing is of degree $\max(d_1, d_2)$, while the multiplication is of degree $d_1 + d_2$.

2.3 Rank Metric and Hard Problems for Cryptography

We will first recall some background on the Rank Metric, and we will then define hard problems we will use (RSD and $\mathsf{MinRank}$).

Definition 1 (Rank Metric over $\mathbb{F}_{q^m}^n$). *Let $\boldsymbol{x} = (x_1, \dots, x_n) \in \mathbb{F}_{q^m}^n$, and $\mathcal{B} = (b_1, \dots, b_m) \in \mathbb{F}_{q^m}^m$ an $\mathbb{F}_q$-basis of $\mathbb{F}_{q^m}$. Each coordinate x_j can be associated with a vector $(x_{j,1}, \dots, x_{j,m}) \in \mathbb{F}_q^m$ such that $x_j = \sum_{i=1}^m x_{j,i} b_i$. Let us define the following notations:*

- $\boldsymbol{M_x} = (x_{j,i})_{(j,i)\in[1,n]\times[1,m]}$ *is the* matrix associated to the vector $\boldsymbol{x}$*;*
- *the* rank weight *is defined as:* $w_R\big(\boldsymbol{x}\big) = \mathsf{rank}(\boldsymbol{M_x})$*;*
- *the* distance *between two vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ in $\mathbb{F}_{q^m}^n$ is: $d(x, y) = w_R\big(\boldsymbol{x} - \boldsymbol{y}\big)$;*
- *the* support *of a vector* $\mathrm{Supp}(\boldsymbol{x})$ *is the $\mathbb{F}_q$-linear subspace of $\mathbb{F}_{q^m}$ generated by its coordinates:* $\mathrm{Supp}(\boldsymbol{x}) = \langle x_1, \dots, x_n \rangle$*.*

Definition 2. *A linear code $\mathcal{C}$ over $\mathbb{F}_{q^m}$ of dimension k and length n is a linear subspace of $\mathbb{F}_{q^m}^n$ of dimension k. The elements of $\mathcal{C}$ are called codewords. The code $\mathcal{C}$ can be represented in two ways:*

- *by a generator matrix $\boldsymbol{G}$, where $\mathcal{C} = \{\boldsymbol{mG}, \boldsymbol{m} \in \mathbb{F}_{q^m}^k\}$, or*
- *by a parity-check matrix $\boldsymbol{H} \in \mathbb{F}_{q^m}^{(n-k)\times n}$ where $\mathcal{C} = \{\boldsymbol{x} \in \mathbb{F}_{q^m}^n : \boldsymbol{H}\boldsymbol{x}^\top = \boldsymbol{0}^\top\}$*

We now continue by formally recalling the definition of the rank syndrome decoding (RSD) problem.

Definition 3 (RSD problem). *Let q, m, n, k and r be positive integers. Let $\boldsymbol{H} \xleftarrow{\$} \mathbb{F}_{q^m}^{(n-k)\times n}$ and $\boldsymbol{x} \xleftarrow{\$} \mathbb{F}_{q^m}^n$ such that $w_R\big(\boldsymbol{x}\big) = r$. Let $\boldsymbol{y}^\top = \boldsymbol{H}\boldsymbol{x}^\top$. Given $(\boldsymbol{H}, \boldsymbol{y})$, the computational* $\mathsf{RSD}(q, m, n, k, r)$ *problem asks to find a vector $\tilde{\boldsymbol{x}} \in \mathbb{F}_{q^m}^n$ such that $\boldsymbol{H}\tilde{\boldsymbol{x}}^\top = \boldsymbol{y}^\top$ and $w_R\big(\tilde{\boldsymbol{x}}\big) = r$.*

We now introduce a variant of the above problem, the RSD_s problem and later argue that it is as hard as the standard RSD problem.

Definition 4 (RSD_s problem). *Let q, m, n, k and r be positive integers. Let $\boldsymbol{H} \xleftarrow{\$} \mathbb{F}_{q^m}^{(n-k)\times n}$ and $\boldsymbol{x} = (x_i) \xleftarrow{\$} \mathbb{F}_{q^m}^n$ such that $w_R\big(\boldsymbol{x}\big) = r$, $x_1 = 1 \in \mathbb{F}_{q^m}$ and $\langle x_1, \dots, x_r \rangle_{\mathbb{F}_q} = \mathrm{Supp}(\boldsymbol{x})$. Let $\boldsymbol{y}^\top = \boldsymbol{H}\boldsymbol{x}^\top$. Given $(\boldsymbol{H}, \boldsymbol{y})$, the computational* $\mathsf{RSD}_s(q, m, n, k, r)$ *problem asks to find a vector $\tilde{\boldsymbol{x}} \in \mathbb{F}_{q^m}^n$ such that $\boldsymbol{H}\tilde{\boldsymbol{x}}^\top = \boldsymbol{y}^\top$ and $w_R\big(\tilde{\boldsymbol{x}}\big) = r$.*

The last problem we will rely on is the well-known $\mathsf{MinRank}$ problem:

Definition 5 ($\mathsf{MinRank}$ problem). *Let q, m, n, k and r be positive integers. Let $\boldsymbol{M}_1, \dots, \boldsymbol{M}_k, \boldsymbol{E} \in \mathbb{F}_q^{m\times n}$ and $\boldsymbol{x} := (x_1, \dots, x_k) \in \mathbb{F}_q^k$ be uniformly sampled such that*

$$\mathsf{rank}(\boldsymbol{E}) \leq r \quad \textit{with} \quad \boldsymbol{M} := \boldsymbol{E} - \sum_{i=1}^{k} x_i \boldsymbol{M}_i.$$

Given $\boldsymbol{M}, \boldsymbol{M}_1 \dots, \boldsymbol{M}_k$, the computational $\mathsf{MinRank}(q, m, n, k, r)$ *problem asks to retrieve the vector $\boldsymbol{x}$.*

The last notion to recall is the Gilbert-Varshamov bound for the rank metric and for MinRank. This bound in rank metric has been introduced in [30]. It can be seen as the probable minimum weight of a random code.

Definition 6 (Rank Gilbert-Varshamov Bound). *Let S_r be the number of elements of the sphere in $\mathbb{F}_{q^m}^n$ of radius r centered in 0, i.e, the number of elements in $\mathbb{F}_{q^m}^n$ of weight exactly r. We have $S_0 = 1$, and for $r \geq 1$,*

$$S_r = \prod_{j=0}^{r-1} \frac{(q^n - q^j)(q^m - q^j)}{q^r - q^j}.$$

Let $B_r := \sum_{i=0}^{r} S_r$ be the number of elements of the ball in $\mathbb{F}_{q^m}^n$ of radius r centered in 0. The Rank Gilbert-Varshamov (RGV) bound for an $[n, k]$ linear code over $\mathbb{F}_{q^m}$ is the smallest integer r such that

$$q^{m(n-k)} \leq B_r$$

Using the approximation $B_r \approx q^{(m+n-r)r}$, one can say the RGV bound is the smallest r such that $m(n-k) \leq (m-r)r + nr$. We call this value d_{RGV}. The same bound exists for matrix codes (i.e, for MinRank) as they are simply $\mathbb{F}_q$-linear codes. Courtois described this bound in [16, Section 24.2], and it can also be derived from the one above easily (consider a $[m \times n, k]$ linear code over $\mathbb{F}_q$ instead of $[n, k]$ linear over $\mathbb{F}_{q^m}$). This bound is also mentioned in attacks on MinRank ([9], [8] for instance). Concretely, this states that, for an instance of MinRank with parameters (q, m, n, k, r), we do not expect to obtain more than one solution if r is chosen such that $k + 1 \leq (m-r)(n-r)$.

Complexity of attacks for parameters on the GV bound. For RSD, the parameter r is taken as $d_{\mathsf{RGV}} - 1$, i.e, the highest r such that $(m-r)r + nr < m(n-k)$. With this parameter, if $\boldsymbol{H}$ and $\boldsymbol{y}$ were to be randomly sampled, one would expect to have a solution with probability $q^{(m+n-r)r-m(n-k)}$. Since $\boldsymbol{y}$ is set so there is a solution and since we are below RGV, it is not expected to have another solution. For MinRank, we take parameters on the RGV bound, with $k+1 = (m-r)(n-r)$. For $k+1$ matrices randomly sampled $(\boldsymbol{M}, \boldsymbol{M}_1, \ldots, \boldsymbol{M}_k)$, the probability to have a solution to the MinRank instance is $q^{(m+n-r)r-(mn-k)}$. Since $\boldsymbol{M}$ is set so that there is a solution and since we are on GV, it is not expected to have another solution for the instance. Let us now explain why in addition to having only one solution, it is important to take parameters according to these bounds. Since the combinatorial attacks from [34] for RSD and [26] for MinRank, very few improvements have been made in the complexity. For MinRank, the kernel attack is still the best combinatorial attack, and for RSD, the exponential part of the complexities is still quadratic and has known almost no improvement for over 20 years (with the exception of [6], which slightly improved the complexity). Regarding the algebraic attacks, introduced in [7] and improved in [10] and [8], they managed to greatly reduce the complexity for the RQC and LRPC schemes. However, this came from the fact that these parameters were not on RGV. The attacked parameters were in $\mathcal{O}\left(\sqrt{n-k}\right)$, which made them easier to attack, whereas we will consider parameters around the RGV bound, in $\mathcal{O}(n)$. In practice, for parameters taken at the RGV bound, or just below, the algebraic attacks have roughly the same complexity as the combinatorial ones ([8]). Overall, this means that, for parameters taken on the Rank Gilbert-Varshamov bound, the attacks have known no significant amelioration for the last 20 years.

3 Security and Parameters for RSD_s and MinRank

We give here the well known reduction from RSD to RSD_s, and then the attacks considered against RSD and MinRank, which we will use in order to establish parameters for the signature schemes. We will also use these attacks in order to establish parameters to compare the different modelings in Section 4.

3.1 Security of the Rank Syndrome Decoding Problem

We deal here with the RSD problem, first by explaining the relation between RSD and RSD_s, and then the attacks on RSD.

Security Reduction The RSD_s problem was most notably used in the RQC scheme in order to optimize it [31]. In the following, we show that the RSD_s problem is as hard as the standard RSD problem. More precisely, we show that any RSD instance can be solved by an RSD_s solver. This is the same reduction as in [34], [6], [7], and others, used to specialize some variables. We exhibit below the reduction which has not formally been described in previous works (as part of the folklore of rank-based cryptography).

Proposition 1. *Let q, m, n, k, r be positive integers such that $n > k$. Let $\mathcal{A}_s$ be an algorithm which solves a $(q, m, n, k+1, r)$-instance of the RSD_s problem in time t with success probability ε_s. Then there exists an algorithm $\mathcal{A}$ which solves a (q, m, n, k, r)-instance of the RSD problem in time t with probability ε, where*

$$\varepsilon \geq \left(\prod_{i=0}^{r-1} \frac{q^n - q^{n-r+i}}{q^n - q^i}\right) \cdot \varepsilon_s$$

under the assumption that the code $\mathcal{C}$ associated to the parity-check matrix $\boldsymbol{H}$ of the RSD instance contains no words of weight r.

Proof. To prove the theorem, we build below an algorithm $\mathcal{A}$ to solve the RSD problem of parameters (q, m, n, k, r) using an algorithm $\mathcal{A}_\mathrm{s}$ which solves the RSD_s problem with parameters $(q, m, n, k+1, r)$, assuming that the code that corresponds to the input instance does not contain words of weight r.

```
Algorithm $\mathcal{A}$ (on input an $\mathsf{RSD}$ instance $(\boldsymbol{H}, \boldsymbol{y})$):

1. Sample an invertible matrix $\boldsymbol{U} \in \mathbb{F}_q^{n \times n}$.
2. Compute $\hat{\boldsymbol{H}}^\top \in \mathbb{F}_{q^m}^{(n-k) \times n}$ as $\boldsymbol{U}\boldsymbol{H}^\top$.
3. Find $\boldsymbol{z}$ such that $\boldsymbol{y} = \boldsymbol{z}\hat{\boldsymbol{H}}^\top$.
4. Build $\hat{\boldsymbol{H}}' \in \mathbb{F}_{q^m}^{(n-k-1) \times n}$ as the parity check matrix of $\mathcal{C} + \langle \boldsymbol{z} \rangle$,
      where $\mathcal{C}$ is the linear code which has $\hat{\boldsymbol{H}}$ as parity-check matrix.
5. Run $\mathcal{A}_{\mathrm{s}}$ on input $(\hat{\boldsymbol{H}}', \boldsymbol{0})$ to get $\hat{\boldsymbol{x}}$.
6. If $\hat{\boldsymbol{x}} = \perp$, return $\perp$.
7. Compute $\alpha \in \mathbb{F}_{q^m}$ such that $\hat{\boldsymbol{x}}\hat{\boldsymbol{H}}^\top = \alpha \cdot \boldsymbol{y}$.
8. Compute $\hat{\hat{\boldsymbol{x}}}$ as $\alpha^{-1} \cdot \hat{\boldsymbol{x}} \cdot \boldsymbol{U}$.
9. Return $\hat{\hat{\boldsymbol{x}}}$.
```

By definition, we know that the RSD instance $(\boldsymbol{H}, \boldsymbol{y})$ has a solution, meaning that there exists a vector $\boldsymbol{x}$ such that $\boldsymbol{y} = \boldsymbol{x}\boldsymbol{H}^\top$ and $w_R\big(\boldsymbol{x}\big) = r$. First, we define $\boldsymbol{x}'$ as $\boldsymbol{x}\boldsymbol{U}^{-1}$. The probability that $\boldsymbol{x}'$ has its r first coordinates which are full rank (under the randomness of $\boldsymbol{U}$) is

$$\varepsilon_1 := \frac{\prod_{i=0}^{r-1}\left(q^n - q^{n-r+i}\right)\prod_{j=r}^{n}(q^n - q^j)}{\#\{\text{invertible matrices of } \mathbb{F}_q^{n \times n}\}} = \prod_{i=0}^{r-1} \frac{q^n - q^{n-r+i}}{q^n - q^i}.$$

We now detail how we obtain this probability. Let $\mathsf{Ker}(\boldsymbol{x})$ be the right kernel of x, *i.e.* $\mathsf{Ker}(\boldsymbol{x}) := \{\boldsymbol{v} \in \mathbb{F}_q^n : \boldsymbol{x}\boldsymbol{v}^\top = 0\}$. It is a $\mathbb{F}_q$-linear subspace of dimension $n - r$ of $\mathbb{F}_q^n$. To obtain $\boldsymbol{x}'$ where the first r coordinates are of rank r, $\boldsymbol{U}^{-1}$ must be as follows (we write the i-th column of $\boldsymbol{U}^{-1}$ as $\boldsymbol{u}_i$):

- $\boldsymbol{u}_1 \notin \mathsf{Ker}(\boldsymbol{x})$;
- $\boldsymbol{u}_2 \notin (\mathsf{Ker}(\boldsymbol{x}) + \langle \boldsymbol{u}_1 \rangle)$;
- More generally, $\boldsymbol{u}_i \notin (\mathsf{Ker}(\boldsymbol{x}) + \langle \boldsymbol{u}_1, \ldots, \boldsymbol{u}_{i-1} \rangle)$.

Let us count the number of successful $\boldsymbol{U}^{-1}$: there are $q^n - q^{n-r}$ choices for $\boldsymbol{u}_1$, $q^n - q^{n-r+1}$ choices for $\boldsymbol{u}_2$, and more generally $q^n - q^{n-r+i}$ choices for $\boldsymbol{u}_i$. In total, there are $\prod_{i=0}^{r-1}\left(q^n - q^{n-r+i}\right)$ choices for the first r columns of $\boldsymbol{U}^{-1}$. The $n - r$ last ones need to be such that $\boldsymbol{U}^{-1}$ is of full rank. Because each additional column should not be included in the subspace spanned by the previous ones, there are $\prod_{j=r}^{n}(q^n - q^j)$ choices for them. By combining the two products, we obtain the probability ε_1.

We assume that the event in which the r first coordinates of $\boldsymbol{x}'$ are full rank occurs. Let us define $\boldsymbol{c} := \boldsymbol{x}' - \boldsymbol{z}$. We have that

$$\boldsymbol{c}\hat{\boldsymbol{H}}^\top = (\boldsymbol{x}' - \boldsymbol{z})\hat{\boldsymbol{H}}^\top = \boldsymbol{x}\boldsymbol{U}^{-1}\boldsymbol{U}\boldsymbol{H}^\top - \boldsymbol{z}\hat{\boldsymbol{H}}^\top = \boldsymbol{x}\boldsymbol{H}^\top - \boldsymbol{y} = \boldsymbol{0},$$

so $\boldsymbol{c}$ is a codeword of $\mathcal{C}$. By defining $\boldsymbol{x}'' := (x_1')^{-1} \cdot \boldsymbol{x}'$ (x_1' is not zero because the r first coordinates of $\boldsymbol{x}'$ are full rank by assumption), we have that $\boldsymbol{x}'' = (x_1')^{-1} \cdot \boldsymbol{c} + (x_1')^{-1} \cdot \boldsymbol{z}$ is a codeword of $\mathcal{C} + \langle \boldsymbol{z} \rangle$. Therefore $\boldsymbol{x}''\hat{\boldsymbol{H}}'$ is equal to $\boldsymbol{0}$. Moreover, the first coordinate of $\boldsymbol{x}''$ is equal to $(x_1')^{-1} \cdot x_1' = 1$ and the r first coordinates of $\boldsymbol{x''}$ are full rank (because those of $\boldsymbol{x'}$ are full rank). We thus have that $(\boldsymbol{\hat{H}'}, \mathbf{0})$ is a $\mathsf{RSD_s}$ instance with probability ε_1.

Let us consider that $\mathcal{A}_\mathrm{s}$ outputs $\boldsymbol{\hat{x}}$ such that $\boldsymbol{\hat{x}} \neq \perp$. We have $\boldsymbol{\hat{x}}\boldsymbol{\hat{H}'}^T = \mathbf{0}$ and $w_R(\boldsymbol{\hat{x}}) = r$. Since $\boldsymbol{\hat{x}}$ belongs to $\mathcal{C} + \langle \boldsymbol{z} \rangle$ (because $\boldsymbol{\hat{x}}\boldsymbol{\hat{H}'}^T = \mathbf{0}$), $\boldsymbol{\hat{x}}$ can be written as

$$\boldsymbol{\hat{x}} := \gamma_1 \cdot \boldsymbol{c_1} + \ldots + \gamma_k \cdot \boldsymbol{c_k} + \alpha \cdot \boldsymbol{z}$$

for some $\gamma_1, \ldots, \gamma_k, \alpha \in \mathbb{F}_{q^m}$, where $(\boldsymbol{c_1}, \ldots, \boldsymbol{c_k})$ is a basis of $\mathcal{C}$. In that case, we have that

$$\begin{aligned} \boldsymbol{\hat{x}}\boldsymbol{\hat{H}}^T &= \gamma_1 \cdot \boldsymbol{c_1}\boldsymbol{\hat{H}}^T + \ldots + \gamma_k \cdot \boldsymbol{c_k}\boldsymbol{\hat{H}}^T + \alpha \cdot \boldsymbol{z}\boldsymbol{\hat{H}}^T \\ &= \mathbf{0} + \ldots + \mathbf{0} + \alpha \cdot \boldsymbol{y} \end{aligned}$$

If $\alpha = 0$, then there would be a codeword of weight r in the code $\mathcal{C}$. Since we assume this is not the case, we get that $\alpha \neq 0$ and so $\boldsymbol{\hat{\hat{x}}}$ is well-defined in Step 8. We thus obtain that

$$\boldsymbol{\hat{\hat{x}}}\boldsymbol{H}^\top = \alpha^{-1} \cdot \boldsymbol{\hat{x}}\boldsymbol{U}\boldsymbol{H}^\top = \alpha^{-1} \cdot \boldsymbol{\hat{x}}\boldsymbol{\hat{H}}^T = \boldsymbol{y}.$$

Moreover, since multiplying by an invertible matrix over $\mathbb{F}_q$ does not change the support, we have $\mathrm{Supp}\left(\boldsymbol{\hat{\hat{x}}}\right) = \alpha^{-1} \cdot \mathrm{Supp}(\boldsymbol{\hat{x}})$, implying that $w_R\left(\boldsymbol{\hat{\hat{x}}}\right) = w_R(\boldsymbol{\hat{x}}) = r$. The algorithm $\mathcal{A}$ outputs a valid RSD solution or $\perp$, and the probability that $\mathcal{A}$ does not output $\perp$ is lower bounded by

$$\begin{aligned} \varepsilon &:= \Pr[\mathcal{A}(\boldsymbol{H}, \boldsymbol{y}) \neq \perp] = \Pr[\mathcal{A}_\mathrm{s}(\boldsymbol{\hat{H}'}, \mathbf{0}) \neq \perp] \\ &\geq \Pr[(\boldsymbol{\hat{H}'}, \mathbf{0}) \text{ is a } \mathsf{RSD_s} \text{ instance} \cap \mathcal{A}_\mathrm{s}(\boldsymbol{\hat{H}'}, \mathbf{0}) \neq \perp] \\ &= \varepsilon_1 \cdot \Pr[\mathcal{A}_\mathrm{s}(\boldsymbol{\hat{H}'}, \mathbf{0}) \neq \perp \mid (\boldsymbol{\hat{H}'}, \mathbf{0}) \text{ is a } \mathsf{RSD_s} \text{ instance}] \\ &= \varepsilon_1 \cdot \varepsilon_s \ . \end{aligned}$$

□

Remark 1. In practice, the loss factor in Proposition 1 tends to 1 when q grows. For our considered parameters, with $q = 2$, its value is around 0.3. Moreover, one can get the average number of codewords of $\mathcal{C}$ of weight r to justify our assumption. Let $\mathcal{S}_r = \prod_{i=0}^{r-1} \frac{(q^n - q^i)(q^m - q^i)}{q^r - q^i}$ be the number of words in $\mathbb{F}_{q^m}^n$ of weight exactly r. Then, on average, there are $\frac{S_r}{q^{m(n-k)}}$ words of rank r in the code. When below RGV, this makes the probability that a random code $\mathcal{C}$ contains no codeword of weight r close to 1.

Remark 2. The best known attacks on RSD use the reduction to $\mathsf{RSD_s}$ in order to solve the instance ([34], [6], [10], [8]), meaning that in practice we consider the best attacks on RSD to evaluate the security of $\mathsf{RSD_s}$.

3.2 Parameters choice for $\mathsf{RSD_s}$

We recall here the best attacks on RSD.

Ourivski-Johansson. The attack [34] first apply the reduction of Proposition 1, and exhibits a system of quadratic equations. The aim of this attack is to linearize the equations, which is done after fixing a number of values. This algorithm solves the problem in

$$\mathcal{O}\left((rm)^{\omega} q^{(r-1)(k+1)}\right).$$

AGHT: improved GRS. The idea of the GRS attack [24] is to sample a subspace E' of dimension $r' \geq r$, and hope that it includes $E = \mathrm{Supp}(\boldsymbol{x})$. Then, one solves a linear system, when $r' \leq \lfloor \frac{(n-k)m}{n} \rfloor$. The improvement of [6] uses the reduction of Proposition 1, where the success condition is if E' contains αE for any $\alpha \in \mathbb{F}_{q^m}^{*}$. The resulting complexity is

$$\mathcal{O}\left((n-k)^{\omega} m^{\omega} q^{r \lfloor \frac{(k+1)m}{n} \rfloor - m}\right).$$

Algebraic attacks. There are two main algebraic attacks for RSD. The first one is the *MaxMinors* modeling [7]. It consists in solving the minors of size r of the matrix $\boldsymbol{C}\boldsymbol{H}^{\top}$, where $\boldsymbol{x} = \boldsymbol{s}\boldsymbol{C}$ for $\boldsymbol{s} \in \mathbb{F}_{q^m}^{r}$ and $\boldsymbol{C} \in \mathbb{F}_{q}^{r \times n}$. The system is then solved, and yields a complexity of

$$\mathcal{O}\left(q^{ar} \binom{n-a-p}{r}^{\omega}\right)$$

where a is the parameter of the *Hybrid method* (see [8]), and p is the number of positions punctured.

The second algebraic attack [9] [8] is the *Support Minors*. In this modeling, one constructs a vector $\boldsymbol{v} = -\boldsymbol{m}\boldsymbol{G} + \boldsymbol{x}$ where $-\boldsymbol{m}\boldsymbol{G} \in \mathcal{C}$, and write it as a product $\boldsymbol{s}\boldsymbol{C}$ where $\boldsymbol{s} \in \mathbb{F}_{q^m}^{r}, \boldsymbol{C} \in \mathbb{F}_{q}^{r \times n}$. The equations come from $\begin{pmatrix} \boldsymbol{r}_i \\ \boldsymbol{C} \end{pmatrix}$ where $\boldsymbol{r}_i$ is the i-th row of $\boldsymbol{x} - \boldsymbol{m}\boldsymbol{G}$. When applying this modeling, by computing

$$\begin{aligned} N = & \sum_{i=1}^{k} \binom{n-a-i}{r} \binom{k-a+b-1-i}{b-1} - \binom{n-k-1}{r} \binom{k-a+b-1}{b} \\ & - (m-1) \sum_{i=1}^{b} (-1)^{i+1} \binom{k-a+b-i-1}{b-i} \binom{n-k-1}{r+i} \end{aligned}$$

and

$$M = \binom{k-a+b-1}{b} \left(\binom{n-a}{r} - m \binom{n-k-1}{r} \right),$$

as soon as $N \geq M - 1$, we obtain the complexity of

$$\mathcal{O}\left(q^{ar} m^2 N M^{\omega-1}\right)$$

where, as before, a is the parameter of the hybrid attack, and the parameter b minimizes the above quantities.

Parameter selection. According to the above attacks, we give in Table 3 the parameters which we will use for our RSD_s instances.

NIST Security level	q	m	n	k	r
I	2	53	53	45	4
III	2	79	75	67	4
V	2	97	95	87	4

Table 3: Choice of parameters for RSD_s

3.3 Parameters choice for MinRank

We now recall the attacks on MinRank. In this case, the *Hybrid method* works well for both combinatorial and algebraic attacks. In particular, for a cost of q^{ar} repetitions, it is possible to reduce a (q, m, n, k, r) MinRank instance into a $(q, m, n-a, k-am, r)$ one.

Kernel attack. The attack, introduced by Goubin and Courtois [26], consists in sampling randomly a matrix vectors of $\mathbb{F}_q^n$, and hoping they are in the right kernel of the matrix $\boldsymbol{E} = \boldsymbol{M} + \sum_{i=1}^{k} x_i \boldsymbol{M}_i$. Since the kernel is of dimension $n-r$, the probability to sample a vector in the kernel is $\frac{1}{q^r}$. When sampling l vectors and multiplying $\boldsymbol{E}$ by these vectors on the right, we obtain k unknowns and $m \cdot l$ equations. We are able to solve it when $l = \lceil \frac{k}{m} \rceil$. The overall complexity is thus

$$\mathcal{O}\left(k^{\omega} q^{r\lceil \frac{k}{m} \rceil}\right).$$

Algebraic attacks. As for RSD, the first algebraic attack is *MaxMinors* [9]. The modeling is simply to write $\boldsymbol{E} = \boldsymbol{M} + \sum_{i=1}^{k} x_i \boldsymbol{M}_i$, and to compute its minors of rank $r+1$. The complexity of the attack depends on the Hilbert series

$$HS(t)\left[(1-t)^{(m-r)(n-r)-(k+1)} \frac{\det(A(t))}{t^{\binom{r}{2}}}\right],$$
$$\text{with } A(t) = \left(\sum_{\ell=0}^{\max(m-i,n-j)} \binom{m-i}{\ell}\binom{n-j}{\ell} t^{\ell}\right)_{1\leq i\leq r, 1\leq j\leq r}$$

The total complexity is

$$\mathcal{O}\left(\binom{k+D}{D}^{\omega}\right)$$

where D is the degree of regularity of the system.

The second modeling, the *Support Minors* modeling [9] [8], allows to obtain equations by setting $\boldsymbol{E} = \boldsymbol{SC}$ where $\boldsymbol{S} \in \mathbb{F}_q^{m\times r}, \boldsymbol{C} \in \mathbb{F}_q^{r\times n}$, setting $\boldsymbol{r}_i$ the i-th row of $\boldsymbol{M} + \sum_{i=1}^{k}$, and computing the maximal minors of $\begin{pmatrix} \boldsymbol{r}_i \\ \boldsymbol{C} \end{pmatrix}$ The final complexity is

$$\mathcal{O}\left(NM^{\omega-1}\right)$$

where

$$N = \sum_{i=1}^{b}(-1)^{i+1}\binom{n}{r+i}\binom{k+b-1-i}{b-i}\binom{m+i-1}{i}$$

and

$$M = \binom{k+b-1}{b}\binom{n}{r},$$

with $N \geq M-1$ and $b \leq \min(q-1, r+1)$.
When $q = 2$, the complexity is slightly different, with

$$N = \sum_{j=1}^{b}\sum_{i=1}^{j}(-1)^{i+1}\binom{n}{r+i}\binom{k}{j-i}\binom{m+i-1}{i}$$

and

$$M = \sum_{j=1}^{b}\binom{k}{j}\binom{n}{r},$$

with $b < r+2$.

Parameter selection. According to the above attacks, we give in Table 4 the parameters which we will use for our MinRank instances.

NIST Security level	q	m	n	k	r
I	2	43	43	1520	4
III	2	60	60	3135	4
V	2	75	75	5040	4

Table 4: Choice of parameters for MinRank

4 MPCitH Modeling for $\mathsf{RSD_s}$ and MinRank

A zero-knowledge proof constructed using the MPCitH paradigm is composed of two parts, a “symmetric part” made of GGM trees (or Merkle trees) and an “arithmetic part” composed of the open party views and broadcast shares of the MPC protocol. While for a given security level the symmetric part is of rather fixed size (e.g., around 2kB for GGM trees and 4kB for Merkle trees at a 128-bit security level), the arithmetic part depends on the modeling (i.e., the way the problem instance is verified) and the associated MPC protocol. For the recent

TCitH and VOLEitH techniques, the arithmetic part is actually mainly impacted by the size of the witness, which favors modelings with low-size witnesses.

In this section, we study different modelings for RSD and $\mathsf{MinRank}$ with respect to the witness size criterion. For the RSD problem, we recall the permuted secret, q-polynomial and Kipnis-Shamir modelings. We propose an other modeling, named *dual support decomposition*, which can be seen as an improvement of the rank decomposition from [19]. We also slightly improve all the modelings by relying on the RSD_s variant. For the $\mathsf{MinRank}$ problem, we recall the q-polynomial and Kipnis-Shamir modelings and propose an adaptation of the dual support decomposition modeling for $\mathsf{MinRank}$.

4.1 Modelings for the RSD_s Problem

Permuted Secret. We start by recalling the permuted secret technique, which was used for RSD in [15]. The idea of this technique consists in revealing a "permuted" and a "masked" versions of the secret: let us denote σ an isometry in the rank metric (such a isometry consists of multiplying the secret matrix by a invertible matrices on both sides) and $\boldsymbol{u}$ a vector of the left kernel of $\boldsymbol{H}$, one reveals $\boldsymbol{v} := \sigma(\boldsymbol{x})$ and $\tilde{\boldsymbol{x}} := \boldsymbol{x} + \boldsymbol{u}$ and the goal is to find such values σ and $\boldsymbol{u}$. More precisely, the rank syndrome decoding problem consists, from two vectors $\boldsymbol{v}, \tilde{\boldsymbol{x}} \in \mathbb{F}_{q^m}^n$ satisfying $w_R(\boldsymbol{v}) = r$ and $\boldsymbol{H}\tilde{\boldsymbol{x}}^\top = \boldsymbol{y}^\top$, in finding an isometry σ and a vector $\boldsymbol{u} \in \mathbb{F}_{q^m}^n$ such that

$$\begin{cases} \boldsymbol{H}\boldsymbol{u}^\top = \boldsymbol{0}^\top, \\ \sigma(\tilde{\boldsymbol{x}}) = \boldsymbol{v} + \sigma(\boldsymbol{u}). \end{cases}$$

Indeed, if we get both σ and $\boldsymbol{u}$, we can easily restore the initial secret as $\boldsymbol{x} := \tilde{\boldsymbol{x}} - \boldsymbol{u}$: we have $\boldsymbol{H}\boldsymbol{x}^\top = \boldsymbol{y}^\top - \boldsymbol{0}^\top$ and $w_R(\boldsymbol{x}) = w_R(\sigma(\boldsymbol{x})) = w_R(\sigma(\tilde{\boldsymbol{x}}) - \sigma(\boldsymbol{u})) = w_R(\boldsymbol{v}) = r$.

Unfortunately, this modeling is not compatible with the recent MPCitH techniques as TCitH or VOLEitH. Such techniques requires at least additive sharings over a commutative group (or for the more recent techniques, Shamir's secret sharing over a ring). However, the isometry σ lives in a non-commutative group, so it requires to rely on a special form of MPCitH named the shared-permutation framework [15,20].

Rank Decomposition. The Rank Decomposition protocol proposed in [19] aims to verify the rank of the witness by using the rank decomposition theorem. In this modeling, the rank syndrome decoding problem consists in finding a vector $\boldsymbol{x} \in \mathbb{F}_{q^m}^n$ and two matrices $\boldsymbol{T} \in \mathbb{F}_q^{m \times r}$ and $\boldsymbol{R} \in \mathbb{F}_q^{r \times n}$ such that

$$\boldsymbol{H}\boldsymbol{x}^\top = \boldsymbol{y}^\top \quad \text{and} \quad \boldsymbol{X} = \boldsymbol{T}\boldsymbol{R},$$

where $\boldsymbol{X} \in \mathbb{F}_q^{m \times n}$ is the matrix associated to $\boldsymbol{x}$. Concretely, the MPC protocol takes as input some shares of $\boldsymbol{x}$, of $\boldsymbol{T}$ and of $\boldsymbol{R}$. The protocol then checks that $\boldsymbol{X} = \boldsymbol{T}\boldsymbol{R}$, where the shares of $\boldsymbol{X}$ is derived from those of $\boldsymbol{x}$. Using the standard representation $\boldsymbol{H} = \big(\boldsymbol{I_{n-k}} || \boldsymbol{H'}\big)$, one can send only the right part of $\boldsymbol{x}$ of size k, denoted as $\boldsymbol{x_B}$. Furthermore, it is possible to send one less column of the matrix $\boldsymbol{T}$, since $1 \in \mathrm{Supp}(\boldsymbol{x})$ (see [4] for the optimization) and as a result the size of witness is (in bits):

$$(\underbrace{k \cdot m}_{\boldsymbol{x_B}} + \underbrace{(r-1) \cdot m}_{\boldsymbol{T}} + \underbrace{r \cdot (n-r)}_{\boldsymbol{R}}) \cdot \log_2(q).$$

q-Polynomial. The q-polynomial technique proposed in [19] to check the rank metric constitutes an improvement compared to a number of previous methods. Let us first recall the definition of a q-polynomial.

Definition 7 (q-polynomial). *A q-polynomial of q-degree r is a polynomial in $\mathbb{F}_{q^m}[X]$ of the form:*

$$P(X) = X^{q^r} + \sum_{i=0}^{r-1} p_i \cdot X^{q^i} \qquad \textit{with } p_i \in \mathbb{F}_{q^m}.$$

The roots of a q-polynomial of q-degree r form a linear subspace of $\mathbb{F}_{q^m}$ of dimension at most r. Moreover, for each linear subspace of $\mathbb{F}_{q^m}$ of dimension at most r, there exists a unique monic q-polynomial of q-degree r annihilating all the elements of the subspace. Let $\boldsymbol{x} = (x_1, \dots, x_n) \in \mathbb{F}_{q^m}^n$ of rank $w_R\big(\boldsymbol{x}\big) = r$ and let $P_{\boldsymbol{x}}(X)$ the monic q-polynomial annihilating $\mathrm{Supp}(\boldsymbol{x})$. In this modeling, the rank syndrome decoding problem consists in finding a vector $\boldsymbol{x} \in \mathbb{F}_{q^m}^n$ and a q-polynomial $P_{\boldsymbol{x}} \in \mathbb{F}_{q^m}[X]$ of q-degree r such that

$$\boldsymbol{H}\boldsymbol{x}^\top = \boldsymbol{y}^\top \quad \text{and} \quad \forall i\ , P_{\boldsymbol{x}}(x_i) = 0.$$

Concretely, the MPC protocol based on the q-polynomial technique takes as input some shares of $\boldsymbol{x}$ and some shares of $P_{\boldsymbol{x}}(X)$. The protocol then checks that $P_{\boldsymbol{x}}(x_i) = 0$ for all $i \in [1, n]$. As previously, one can send only the right part of $\boldsymbol{x}$ of size k using the standard representation of $\boldsymbol{H}$. Furthermore, it is possible to send one less coefficient of the polynomial $P_{\boldsymbol{x}}$ when relying on $\mathsf{RSD_s}$. As a result the size of witness is (in bits):

$$(\underbrace{k \cdot m}_{\boldsymbol{x_B}} + \underbrace{(r-1) \cdot m}_{P_{\boldsymbol{x}}}) \cdot \log_2(q).$$

This modeling based on q-polynomials currently leads to the shortest communications for RSD when considering *linear* multiparty computation, but it is not the best one when considering non-linear multiparty computation as in the new MPCitH frameworks.

Kipnis-Shamir. Historically, the Kipnis-Shamir modeling was introduced in the cryptanalysis of the MinRank problem [29]. We can use the same idea to have a modeling of RSD. It consists in giving the right-kernel of the matrix of $\boldsymbol{x}$. We denote this matrix in $\mathbb{F}_q^{m\times n}$ by $\boldsymbol{X}$. If $w_R\big(\boldsymbol{x}\big) = r$, then the right-kernel of $\boldsymbol{X}$ is of dimension $n-r$ and can be represented by an $r\times(n-r)$ matrix.

In the $\mathsf{RSD}_{\mathrm{s}}$ case, the witness is composed of $\boldsymbol{x}$ and of the matrix $\boldsymbol{A} \in \mathbb{F}_q^{r\times(n-r)}$. The MPC protocol takes as input $\boldsymbol{K} = \begin{pmatrix}\boldsymbol{I_{n-r}}\\ \boldsymbol{A}\end{pmatrix}$, and then checks that $\boldsymbol{XK} = \boldsymbol{0}$. It is possible to send only $\boldsymbol{x_B}$, as previously with q-polynomials, and since 1 is in the support, the size of the witness is:

$$(\underbrace{k\cdot m}_{\boldsymbol{x_B}} + \underbrace{(r-1)\cdot(n-r)}_{\boldsymbol{A}})\cdot\log_2(q)\ .$$

Note that transmitting $\boldsymbol{A}$ costs $(r-1)\cdot(n-r)$ only since we know that 1 is in $\boldsymbol{x}$. This approach is slightly better than the q-polynomial technique in terms of witness size.

Dual Support Decomposition. Finally, we introduce an other modeling for $\mathsf{RSD}_{\mathrm{s}}$, using only the support and the coordinates. This can be seen as an improvement of the rank decomposition from [19], as our new modeling does not rely on having $\boldsymbol{x_B} \in \mathbb{F}_{q^m}^k$ as input. To that end, one has as inputs:

- The support of $\boldsymbol{x}$, $\mathrm{Supp}(\boldsymbol{x}) = \langle 1, x_2, \ldots, x_r\rangle$;
- The coordinates of $\boldsymbol{x}$ in this basis, i.e, $\boldsymbol{C} \in \mathbb{F}_q^{r\times(n-r)}$ such that

$$(1, x_2, \ldots, x_r)\cdot\big(\boldsymbol{I_r}\ \boldsymbol{C}\big) = (1, x_2, \ldots, x_n) = \boldsymbol{x}$$

More precisely, in this modeling, the $\mathsf{RSD}_{\mathrm{s}}$ problem consists in finding $x_2, \ldots, x_r \in \mathbb{F}_{q^m}$ and $\boldsymbol{C} \in \mathbb{F}_q^{r\times(n-r)}$ such that

$$\boldsymbol{Hx}^\top = \boldsymbol{y}^\top \quad \text{where} \quad \boldsymbol{x} := (1, x_2, \ldots, x_r)\cdot\big(\boldsymbol{I_r}\ \boldsymbol{C}\big)\,.$$

Concretely, after computing $\boldsymbol{x} = (x_1, \ldots, x_r)\cdot\big(\boldsymbol{I_r}\ \boldsymbol{C}\big)$, one verifies that $\boldsymbol{Hx}^T$ is indeed equal to $\boldsymbol{y}^T$. Since 1 is in the support of $\boldsymbol{x}$, it is possible to transmit only $r-1$ elements for $\mathrm{Supp}(\boldsymbol{x})$, and we can have a gain on the matrix $\boldsymbol{C}$ as well since the r first coordinates are linearly independent. This results in an efficient protocol, where the inputs are of size

$$(\underbrace{(r-1)\cdot m}_{\mathrm{Supp}(\boldsymbol{x})} + \underbrace{r\cdot(n-r)}_{\boldsymbol{C}})\cdot\log_2(q)$$

We see here that the input size does not depend on k anymore, allowing us to take more efficient parameters.

Global Comparison. Table 5 provides a global comparison of the different modelings in terms of witness size for the RSD problem. For each of the described modelings, we provide the size formula as well as the obtained concrete size for optimized parameters reaching a 128-bit security according to the attacks in Section 3.2.

Modeling	**Witness size**	**Parameters for** $\lambda = 128$	
		(q, m, n, k, r)	**Size**
Rank Decomposition	$[\,km + (r-1)m + r(n-r)\,] \cdot \log_2(q)$	$(2, 31, 33, 15, 10)$	122 B
q-polynomial	$[\,km + (r-1)m\,] \cdot \log_2(q)$	$(2, 31, 33, 15, 10)$	93 B
Kipnis-Shamir	$[\,km + (r-1)(n-r)\,] \cdot \log_2(q)$	$(2, 31, 33, 15, 10)$	86 B
Dual Support Decomp.	$[\,(r-1)m + r(n-r)\,] \cdot \log_2(q)$	$(2, 53, 53, 45, 4)$	45 B

Table 5: Witness size for different MPCitH modelings for the $\mathsf{RSD_s}$ problem.

4.2 Modelings for the MinRank Problem

The MinRank problem is closely related to the RSD problem. The two problems indeed share a number of similarities as evidence of the algebraic attacks applying to both problems (see, e.g., [8,10]). Quite naturally, most of the above modelings for RSD can be adapted for MinRank.

Rank Decomposition. Similar to $\mathsf{RSD_s}$, the Rank Decomposition protocol was introduced in [19]. This protocol takes as input (shares of) $\boldsymbol{x} \in \mathbb{F}_q^k$, $\boldsymbol{T} \in \mathbb{F}_q^{m \times r}$, and $\boldsymbol{R} \in \mathbb{F}_q^{r \times n}$. Then, after building $\boldsymbol{E} = \boldsymbol{M} + \sum_{i=1}^{k} x_i \boldsymbol{M}_i$, the protocol checks that $\boldsymbol{E} = \boldsymbol{T}\boldsymbol{R}$. This leads to a witness size (in bits) of

$$(\underbrace{k}_{\boldsymbol{x}} + \underbrace{r \cdot (m-r)}_{\boldsymbol{T}} + \underbrace{r \cdot n}_{\boldsymbol{R}}) \cdot \log_2(q).$$

q-Polynomial. The q-polynomial technique of [19] can be also applied to MinRank: the witness is composed of the shares of $\boldsymbol{x} \in \mathbb{F}_q^k$ and the coefficients $\boldsymbol{\beta} \in \mathbb{F}_{q^m}^r$ of the q-polynomial associated to $\boldsymbol{E}$. The MPC protocol computes $\boldsymbol{E} = \boldsymbol{M} + \sum_{i=1}^{k} x_i \boldsymbol{M}_i$ and verifies that $P_{\boldsymbol{E}}(X) := \sum_{i=0}^{r-1} \beta_i X^{q^i} + X^{q^r}$ is the annihilator polynomial of $\boldsymbol{E}$. This verification relies on the isomorphism between $\mathbb{F}_{q^m}$ and $\mathbb{F}_q^m$, and associates each column of $\boldsymbol{E}$, denoted as $\boldsymbol{e}_i$, to an element of $\mathbb{F}_{q^m}$, e_i. The protocol hence simply checks that $P_{\boldsymbol{E}}(e_i) = 0$ for $i \in [1, n]$.

With this modeling, the size of the witness size is (in bits):

$$(\underbrace{k}_{\boldsymbol{x}} + \underbrace{r \cdot m}_{P_{\boldsymbol{E}}}) \cdot \log_2(q)\ .$$

Kipnis-Shamir. This is the modeling used in MiRitH [1], which is an improvement of MinRank-in-the-Head [2]. The goal of this modeling is to use the right kernel of $\boldsymbol{E}$ in order to prove its rank. Let $\boldsymbol{K} = \begin{bmatrix} \boldsymbol{I_{n-r}} \\ \boldsymbol{A} \end{bmatrix}$ a matrix of rank $n-r$ representing the right kernel of $\boldsymbol{E}$. The witness is composed of $\boldsymbol{x}$ and $\boldsymbol{A} \in \mathbb{F}_q^{r \times (n-r)}$. The protocol recomputes $\boldsymbol{E} = \boldsymbol{M} + \sum_{i=1}^{k} x_i \boldsymbol{M}_i$ and verifies that $\boldsymbol{E} \cdot \boldsymbol{K} = \boldsymbol{0}$. If the verification succeeds, one deduces that $\boldsymbol{E}$ is indeed of rank r since it has a kernel of rank $n-r$.

With this modeling, the witness is of size:

$$(\underbrace{k}_{\boldsymbol{x}} + \underbrace{r \cdot (n-r)}_{\boldsymbol{A}}) \cdot \log_2(q) \ .$$

As for $\mathsf{RSD_s}$, the witness is smaller with this modeling than with the q-polynomials technique.

New Modeling for the MinRank *Problem: Dual Support Decomposition.* We introduce hereafter a new MPCitH modeling for the MinRank problem which achieves smaller witness sizes than the previous modelings. To build our modeling, we will rely on an alternative formulation of this problem, namely, its syndrome version, which has not been previously used to build cryptosystems. More precisely, this problem can be expressed in a syndrome decoding way by using the dual of the matrix code generated by $\boldsymbol{M}_1, \ldots, \boldsymbol{M}_k$. First, one can define the map

$$\begin{array}{cccc} \rho : & \mathbb{F}_q^{m \times n} & \to & \mathbb{F}_q^{mn} \\ & \begin{pmatrix} a_{1,1} & \ldots & a_{1,n} \\ \vdots & & \vdots \\ a_{m,1} & \ldots & a_{m,n} \end{pmatrix} & \mapsto & \begin{pmatrix} a_{1,1}, \ldots, a_{1,n}, \ldots, a_{m,1}, \ldots, a_{m,n} \end{pmatrix} \ . \end{array}$$

The generator matrix of the code $\langle \boldsymbol{M}_1, \ldots, \boldsymbol{M}_k \rangle$ is the matrix $\boldsymbol{G} \in \mathbb{F}_q^{k \times mn}$ whose lines are $\rho(\boldsymbol{M}_1), \ldots, \rho(\boldsymbol{M}_k)$, i.e,

$$\boldsymbol{G} = \begin{pmatrix} \rho(\boldsymbol{M}_1) \\ \vdots \\ \rho(\boldsymbol{M}_k) \end{pmatrix} \ .$$

Solving the MinRank problem then consists, given $\boldsymbol{G}$ and $\boldsymbol{M}$, in finding $\boldsymbol{x}$ and $\boldsymbol{E}$ such that $\rho(\boldsymbol{M}) = -\boldsymbol{x}\boldsymbol{G} + \rho(\boldsymbol{E})$ and $\mathsf{rank}(\boldsymbol{E}) \leq r$. The dual matrix of this code is, as usual, a full-rank matrix $\boldsymbol{H} \in \mathbb{F}_q^{(mn-k) \times mn}$ such that $\boldsymbol{G}\boldsymbol{H}^\top = \boldsymbol{0}$. Then, the MinRank Syndrome problem can be defined by applying $\boldsymbol{H}^\top$ in the linear constraint: given $\boldsymbol{H}$ and $\boldsymbol{y} := \rho(\boldsymbol{M})\boldsymbol{H}^\top$, the MinRank Syndrome problem consists in finding $\boldsymbol{E}$ such that $\boldsymbol{y} = \boldsymbol{0} + \rho(\boldsymbol{E})\boldsymbol{H}^\top$ and $\mathsf{rank}(\boldsymbol{E}) \leq r$.

Definition 8 (MinRank Syndrome **problem).** *Let q, m, n, k and r be positive integers. Let $\boldsymbol{H} := \begin{bmatrix} \boldsymbol{I_{mn-k}} \ \boldsymbol{H'} \end{bmatrix} \in \mathbb{F}_q^{(mn-k) \times mn}$ where $\boldsymbol{H'} \in \mathbb{F}_q^{(mn-k) \times k}$, and $\boldsymbol{y} \in \mathbb{F}_q^{mn-k}$. The* MinRank Syndrome *problem asks to find $\boldsymbol{E}$ such that $\rho(\boldsymbol{E})\boldsymbol{H}^\top = \boldsymbol{y}$ and* $\mathsf{rank}(\boldsymbol{E}) \leq r$.

Proposition 2. *The* MinRank *problem and the* MinRank Syndrome *problem are equivalent.*

Proof. The proof is similar to the proof of equivalence between the decoding problem and the syndrome decoding problem for the Hamming metric.

- Let us explain how to solve a $\mathsf{MinRank\ Syndrome}$ instance $(\boldsymbol{H}, y)$ using a $\mathsf{MinRank}$ solver. The first step consists in finding a solution $\rho(\boldsymbol{M})$ to the linear system $\rho(\boldsymbol{M})\boldsymbol{H}^\top = \boldsymbol{y}$ without any constraint on the rank of $\boldsymbol{M}$. Then, we consider a dual matrix $\boldsymbol{G}$ of $\boldsymbol{H}$, where we interpret each row as a flatten matrix among $\boldsymbol{M}_1, \ldots, \boldsymbol{M}_k$. We can then find $\boldsymbol{x}$ and $\boldsymbol{E}$ such that $\boldsymbol{M} + \sum_{i=1}^k x_i \boldsymbol{M}_i = \boldsymbol{E}$ and $\mathsf{rank}(\boldsymbol{E}) \le r$ using the $\mathsf{MinRank}$ solver. Such an $\boldsymbol{E}$ is directly a solution of our $\mathsf{MinRank\ Syndrome}$ instance, since

$$\rho(\boldsymbol{E})\boldsymbol{H}^\top = [-\boldsymbol{x}\boldsymbol{G} + \rho(\boldsymbol{E})]\boldsymbol{H}^\top = \rho(\boldsymbol{M})\boldsymbol{H}^\top = \boldsymbol{y}.$$

- Let us now explain how to solve a $\mathsf{MinRank}$ instance $(\boldsymbol{M}, \boldsymbol{M}_1, \ldots, \boldsymbol{M}_k)$ using a $\mathsf{MinRank\ Syndrome}$ solver. First, we can build the dual matrix $\boldsymbol{H}$ from $\boldsymbol{M}_1, \ldots, \boldsymbol{M}_k$, such that $\boldsymbol{G}\boldsymbol{H}^\top = \boldsymbol{0}$ where the rows of $\boldsymbol{G}$ are defined as $\rho(\boldsymbol{M}_1), \ldots, \rho(\boldsymbol{M}_k)$. We can then find $\boldsymbol{E}$ such that $\rho(\boldsymbol{E})\boldsymbol{H}^\top = \rho(\boldsymbol{M})\boldsymbol{H}^\top$ and $\mathsf{rank}(\boldsymbol{E}) \le r$ using the $\mathsf{MinRank\ Syndrome}$ solver (defining the syndrome as $\boldsymbol{y} = \rho(\boldsymbol{M})\boldsymbol{H}^\top$). Therefore, $\rho(\boldsymbol{E} - \boldsymbol{M})$ is in the right kernel of $\boldsymbol{H}$, for which a basis is described by the rows of $\boldsymbol{G}$. By Gaussian elimination, we can thus find $\boldsymbol{x}$ such that $\rho(\boldsymbol{E} - \boldsymbol{M}) = \boldsymbol{x}\boldsymbol{G}$, i.e., such that $\rho(\boldsymbol{E} - \boldsymbol{M}) = \sum_{i=1}^k x_i \cdot \rho(\boldsymbol{M}_i)$. Such an $\boldsymbol{x}$, together with $\boldsymbol{E}$, forms a solution of our $\mathsf{MinRank}$ instance.

□

In the *Dual Support Decomposition* modeling, we rely on the $\mathsf{MinRank\ Syndrome}$ problem instead of the standard $\mathsf{MinRank}$ problem. In this modeling, the protocol aims at verifying that a matrix $\boldsymbol{E}$ is solution of the constraints $\rho(\boldsymbol{E})\boldsymbol{H}^\top = \boldsymbol{y}$ and $\mathsf{rank}(\boldsymbol{E}) \le r$ for some $\boldsymbol{H} \in \mathbb{F}_q^{(mn-k)\times mn}$ and $\boldsymbol{y} \in \mathbb{F}_q^{mn-k}$. As in the rank decomposition method from [19], one can view $\boldsymbol{E}$ as a product of two matrices, $\boldsymbol{E} = \boldsymbol{S}\boldsymbol{C}$, with $\boldsymbol{S} \in \mathbb{F}_q^{m\times r}$ and $\boldsymbol{C} \in \mathbb{F}_q^{r\times n}$. Furthermore, one can write without loss of generality $\boldsymbol{S}$ as $\begin{bmatrix}\boldsymbol{I_r}\\ \boldsymbol{S'}\end{bmatrix}$ for some matrix $\boldsymbol{S'} \in \mathbb{F}_q^{(m-r)\times r}$ (this is always possible up to a permutation of the lines). Then, one can simply set $\boldsymbol{E} = \begin{bmatrix}\boldsymbol{I_r}\\ \boldsymbol{S'}\end{bmatrix} \cdot \boldsymbol{C}$. Therefore, given $\boldsymbol{S'}$ and $\boldsymbol{C}$, the protocol simply checks:

$$\rho(\boldsymbol{S}\boldsymbol{C})\boldsymbol{H}^\top = \boldsymbol{y} \quad \text{with} \quad \boldsymbol{S} = \begin{bmatrix}\boldsymbol{I_r}\\ \boldsymbol{S'}\end{bmatrix}.$$

The use of the dual form of the $\mathsf{MinRank}$ matrix code makes a significant difference compared to the Rank Decomposition, as the scheme now avoids relying on the vector $\boldsymbol{x} \in \mathbb{F}_q^k$, leading to a *much smaller* witness. Overall, with the identity in $\boldsymbol{S}$, the inputs are of size

$$(\underbrace{r \cdot (m-r)}_{\boldsymbol{S}} + \underbrace{r \cdot n}_{\boldsymbol{C}}) \cdot \log_2(q).$$

As for RSD, the size does not depend on k anymore, which allows a better selection of the parameters.

Global comparison. Table 6 provides a global comparison of the different modelings in terms of witness size for the MinRank problem. For each of the described modelings, we provide the size formula as well as the obtained concrete size for optimized parameters reaching a 128-bit security for the attacks described in section 3.3.

Modeling	**Witness size**	**Parameters for $\lambda = 128$**	
		(q, m, n, k, r)	**Size**
Rank Decomposition	$[\,k + r(m - r) + rn\,] \cdot \log_2(q)$	$(16, 15, 15, 78, 6)$	111 B
q-polynomial	$[\,k + rm\,] \cdot \log_2(q)$	$(16, 15, 15, 78, 6)$	76 B
Kipnis-Shamir	$[\,k + r(n - r)\,] \cdot \log_2(q)$	$(16, 15, 15, 78, 6)$	66 B
Dual Support Decomp.	$[\,r(m - r) + rn\,] \cdot \log_2(q)$	$(2, 43, 43, 1520, 4)$	41 B

Table 6: Modeling for MinRank and resulting witness size in MPC protocols.

5 The TCitH and VOLEitH Frameworks

The MPCitH paradigm [27] is a versatile method introduced in 2007 to build zero-knowledge proof systems using techniques from secure multi-party computation (MPC). This paradigm has been drastically practically improved in recent years (see, e.g., [3,18,22,28]) and is particularly efficient to build zero-knowledge proofs for small circuits such as those involved in (post-quantum) signature schemes. The MPCitH paradigm can be summarized as follows. The prover emulates “in his head” an ℓ-private MPC protocol with N parties and commits each party’s view independently. The verifier then challenges the prover to reveal the views of a random subset of ℓ parties. By the privacy of the MPC protocol, nothing is revealed about the plain input, which implies the zero-knowledge property. On the other hand, a malicious prover needs to cheat for at least one party, which shall be discovered by the verifier with high probability, hence ensuring the soundness property.

In what follows, we describe two recently introduced MPCitH-based frameworks, namely the *VOLE-in-the-Head* (VOLEitH) framework from [12] and the *Threshold-Computation-in-the-Head* (TCitH) framework from [21,22]. We then present the recent optimisations proposed by [11].

5.1 Threshold-Computation-in-the-Head Framework

The TCitH framework has been recently introduced in [21] as an extension of a previous work [22] published at Asiacrypt 2023. While almost all the former MPCitH-based proof system relied on additive sharings, the TCitH framework shows how using Shamir’s secret sharings (instead of additives sharings) lead to faster schemes with shorter communication.

We refer the reader to [21,22] for a detailed exposition of the TCitH framework which is only briefly abstracted here. In a nutshell, the TCitH framework relies on MPC protocols with broadcasting, randomness oracle and hint oracle (as previous MPCitH schemes) but using Shamir's secret sharing unlocks the use of non-linear multiparty computation (whereas previous MPCitH schemes are based on linear multiparty computation). More precisely, in the considered MPC protocols, one can compute a sharing $[\![a \cdot b]\!]$ of a product $a \cdot b$ from the sharings $[\![a]\!]$ and $[\![b]\!]$ of the operands by share-wise multiplication (for all i, $[\![a \cdot b]\!]_i \leftarrow [\![a]\!]_i \cdot [\![b]\!]_i$).

The TCitH framework comes with two variants depending on how one commits the input shares: either relying on GGM trees [25] or on Merkle trees [32]. In the present work, we focus on the GGM-tree variant which leads to shorter signature sizes for the considered statements. Moreover, we only consider 1-private Shamir's secret sharings, *i.e.* $\ell = 1$, which gives the best results in our context.

Given some degree-d polynomials $f_1, \ldots, f_m$ from $\mathbb{F}[X_1, \ldots, X_{|w|}]$, we want a zero-knowledge proof of knowledge of a witness w satisfying

$$\forall j \in [1, m], \;\; f_j(w) = 0.$$

We shall use the proof system TCitH-$\mathit{\Pi}_{\text{PC}}$ described in [21, Section 5.2]. We recall the underlying MPC protocol $\mathit{\Pi}_{\text{PC}}$ in Protocol 1. The sharing $[\![0]\!]$ used in Step 4 of the MPC protocol is a publicly-known degree-1 sharing of zero (for example, $[\![0]\!]_i = \omega_i$ when $e = 0$). This MPC protocol is ℓ-private and sound with false positive probability $\frac{1}{|\mathbb{F}|}$ (see [21, Lemma 2]). In practice, the MPC protocol is repeated ρ times in parallel to achieve a false positive probability of $\frac{1}{|\mathbb{F}|^\rho}$. The soundness error of TCitH-$\mathit{\Pi}_{\text{PC}}$ (when $\ell = 1$) is

$$\epsilon = \frac{1}{|\mathbb{F}|^\rho} + \left(1 - \frac{1}{|\mathbb{F}|^\rho}\right) \cdot \frac{d}{N} \; .$$

To obtain a signature scheme, we first transform the above MPC protocol into a proof of knowledge (PoK) of soundness error ϵ by applying the TCitH transform. We then perform τ parallel repetitions of this PoK and apply the Fiat-Shamir transform [23]. To achieve a λ-bit security, we take the number ρ of MPC repetitions such that $\frac{1}{|\mathbb{F}|^\rho} \leq 2^{-\lambda}$ and the number τ of PoK repetitions such that $\left(\frac{d}{N}\right)^\tau \leq 2^{-\lambda}$.

The proof transcript (*i.e.* the signature) includes:

- The opened shares $[\![w]\!]_I$ of the witness $w \in \mathbb{F}^{|w|}$, for each of the τ PoK repetitions. In practice, the sent values are the auxiliary values $\mathit{\Delta} w$.
- The opened shares of $[\![v]\!]_I$: because v is uniformly-sampled, these shares are communication-free since we rely on the TCitH-GGM variant.
- The degree-d sharing $[\![\alpha]\!]$, for each of the ρ MPC repetitions of the τ PoK repetitions. Since $[\![\alpha]\!]_I$ can be recomputed by the verifier and since the α should be zero, the prover just needs to send $(d+1) - 1 - 1 = (d-1)$ shares.
- The sibling paths in the GGM trees, together with the unopened seed commitments.

1. The parties receive a sharing $\llbracket w \rrbracket$, with $\deg\llbracket w \rrbracket = 1$.
2. The parties get a uniformly-random degree-$(d-1)$ sharing $\llbracket v \rrbracket$ of a random value $v \in \mathbb{F}$ from O_H.
3. The parties receive random values $\gamma_1, \ldots, \gamma_m \in \mathbb{F}$ from O_R.
4. The parties locally compute
$$\llbracket \alpha \rrbracket = \llbracket v \rrbracket \cdot \llbracket 0 \rrbracket + \sum_{j=1}^{m} \gamma_j \cdot f_j(\llbracket w \rrbracket) .$$
5. The parties open $\llbracket \alpha \rrbracket$ to publicly recompute α.
6. The parties output Accept if and only if $\alpha = 0$.

Protocol 1: $\mathit{\Pi}_{\text{PC}}$ – Verification of polynomial constraints. O_R is an oracle which provides public trusted randomness to the parties: in a MPCitH setting, this randomness is provided by the verifier. O_H is an oracle which provides sharings of untrusted values named hints: in a MPCitH setting, these sharings are provided by the prover.

Moreover, the signature includes a 2λ-bit salt and a 2λ-bit commitment digest that correspond to the last verifier challenge (in the Fiat-Shamir heuristic). Therefore, the signature size when using the TCitH framework in the above setting is (in bits):

$$\textsc{Size}_{\textsc{TCitH}} = 4\lambda + \tau \cdot \left(\underbrace{|w| \cdot \log_2 |\mathbb{F}|}_{\llbracket w \rrbracket_I} + \underbrace{(d-1) \cdot \rho \cdot \log_2 |\mathbb{F}|}_{\llbracket \alpha \rrbracket} + \underbrace{\lambda \cdot \log_2 N}_{\text{GGM tree}} + 2\lambda \right) .$$

5.2 VOLE-in-the-Head Framework

The VOLEitH framework has been introduced at Crypto 2023 [12]. This work provides a way to compile any zero-knowledge protocol in the VOLE-hybrid model into a publicly verifiable protocol. While it has not been introduced as a MPCitH construction, it can yet be interpreted as such. Specifically, [21] shows that the VOLEitH framework can be described in the TCitH syntax. Indeed, this framework is similar to the TCitH framework with $\ell = 1$ and GGM trees, up to several details:

- The secret is stored at $P(\infty)$ when sharing, meaning that $e = \infty$. As a result, to share a value v, one samples a random value r and builds the Shamir's polynomial P as $P(X) := vX + r$. While multiplying two Shamir's sharings when $e = \infty$ is similar than when $e \neq \infty$, the addition operation is slightly different: to add two Shamir's sharings $\llbracket a \rrbracket$ and $\llbracket b \rrbracket$ of degrees respectively d_1 and d_2 (such that $d_1 \leq d_2$) when $e = \infty$, the parties can compute the following d_2-degree sharing
$$\forall i,\ \llbracket a + b \rrbracket_i \leftarrow \llbracket a \rrbracket_i \cdot \omega_i^{d_2 - d_1} + \llbracket b \rrbracket_i ,$$

where ω_i is the evaluation point of the i^{th} party.

- The VOLEitH framework relies on a *large field embedding*: in the commitment phase, the prover commits τ N-sharings $[\![w]\!]^{(1)}, \ldots, [\![w]\!]^{(\tau)}$ of the witness w. In the basic TCitH framework, the prover runs τ MPC protocols in parallel, each of them on a different sharing $[\![w]\!]^{(j)}$. In the VOLEitH framework, these N sharings are merged to obtain a N^τ-sharing $[\![w]\!]^{(\phi)}$ living in a large field extension $\mathbb{K}$ such that the extension degree $[\mathbb{K} : \mathbb{F}]$ is ρ, then the prover runs a unique MPC protocol which takes as input this N^τ-sharing. More precisely, the i^{th} share of $[\![w]\!]^{(\phi)}$ is computed as

$$[\![w]\!]_i^{(\phi)} \leftarrow \phi\left([\![w]\!]_{i_1}^{(1)}, \ldots, [\![w]\!]_{i_\tau}^{(\tau)}\right)$$

  where $i_1, \ldots, i_\tau \in [1, N]$ satisfy $(i-1) = (i_1 - 1) + (i_2 - 1) \cdot N + \ldots + (i_\tau - 1) \cdot N^{\tau-1}$ and ϕ is an one-to-one ring homomorphism between $\mathbb{F}^\tau$ and $\mathbb{K}$ $(\rho \geq \tau)$. If the sharings $[\![w]\!]^{(1)}, \ldots, [\![w]\!]^{(\tau)}$ encode the *same* witness w, then we get that $[\![w]\!]^{(\phi)}$ is a valid Shamir's secret sharing of w for which the evaluation point of the i^{th} party is $\phi(\omega_{i_1}, \ldots, \omega_{i_\tau})$ (with ω_i the i^{th} party evaluation point in the standard TCitH setting). The main advantage of this large field embedding is that the resulting soundness error of the proof system is $\frac{d}{N^\tau}$ instead of being $\left(\frac{d}{N}\right)^\tau$ (up to the false positive probability).
- The above optimisation requires that the prover ensures that the τ sharings encode the same value (without revealing this value). To ensure this property, the VOLEitH framework introduces an additional prover-verifier pair of rounds. After committing the input shares (including the hint sharings),
  - the prover commits τ additional uniformly-random sharings $[\![u]\!]^{(1)}, \ldots,$ $[\![u]\!]^{(\tau)}$ of the *same* random value $u \in \mathbb{F}^{\rho+B}$, for $B \geq 0$ an additional parameter,
  - the verifier sends a challenge $(H_1|H_2) \in \mathbb{F}^{(\rho+B)\times(n+\rho)}$,
  - for all $j \in [1, \tau]$, the prover reveals the *digest sharing* $[\![\alpha']\!]^{(j)} := H_1[\![w]\!]^{(j)} + H_2[\![v]\!]^{(j)} + [\![u]\!]^{(j)}$, where $\alpha' \in \mathbb{F}^{\rho+B}$.

  The idea behind this process is that the prover computes the digests of all the plain values encoded in $[\![w]\!]^{(1)}, \ldots, [\![w]\!]^{(\tau)}$ (and in $[\![v]\!]^{(1)}, \ldots, [\![v]\!]^{(\tau)}$) and compares them. If $([\![w]\!]^{(i)}, [\![v]\!]^{(i)})$ and $([\![w]\!]^{(j)}, [\![v]\!]^{(j)})$ encode different values, then their digests $[\![\alpha']\!]^{(i)}$ and $[\![\alpha']\!]^{(j)}$ will differ with high probability. In practice, the parameters ρ and B are chosen such that the probability that two different plain values lead to the same digest is negligible. We further note that taking $(H_1|H_2)$ uniformly at random gives the smallest probability but requires to perform matrix-vector multiplications. Other strategies are possible for $(H_1|H_2)$ such as relying on a polynomial-based hash: this increases a bit the collision probability (so one needs to increase B to compensate) but lightens the computation. This strategy is used in the FAEST signature scheme [13].

We use the VOLEitH framework with the same MPC protocol than with the TCitH framework, namely the MPC protocol Π_{PC} described in Protocol 1, which is equivalent to the QuickSilver VOLE-based protocol [39] in the VOLE setting.

The publicly-known degree-1 sharing $[\![0]\!]$ in Protocol 1 when $e = \infty$ can be built as $[\![0]\!]_i = 1$ for all i.

To achieve a PoK with λ-bit security (i.e. $2^{-\lambda}$ soundness error), we take the field extension $\mathbb{K}$ of degree ρ such that $\frac{1}{|\mathbb{F}|^\rho} \leq 2^{-\lambda}$, the number τ of sharings $[\![w]\!]^{(j)}$ such that $\frac{d}{N^\tau} \leq 2^{-\lambda}$ and the additional parameter B such[6] that $B \cdot \log_2 |\mathbb{F}| \geq 16$ (the latter choice corresponds to the choice in the specification of FAEST [13]). Then we obtain a signature scheme by applying the Fiat-Shamir transform [23] as previously.

The proof transcript (*i.e.* the signature) includes:

- The opened shares $[\![w]\!]_I$ of the witness $w \in \mathbb{F}^{|w|}$. In practice, one sends the auxiliary values of the sub-sharings $[\![w]\!]^{(1)}, \ldots, [\![w]\!]^{(\tau)}$.
- The opened shares of $[\![v]\!]_I$. When v is uniformly-sampled, the shares are usually communication-free. However, we need τ sub-sharings of the same (uniformly-random) value v. While the first sharing is communication-free, the $\tau - 1$ others require an auxiliary value to ensure that all the sub-sharings encode the same value.
- The degree-d sharing $[\![\alpha]\!]$, for the single MPC execution. Since $[\![\alpha]\!]_I$ can be recomputed by the verifier and since the α should be zero, the prover just needs to send $(d + 1) - 1 - 1 = d - 1$ shares.
- The sibling paths in the GGM trees, together with the unopened seed commitments.
- The opened shares $[\![u]\!]_I$. As for $[\![v]\!]_I$, since all the τ sub-sharings must encode the same random value u, only the first sharing is communication-free and the $\tau - 1$ others require an auxiliary value.
- The degree-1 sharings $[\![\alpha']\!]^{(1)}, \ldots, [\![\alpha']\!]^{(\tau)}$. Since the plaintext value α' is the same for all these sharings and since $[\![\alpha]\!]_I^{(j)}$ can be recomputed by the verifier for all j, sending all these sharings costs only $(\rho + B)$ field elements.

Moreover, the signature includes a 2λ-bit salt and a 2λ-bit commitment digest that correspond to the last verifier challenge (in the Fiat-Shamir heuristic).

[6] As explained previously, the parameter B aims to compensate the security loss due to the use of a polynomial-based hash. Such a hash consists in evaluating in a large domain the polynomial which has the hashed values as coefficients. Thanks to the Schwartz-Zippel lemma, we get that the security loss is of a factor $n + \rho$ (which is the length of the hashed vector). By taking $B \cdot \log_2 |\mathbb{F}| \geq 16$ as in the specification of FAEST, we can securely hash vectors of length at most 2^{16}.

Therefore, the signature size when using the VOLEitH framework is (in bits):

$$\textsc{Size}_{\textsc{VOLEitH}} = $$

$$4\lambda + \tau \cdot \left(\underbrace{|w| \cdot \log_2 |\mathbb{F}|}_{[\![w]\!]_I} + \underbrace{\lambda \cdot \log_2 N}_{\text{GGM tree}} + 2\lambda \right) + \underbrace{(d-1)\rho \cdot \log_2 |\mathbb{F}|}_{[\![\alpha]\!]}$$

$$+ (\tau - 1) \cdot \left(\underbrace{(d-1)\rho \cdot \log_2 |\mathbb{F}|}_{[\![v]\!]_I} + \underbrace{(\rho + B) \log_2 |\mathbb{F}|}_{[\![u]\!]_I} \right) + \underbrace{(\rho + B) \cdot \log_2 |\mathbb{F}|}_{[\![\alpha']\!]} .$$

5.3 Additional MPCitH Optimisations

New generic optimizations for MPCitH-based schemes relying on GGM trees have been proposed in a recent work [11]. The improvements are threefold:

1. Instead of considering τ independent GGM trees of N leaves in parallel, the authors propose to rely on a unique large GGM tree of $\tau \cdot N$ leaves where the i^{th} share of the e^{th} PoK repetition is associated to the $(e \cdot N + i)^{\text{th}}$ leaf of the large GGM tree. As explained in [11], "opening all but τ leaves of the big tree is more efficient than opening all but one leaf in each of the τ smaller trees, because with high probability some of the active paths in the tree will merge relatively close to the leaves, which reduces the number of internal nodes that need to be revealed."
2. The authors further propose to improve the previous approach using the principle of *grinding*. When the last Fiat-Shamir challenge is such that the number of revealed nodes in the revealed sibling paths exceed a threshold T_{open}, the signer rejects the challenge and recompute the hash with an incremented counter. This process is done until the number of revealed nodes is $\leq T_{\text{open}}$. For example, if we consider $N = 256$ and $\tau = 16$, the number of revealed nodes is smaller than (or equal to) $T_{\text{open}} := 110$ with probability ≈ 0.2. The selected value of T_{open} induces a rejection probability $p_{\text{rej}} = 1 - 1/\theta$, for some $\theta \in (0, \infty)$, and the signer hence needs to perform an average of θ hash computations for the challenge (instead of 1). While this strategy decreases the challenge space by a factor θ, it does not change the average number of hashes that must be computed to succeed an attack (since the latter is multiplied by θ). As noticed by the authors of [11], this strategy can be thought of as loosing $\log_2 \theta$ bit of security (because of a smaller challenge space) which are regained thanks to a proof-of-work (performing an average of θ hash computations before getting a valid challenge).
3. Finally, [11] proposes to add another explicit proof-of-work to the Fiat-Shamir hash computation of the last challenge. The signer must get a hash digest for which the w last bits are zero, for w a parameter of the scheme. The same counter as for the previous improvement is used as a nonce in this hash and increased until the w-zeros property is satisfied. This strategy

increases the cost of hashing the last challenge by a factor 2^w and hence increases the security of w bits. This thus allows to take smaller parameters (N, τ) for the large tree, namely parameters achieving $\lambda - w$ bits of security instead of λ.

While the authors of [11] focus on VOLEitH, the same optimisations also apply to TCitH. In summary, for a given w, one picks parameters (N, τ) ensuring $\lambda - w$ bits of security. Then fixing T_{open} for these (N, τ) yields a rejection probability $p_{\text{rej}} = 1 - 1/\theta$. The gain in size comes from the smaller parameters (N, τ) on the one hand, and the smaller sibling paths (of size $\leq T_{\text{open}}$ instead of $\approx \tau \log_2 N$) on the other hand. This gain in size is traded for an increased number of Fiat-Shamir hash attempts ($\theta \cdot 2^w$ on average instead of 1).

6 New Signatures Based on $\mathsf{RSD_s}$ and $\mathsf{MinRank}$

In this section, we propose new signature schemes based on the rank syndrome decoding problem and on the MinRank problem. To proceed, we rely on the TCitH and VOLEitH frameworks to obtain non-interactive zero-knowledge proofs of knowledge for these two problems using the new Dual Support Decomposition model described in Section 4 and we use the recent MPCitH optimisation presented in Section 5.3. Moreover, to have more granularity in the choice of the parameters, we consider that the τ emulations of the MPC protocol might not involve the same number of parties: there will be τ_1 emulations with N_1 parties and $\tau_2 := \tau - \tau_1$ emulations with N_2 parties. The schemes are then secure if $\left(\frac{d}{N_1}\right)^{\tau_1} \cdot \left(\frac{d}{N_2}\right)^{\tau_2}$ for TCitH and $\frac{d}{N_1^{\tau_1} \cdot N_2^{\tau_2}}$ for VOLEitH are negligible (instead of simply $\left(\frac{d}{N}\right)^{\tau}$ and $\frac{d}{N^{\tau}}$).

6.1 New Signatures Based on $\mathsf{RSD_s}$

The TCitH and VOLEitH frameworks enable us to prove the knowledge of a witness that satisfies some polynomial constraints. In order to get a signature scheme based on the rank syndrome decoding problem, one just needs to exhibit the polynomial constraints which is satisfied by a rank syndrome decoding solution. As shown in Section 4.1, solving an $\mathsf{RSD_s}$ instance for $\boldsymbol{y}$ and $\boldsymbol{H}$ is equivalent to finding $\boldsymbol{s} = (x_2, \ldots, x_r)$ where $x_i \in \mathbb{F}_{q^m}$ for $i \in \{2, \ldots, r\}$ and $\boldsymbol{C} \in \mathbb{F}_q^{r \times (n-r)}$ such that

$$\boldsymbol{x}\boldsymbol{H}^T - \boldsymbol{y} = \boldsymbol{0} \quad \text{with} \quad \boldsymbol{x} := \begin{pmatrix} 1 & \boldsymbol{s} \end{pmatrix} \cdot \begin{pmatrix} \boldsymbol{I_r} & \boldsymbol{C} \end{pmatrix} \in \mathbb{F}_{q^m}^n \tag{1}$$

Equation 1 directly gives degree-2 polynomial constraints into the coefficients of $\boldsymbol{s}$ and $\boldsymbol{C}$. Let us assume that the $\boldsymbol{H}$ is in standard form, meaning it can be written as $\boldsymbol{H} = \begin{pmatrix} \boldsymbol{I_{n-k}} & \boldsymbol{H'} \end{pmatrix}$, where $\boldsymbol{H'} \in \mathbb{F}_{q^m}^{(n-k) \times k}$. Given the inputs $[\![\boldsymbol{s}]\!]$ and $[\![\boldsymbol{C}]\!]$, the hint $[\![\boldsymbol{v}]\!]$ with $\boldsymbol{v} \in \mathbb{F}_{q^m}^{\rho}$ and the MPC randomness $\boldsymbol{\Gamma} = (\gamma_{i,j})_{i,j} \in \mathbb{F}_{q^m}^{(n-k) \times \rho}$, the emulated MPC protocol (repeated ρ times) described in Protocol 1 thus consists of computing

$$[\![\boldsymbol{\alpha}]\!] \leftarrow [\![\boldsymbol{v}]\!] \cdot [\![0]\!] + ([\![\boldsymbol{x_A}]\!] + [\![\boldsymbol{x_B}]\!]\boldsymbol{H'}^T - \boldsymbol{y})\boldsymbol{\Gamma}$$

where $[\![\boldsymbol{x_A}]\!]$ and $[\![\boldsymbol{x_B}]\!]$ are built such as $[\![\begin{pmatrix} \boldsymbol{x_A} & \boldsymbol{x_B} \end{pmatrix}]\!] = \begin{pmatrix} 1 & [\![\boldsymbol{s}]\!] \end{pmatrix} \cdot \begin{pmatrix} \boldsymbol{I_r} & [\![\boldsymbol{C}]\!] \end{pmatrix}$.

Signature size. According to Section 5, the signature size using the TCitH framework is (in bits):

$$\textsc{Size}_{\text{TCitH}} = 4\lambda + \underbrace{\lambda \cdot T_{\text{open}}}_{\text{GGM tree}} + \tau \cdot \left(\underbrace{[(r-1)m + r(n-r)] \cdot \log_2 q}_{[\![s]\!]_I, [\![C]\!]_I} \right. \\ \left. + \underbrace{(d-1) \cdot \rho \cdot \log_2 q}_{[\![\alpha]\!]} + 2\lambda \right),$$

while the signature size using the VOLEitH framework is (in bits):

$$\textsc{Size}_{\text{VOLEitH}} = 4\lambda + \underbrace{\lambda \cdot T_{\text{open}}}_{\text{GGM tree}} \\ + \tau \cdot \left(\underbrace{[(r-1)m + r(n-r)] \cdot \log_2 q}_{[\![s]\!]_I, [\![C]\!]_I} + 2\lambda \right) + \underbrace{(d-1) \cdot \rho \cdot \log_2 q}_{[\![\alpha]\!]} \\ + (\tau - 1) \cdot \left(\underbrace{\rho \cdot \log_2 q}_{[\![v]\!]_I} + \underbrace{(\rho + B) \log_2 q}_{[\![u]\!]_I} \right) + \underbrace{(\rho + B) \cdot \log_2 q}_{[\![\alpha']\!]} .$$

Computational cost. The running time of the signing algorithm can be split in three main parts:

1. The generation of the input shares using seed trees and their commitment. The computational cost scales linearly with the number of input shares. When there are τ_1 MPC emulations with N_1 parties and τ_2 MPC emulations with N_2 parties, the total number of input shares is $\tau_1 \cdot N_1 + \tau_2 + N_2$.
2. The MPC emulation. This step consists in computing the degree-2 broadcast sharing $[\![\boldsymbol{\alpha}]\!]$, knowing that $\boldsymbol{\alpha} = 0$. Let us estimate the cost of emulating the MPC protocol. We only count multiplications which are predominant (compared to additions) for the considered extension fields. We recall that multiplying two degree-1 sharings costs 2 multiplications in the underlying field, assuming we already know the plain value.
   - With TCitH, the MPC emulation will be repeated $\tau := \tau_1 + \tau_2$ times. Each repetition includes 2 vector-matrix multiplications with a matrix $\mathbb{F}_{q^m}^{(r-1)\times(n-r)}$ to compute $[\![\boldsymbol{x}]\!] := [\![(\boldsymbol{x_A}\ \boldsymbol{x_B})]\!]$, 2 vector-matrix multiplications with a matrix of $\mathbb{F}_{q^m}^{k\times(n-k)}$ to compute $[\![\boldsymbol{r}]\!] := [\![\boldsymbol{x_A}]\!] + [\![\boldsymbol{x_B}]\!]\boldsymbol{H'}^T - \boldsymbol{y}$, and 2 vector-matrix multiplications with matrix of $\mathbb{F}_{q^m}^{(n-k)\times\rho}$ to compute $[\![\boldsymbol{\alpha}]\!]$.
   - With VOLEitH, the MPC emulation is executed only once, but in a larger extension field $\mathbb{K}$ where $[\mathbb{K} : \mathbb{F}_{q^m}] = \rho$. The emulation includes 2 vector-matrix multiplications with a matrix $\mathbb{K}^{(r-1)\times(n-r)}$ to compute

$[\![\boldsymbol{x}]\!] := [\![(\boldsymbol{x_A}\ \boldsymbol{x_B})]\!]$, 2ρ vector-matrix multiplications with a matrix of $\mathbb{F}_{q^m}^{k\times(n-k)}$ to compute $[\![\boldsymbol{r}]\!] := [\![\boldsymbol{x_A}]\!] + [\![\boldsymbol{x_B}]\!]\boldsymbol{H'}^T - \boldsymbol{y}$, and 2 vector-matrix multiplications with matrix of $\mathbb{K}^{(n-k)\times 1}$ to compute $[\![\boldsymbol{\alpha}]\!]$.

3. The global proof-of-work, composed of the grinding process on the seed trees and the explicit proof-of-work on the Fiat-Shamir hash computation. Its average cost is $\theta \cdot 2^w$ Fiat-Shamir hash computations.

The running time of the other parts of the signing algorithm is negligible compared to those three components. Regarding the running time of the verification algorithm, since the verifier should also expand the seed trees and emulate some parties, the verification time will be similar (a bit smaller) than the signing time.

Parameter selection. We select some parameter sets for our signature schemes. To have a fair comparison between both frameworks (TCitH and VOLEitH), we chose the parameters such that the cost of generating the input shares and the cost of the proof-of-work are similar (namely, we chose parameters such that $\tau_2 \cdot N_1 + \tau_2 \cdot \tau_2$ and $\theta \cdot 2^w$ are roughly equal). We present in Table 7 the sizes obtained for the signature scheme.

While proposing optimized implementations of our signature scheme is left for future work, we provide some (upper bound) estimates for its running time in Table 8. The timings of the symmetric components (generation and commitment of the input shares and proof of work) are estimated based on the benchmarks from [11]. Since we rely on the same parameters for the symmetric components (same $\tau_1 \cdot N_1 + \tau_2 \cdot N_2$ and same $\log_2 \theta + w$), we can use their timings as upper bounds. For example, their scheme MandaRain-3-128s includes a generation and commitment of 22 528 input shares and has a total proof-of-work of 14 bits as our "short" instances. Since it runs in 2.8 ms on a 5 GHz CPU, we deduce that the symmetric components cost is *at most* 14 Mcycles.[7] Then, we derived and benchmarked a naive implementation of the MPC emulation, which gives us an upper bound for the emulation cost. Despite this pessimistic estimation, the results presented in Table 8 show that our scheme is competitive with the NIST candidate RYDE. In particular, all our variants relying on VOLEitH are faster than RYDE.

Comparison. Table 9 summarizes the state of the art of signature schemes based on RSD. We include in the comparison only short parameters, i.e, with $N = 256$ for MPCitH-based signatures, and $N = 32$ for [15]. We include the schemes of Stern [37] and Véron [38] applied to the rank metric. For 128 bits of security, these two schemes have signature sizes of around 30 kB. These sizes were roughly halved in [20] and [15]. Finally, [19] reduced it below 6 kB and our work achieves sizes below 4 kB.

[7] In making this consideration, we include the overhead of emulating their MPC protocol to our estimates of the symmetric part.

Security	Trade-off	Framework	Scheme Parameters				Computational Cost			Signature
			τ	(τ_1, N_1)	(τ_2, N_2)	T_{open}	#Leaves	$\log_2 \theta$	w	
NIST I	Short	TCitH	12	$(10, 2^{11})$	$(2, 2^{10})$	111	22528	5.0	9	2 937 B
		VOLEitH	11	$(0, 2^{12})$	$(11, 2^{11})$	99	22528	7.2	7	2 851 B
	Fast	TCitH	20	$(4, 2^{8})$	$(16, 2^{7})$	113	3072	7.1	3	3 708 B
		VOLEitH	16	$(8, 2^{8})$	$(8, 2^{7})$	102	3072	2.9	8	3 450 B
NIST III	Short	TCitH	18	$(2, 2^{12})$	$(16, 2^{11})$	174	40960	4.9	9	6 713 B
		VOLEitH	16	$(4, 2^{12})$	$(12, 2^{11})$	162	40960	2.7	12	6 566 B
	Fast	TCitH	30	$(10, 2^{8})$	$(20, 2^{7})$	178	5120	6.9	1	8 454 B
		VOLEitH	24	$(16, 2^{8})$	$(8, 2^{7})$	176	5120	0.0	8	8 207 B
NIST V	Short	TCitH	25	$(5, 2^{12})$	$(20, 2^{11})$	245	61440	5.6	0	12 371 B
		VOLEitH	22	$(8, 2^{12})$	$(14, 2^{11})$	248	61440	0.0	6	12 682 B
	Fast	TCitH	39	$(17, 2^{8})$	$(22, 2^{7})$	247	7168	3.7	4	14 926 B
		VOLEitH	32	$(24, 2^{8})$	$(8, 2^{7})$	247	7168	0.0	8	14 768 B

Table 7: Parameters and resulting sizes for the new signature scheme based on $\mathsf{RSD_s}$. The used parameters for the rank syndrome decoding problem are those of Table 3.

Resilience Property. One should note that our scheme is highly resilient to hypothetical cryptanalytic progress on $\mathsf{RSD_s}$. Indeed, if we were to take the set of parameters for $\mathsf{RSD_s}$ corresponding to NIST III, applied to the proof of knowledge for NIST I, i.e, a security of $\lambda = 192$ for $\mathsf{RSD_s}$ and $\lambda = 128$ for the protocol, we would get an increase of only 0.4 kB (for $N = 512$) or 0.3 kB (for $N = 2048$) in the signature size. Namely, we can take a large margin of security for the parameters of $\mathsf{RSD_s}$ at a moderate cost.

6.2 New Signatures Based on MinRank

The TCitH and VOLEitH frameworks enable us to prove the knowledge of a witness that satisfies some polynomial constraints. In order to get a signature scheme based on $\mathsf{MinRank}$, one just needs to exhibit the polynomial constraints which that a $\mathsf{MinRank}$ solution should satisfy. As shown in Section 4.2, we can use the $\mathsf{MinRank\ Syndrome}$ problem, which asks to find $\boldsymbol{S'} \in \mathbb{F}_q^{(m-r)\times r}$ and $\boldsymbol{C} \in \mathbb{F}_q^{r\times n}$ such that

$$\rho(\boldsymbol{E}) \cdot \boldsymbol{H}^T - \boldsymbol{y} = \boldsymbol{0} \quad \text{with} \quad \boldsymbol{E} := \begin{pmatrix} \boldsymbol{I_r} \\ \boldsymbol{S'} \end{pmatrix} \cdot \boldsymbol{C}, \tag{2}$$

where $\boldsymbol{H} \in \mathbb{F}_q^{(mn-k)\times mn}$ and $\boldsymbol{y} \in \mathbb{F}_q^{mn-k}$. Equation (2) directly gives degree-2 polynomial constraints into the coefficients of $\boldsymbol{S'}$ and $\boldsymbol{C}$. Let us assume that the matrix $\boldsymbol{H}$ is in standard form, meaning it can be written as $\boldsymbol{H} = \left(\boldsymbol{I_{n\cdot m-k}}\ \boldsymbol{H'}\right)$, where $\boldsymbol{H'} \in \mathbb{F}_q^{(n\cdot m-k)\times k}$. Given the inputs $[\![\boldsymbol{S'}]\!]$ and $[\![\boldsymbol{C}]\!]$, the hint $[\![\boldsymbol{v}]\!]$ with $\boldsymbol{v} \in \mathbb{F}_q^{\rho}$ and the MPC randomness $\boldsymbol{\Gamma} = (\gamma_{i,j})_{i,j} \in \mathbb{F}_q^{(n-k)\times\rho}$, the emulated MPC protocol

Security	Trade-off	Framework	Symmetric Part	MPC Emulation			Total	RYDE
			From [11]	$\llbracket \boldsymbol{x} \rrbracket$	$\llbracket \boldsymbol{r} \rrbracket$	$\llbracket \boldsymbol{\alpha} \rrbracket$		
NIST I	Short	TCitH	14	0.38	1.42	0.19	16.0	23.4
		VOLEitH	14	0.43	0.36	0.07	14.9	
	Fast	TCitH	1.8	0.62	2.36	0.24	5.0	5.4
		VOLEitH	1.8	0.43	0.36	0.07	2.7	
NIST III	Short	TCitH	37	3.9	12.8	0.6	54.3	49.6
		VOLEitH	37	1.3	2.1	0.2	40.6	
	Fast	TCitH	4.4	6.5	21.3	1.1	33.3	12.2
		VOLEitH	4.4	1.3	2.1	0.2	8.0	
NIST V	Short	TCitH	45	9.5	24.4	0.9	79.8	94.9
		VOLEitH	45	2.0	2.9	0.2	50.1	
	Fast	TCitH	6.8	14.8	37.8	1.4	60.8	22.7
		VOLEitH	6.8	1.9	2.9	0.17	11.8	

Table 8: Estimation of the signing times of the new signature scheme based on $\mathsf{RSD_s}$ (in mega-cycles).

RSD Parameters	Scheme	N	M	τ	η	ρ	Signature Size
$q=2$ $m=31$ $n=33$ $k=15$ $r=10$	[37]	-	-	219	-	-	33 886 B
	[38]	-	-	219	-	-	28 794 B
	[20]	32	389	28	-	-	14 792 B
	[15]	32	389	28	-	-	12 816 B
	[19] RD	256	-	21	24	-	8 990 B
	[19] LP and [4] ($\mathsf{RSD_s}$)	256	-	20	1	-	5 956 B
$q=2, m=53, n=53$	Our scheme (TCitH)	256	-	20	-	3	3 708 B
$k=45, r=4$	Our scheme (VOLEitH)	256	-	16	-	128	3 450 B

Table 9: Comparison of the signatures relying on RSD, restricting to the schemes using the Fiat-Shamir transform.

(repeated ρ times) described in Protocol 1 thus consists in computing

$$\llbracket \boldsymbol{\alpha} \rrbracket \leftarrow \llbracket \boldsymbol{v} \rrbracket \cdot \llbracket 0 \rrbracket + (\llbracket \boldsymbol{x_A} \rrbracket + \llbracket \boldsymbol{x_B} \rrbracket \boldsymbol{H'}^T - \boldsymbol{y})\boldsymbol{\Gamma}$$

where $\llbracket \boldsymbol{x_A} \rrbracket$ and $\llbracket \boldsymbol{x_B} \rrbracket$ are built such as $\llbracket (\boldsymbol{x_A}\ \boldsymbol{x_B}) \rrbracket = \rho\left(\begin{pmatrix} \boldsymbol{I_r} \\ \llbracket \boldsymbol{S'} \rrbracket \end{pmatrix} \cdot \llbracket \boldsymbol{C} \rrbracket\right)$.

Signature size. According to Section 5, the signature size using the TCitH framework is (in bits):

$$\textsc{Size}_{\textsc{TCitH}} = 4\lambda + \underbrace{\lambda \cdot T_{\text{open}}}_{\text{GGM tree}} + \tau \cdot \left(\underbrace{[r(m-r)+rn] \cdot \log_2 q}_{\llbracket \boldsymbol{S'} \rrbracket_I, \llbracket \boldsymbol{C} \rrbracket_I} + \underbrace{(d-1) \cdot \rho \cdot \log_2 q}_{\llbracket \alpha \rrbracket} + 2\lambda \right),$$

while the signature size using the VOLEitH framework is (in bits):

$$\begin{aligned} \textsc{Size}_{\textsc{VOLEitH}} = {} & 4\lambda + \underbrace{\lambda \cdot T_{\text{open}}}_{\text{GGM tree}} \\ & + \tau \cdot \left(\underbrace{[r(m-r)+rn] \cdot \log_2 q}_{\llbracket \boldsymbol{S'} \rrbracket_I, \llbracket \boldsymbol{C} \rrbracket_I} + 2\lambda \right) + \underbrace{(d-1) \cdot \rho \cdot \log_2 q}_{\llbracket \alpha \rrbracket} \\ & + (\tau - 1) \cdot \left(\underbrace{\rho \cdot \log_2 q}_{\llbracket v \rrbracket_I} + \underbrace{(\rho + B) \log_2 q}_{\llbracket u \rrbracket_I} \right) + \underbrace{(\rho + B) \cdot \log_2 q}_{\llbracket \alpha' \rrbracket} . \end{aligned}$$

Computational cost. As in the previous section, the running time of the signing algorithm can be split in three main parts:

1. The generation of the input share using seed trees and their commitment. The computational cost scales linearly with the number of input shares. When there are τ_1 MPC emulations with N_1 parties and τ_2 MPC emulations with N_2 parties, the total number of input shares is $\tau_1 \cdot N_1 + \tau_2 + N_2$.
2. The MPC emulation. This step consists in computing the degree-2 broadcast sharing $\llbracket \boldsymbol{\alpha} \rrbracket$, knowing that $\boldsymbol{\alpha} = 0$. Let us estimate the cost of emulating the MPC protocol (while only counting multiplications as above).
   - With TCitH, the MPC emulation will be repeated $\tau := \tau_1 + \tau_2$ times. Each repetition includes 2 multiplications between matrices of $\mathbb{F}_N^{(m-r)\times r}$ and $\mathbb{F}_N^{r \times n}$ to compute $\llbracket \boldsymbol{x} \rrbracket$, $2 \cdot [\mathbb{F}_N : \mathbb{F}_q]$ vector-matrix multiplications with a matrix of $\mathbb{F}_q^{k \times (n \cdot m - k)}$ to compute $\llbracket \boldsymbol{x_A} \rrbracket + \llbracket \boldsymbol{x_B} \rrbracket \boldsymbol{H'}^T - \boldsymbol{y}$, and 2 vector-matrix multiplications with matrix of $\mathbb{F}_N^{(n \cdot m - k) \times \rho}$ to compute $\llbracket \boldsymbol{\alpha} \rrbracket$.
   - With VOLEitH, the MPC emulation is executed only once, but in a larger extension field $\mathbb{K}$ where $[\mathbb{K} : \mathbb{F}_N] = \rho$. The emulation includes 2 matrix multiplications of $\mathbb{K}^{(m-r)\times r}$ and $\mathbb{K}^{r \times n}$ to compute $\llbracket \boldsymbol{x} \rrbracket$, $2\rho \cdot [\mathbb{F}_N : \mathbb{F}_q]$ vector-matrix multiplications with a matrix of $\mathbb{F}_q^{k \times (n \cdot m - k)}$ to compute $\llbracket \boldsymbol{x_A} \rrbracket + \llbracket \boldsymbol{x_B} \rrbracket \boldsymbol{H'}^T - \boldsymbol{y}$, and 2 vector-matrix multiplications with matrix of $\mathbb{K}^{(n \cdot m - k) \times 1}$ to compute $\llbracket \boldsymbol{\alpha} \rrbracket$.

3. The global proof-of-work, composed of the grinding process on the seed trees and the explicit proof-of-work on the Fiat-Shamir hash computation. Its average cost is $\theta \cdot 2^w$ Fiat-Shamir hash computations.

The running time of the other parts of the signing algorithm is negligible compared to those three components. Regarding the running time of the verification algorithm, since the verifier should also expand the seed trees and emulate some parties, the verification time will be similar (a bit smaller) than the signing time.

Parameter selection. We select some parameter sets for our signature schemes. To have a fair comparison between both frameworks (TCitH and VOLEitH), we chose the parameters such that the cost of generating the input shares and the cost of the proof-of-work are similar (namely, we chose parameters such that $\tau_2 \cdot N_1 + \tau_2 \cdot \tau_2$ and $\theta \cdot 2^w$ are roughly equal). We present in Table 10 the sizes obtained for the signature scheme.

As previously, we leave optimized implementations for future work and provide (upper bound) estimates of the running time in Table 11 based on the benchmarks from [11] and a naive implementation of the MPC emulation of our scheme. Despite this pessimistic estimation, the results of Table 11 show that our scheme is competitive with the NIST sublmissions MIRA and MiRitH (both applying MPC-in-the-Head to MinRank). In particular, all our variants relying on TCitH are faster than MIRA and the short instances of MiRitH.

Security	Trade-off	Framework	Scheme Parameters				Computational Cost			Signature
			τ	(τ_1, N_1)	(τ_2, N_2)	T_{open}	#Leaves	$\log_2 \theta$	w	
NIST I	Short	TCitH	12	$(10, 2^{11})$	$(2, 2^{10})$	111	22528	5.0	9	2 896 B
		VOLEitH	11	$(0, 2^{12})$	$(11, 2^{11})$	99	22528	7.0	7	2 813 B
	Fast	TCitH	20	$(4, 2^8)$	$(16, 2^7)$	113	3072	7.0	3	3 640 B
		VOLEitH	16	$(8, 2^8)$	$(8, 2^7)$	102	3072	2.8	8	3 396 B
NIST III	Short	TCitH	18	$(2, 2^{12})$	$(16, 2^{11})$	174	40960	5.0	9	6 584 B
		VOLEitH	16	$(4, 2^{12})$	$(12, 2^{11})$	162	40960	2.7	12	6 452 B
	Fast	TCitH	30	$(10, 2^8)$	$(20, 2^7)$	178	5120	6.9	1	8 240 B
		VOLEitH	24	$(16, 2^8)$	$(8, 2^7)$	176	5120	0.0	8	8 036 B
NIST V	Short	TCitH	25	$(5, 2^{12})$	$(20, 2^{11})$	245	61440	5.6	0	12 149 B
		VOLEitH	22	$(8, 2^{12})$	$(14, 2^{11})$	248	61440	0.0	6	12 486 B
	Fast	TCitH	39	$(17, 2^8)$	$(22, 2^7)$	247	7168	3.8	4	14 579 B
		VOLEitH	32	$(24, 2^8)$	$(8, 2^7)$	247	7168	0.0	8	14 484 B

Table 10: Parameters and resulting sizes for the new signature scheme based on MinRank. The used parameters for the MinRank problem are those of Table 4.

Comparison. Table 12 summarizes the state of the art of signature schemes based on MinRank. We include in the comparison only short parameters, i.e, with

Security	Trade-off	Framework	Symmetric Part	MPC Emulation			Total	MIRA	MiRitH
			From [11]	$[\![\boldsymbol{x}]\!]$	$[\![\boldsymbol{r}]\!]$	$[\![\boldsymbol{\alpha}]\!]$			
NIST I	Short	TCitH	14	12.6	4.6	4.5	35.7	46.8	76.5
		VOLEitH	14	54.8	1.4	2.7	72.9		
	Fast	TCitH	1.8	3.7	5.1	1.9	12.5	37.4	8.7
		VOLEitH	1.8	54.8	1.4	2.7	60.7		
NIST III	Short	TCitH	37	37.6	22.0	14.4	111.0	119.7	192.9
		VOLEitH	37	217.8	8.2	7.5	270.5		
	Fast	TCitH	4.4	9.8	23.4	5.2	42.8	107.2	22.5
		VOLEitH	4.4	217.8	8.2	7.5	237.9		
NIST V	Short	TCitH	45	82.4	60.2	33.3	220.9	337.7	308.6
		VOLEitH	45	695.2	3.9	19.1	763.2		
	Fast	TCitH	6.8	15.2	61.7	9.7	93.4	322.3	36.4
		VOLEitH	6.8	694.4	14.6	19.1	734.9		

Table 11: Estimation of the running times of the new signature scheme based on MinRank (in mega-cycles).

$N = 256$ for MPCitH-based signatures, and $N = 32$ for [15]. For the MinRank parameters, we use $q = 16, m = 16, n = 16, k = 142, r = 4$. Historically, the first schemes from [17], [35], and [14] obtained signature sizes no less than 26 kB for 128 bits of security. Then, the technique from [15] applied to MinRank achieved ∼10 kB, and [2] reduced it even below 7 kB. The recent work from [19] reduces it below 6 kB, and the MIRA and MiRitH submissions to the NIST have sizes below 6 kB as well. Finally, our work achieves sizes below 4 kB.

MinRank Parameters	Scheme	N	M	τ	η	ρ	Signature Size
$q = 16$ $m = 16$ $n = 16$ $k = 142$ $r = 4$	[17]	-	-	219	-	-	28 575 B
	[35]	-	-	128	-	-	28 128 B
	[14]	-	256	128	-	-	26 405 B
	[15]	32	389	28	-	-	10 937 B
	[2]	256	-	18	-	-	7 422 B
	[19] RD	256	-	19	9	-	7 122 B
$q = 16, m = 16, n = 16$ $k = 120, r = 5$	[19] LP and MIRA [5]	256	-	18	1	-	5 640 B
$q = 16, m = 15, n = 15$ $k = 78, r = 6$	MiRitH [1]	256	-	19	9	-	5 673 B
$q = 2, m = 43, n = 43$ $k = 1520, r = 4$	Our scheme (TCitH)	256	-	20	-	130	3 640 B
	Our scheme (VOLEitH)	256	-	16	-	128	3 396 B

Table 12: Comparison of the signatures relying on MinRank, restricting to the schemes using the Fiat-Shamir transform.

Resilience Property. As for our scheme based on RSD_s, our above scheme is highly resilient to hypothetical cryptanalytic progress on MinRank. Indeed, if we were to take the set of parameters for MinRank corresponding to NIST III, applied to the proof of knowledge for NIST I, i.e, a security of $\lambda = 192$ for MinRank and $\lambda = 128$ for the protocol, we would get an increase of only 0.4 kB (for $N = 512$) or 0.3 kB (for $N = 2048$) in the signature size. Namely, we can take a large margin of security for the parameters of MinRank at a moderate cost.

References

1. Gora Adj, Stefano Barbero, Emanuele Bellini, Andre Esser, Luis Rivera-Zamarripa, Carlo Sanna, Javier Verbel, and Floyd Zweydinger. MiRitH. NIST's Post-Quantum Cryptography Standardization of Additional Digital Signature Schemes Project (Round 1), https://pqc-mirith.org/, 2023.
2. Gora Adj, Luis Rivera-Zamarripa, and Javier Verbel. Minrank in the head. In Nadia El Mrabet, Luca De Feo, and Sylvain Duquesne, editors, *Progress in Cryptology - AFRICACRYPT 2023*, pages 3–27, Cham, 2023. Springer Nature Switzerland.
3. Carlos Aguilar Melchor, Nicolas Gama, James Howe, Andreas Hülsing, David Joseph, and Dongze Yue. The Return of the SDitH. In Carmit Hazay and Martijn Stam, editors, *Advances in Cryptology - EUROCRYPT 2023 - 42nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Lyon, France, April 23-27, 2023, Proceedings, Part V*, volume 14008 of *Lecture Notes in Computer Science*, pages 564–596. Springer, 2023.
4. Nicolas Aragon, Magali Bardet, Loïc Bidoux, Jesús-Javier Chi-Domínguez, Victor Dyseryn, Thibauld Feneuil, Philippe Gaborit, Antoine Joux, Matthieu Rivain, Jean-Pierre Tillich, and Adrien Vincotte. RYDE. NIST's Post-Quantum Cryptography Standardization of Additional Digital Signature Schemes Project (Round 1), https://pqc-ryde.org/, 2023.
5. Nicolas Aragon, Magali Bardet, Loïc Bidoux, Jesús-Javier Chi-Domínguez, Victor Dyseryn, Thibauld Feneuil, Philippe Gaborit, Romaric Neveu, Matthieu Rivain, and Jean-Pierre Tillich. MIRA. NIST's Post-Quantum Cryptography Standardization of Additional Digital Signature Schemes Project (Round 1), https://pqc-mira.org/, 2023.
6. Nicolas Aragon, Philippe Gaborit, Adrien Hauteville, and Jean-Pierre Tillich. A New Algorithm for Solving the Rank Syndrome Decoding Problem. In *2018 IEEE International Symposium on Information Theory (ISIT)*, pages 2421–2425, 2018.
7. Magali Bardet, Pierre Briaud, Maxime Bros, Philippe Gaborit, Vincent Neiger, Olivier Ruatta, and Jean-Pierre Tillich. An Algebraic Attack on Rank Metric Code-Based Cryptosystems. In Anne Canteaut and Yuval Ishai, editors, *Advances in Cryptology – EUROCRYPT 2020*, pages 64–93, Cham, 2020. Springer International Publishing.
8. Magali Bardet, Pierre Briaud, Maxime Bros, Philippe Gaborit, and Jean-Pierre Tillich. Revisiting algebraic attacks on MinRank and on the rank decoding problem. *Designs, Codes and Cryptography*, 91:3671–3707, 2023.
9. Magali Bardet, Maxime Bros, Daniel Cabarcas, Philippe Gaborit, Ray Perlner, Daniel Smith-Tone, Jean-Pierre Tillich, and Javier Verbel. Improvements of Algebraic Attacks for Solving the Rank Decoding and MinRank Problems. In Shiho

Moriai and Huaxiong Wang, editors, *Advances in Cryptology – ASIACRYPT 2020*, pages 507–536, Cham, 2020. Springer International Publishing.

10. Magali Bardet, Maxime Bros, Daniel Cabarcas, Philippe Gaborit, Ray Perlner, Daniel Smith-Tone, Jean-Pierre Tillich, and Javier Verbel. Improvements of Algebraic Attacks for Solving the Rank Decoding and MinRank Problems. In *Advances in Cryptology – ASIACRYPT 2020*, pages 507–536. Springer International Publishing, 2020.
11. Carsten Baum, Ward Beullens, Shibam Mukherjee, Emmanuela Orsini, Sebastian Ramacher, Christian Rechberger, Lawrence Roy, and Peter Scholl. One tree to rule them all: Optimizing ggm trees and owfs for post-quantum signatures. Cryptology ePrint Archive, Paper 2024/490, 2024. https://eprint.iacr.org/2024/490.
12. Carsten Baum, Lennart Braun, Cyprien Delpech de Saint Guilhem, Michael Klooß, Emmanuela Orsini, Lawrence Roy, and Peter Scholl. Publicly verifiable zero-knowledge and post-quantum signatures from vole-in-the-head. In Helena Handschuh and Anna Lysyanskaya, editors, *Advances in Cryptology – CRYPTO 2023*, pages 581–615, Cham, 2023. Springer Nature Switzerland.
13. Carsten Baum, Lennart Braun, Cyprien Delpech de Saint Guilhem, Michael Klooß, Christian Majenz, Shibam Mukherjee, Emmanuela Orsini, Sebastian Ramacher, Christian Rechberger, Lawrence Roy, and Peter Scholl. FAEST. NIST's Post-Quantum Cryptography Standardization of Additional Digital Signature Schemes Project (Round 1), https://faest.info/, 2023.
14. Emanuele Bellini, Andre Esser, Carlo Sanna, and Javier Verbel. Mr-dss – smaller minrank-based (ring-)signatures. In *Post-Quantum Cryptography: 13th International Workshop, PQCrypto 2022, Virtual Event, September 28–30, 2022, Proceedings*, page 144–169, Berlin, Heidelberg, 2022. Springer-Verlag.
15. Loïc Bidoux and Philippe Gaborit. Compact Post-quantum Signatures from Proofs of Knowledge Leveraging Structure for the PKP, SD and RSD Problems. In *Codes, Cryptology and Information Security (C2SI)*, 2023.
16. Nicolas Courtois. La sécurité des primitives cryptographiques basées sur des problèmes algébriques multivariables mq, ip, minrank, hfe, 2001.
17. Nicolas T. Courtois. Efficient zero-knowledge authentication based on a linear algebra problem minrank. In Colin Boyd, editor, *Advances in Cryptology — ASIACRYPT 2001*, pages 402–421, Berlin, Heidelberg, 2001. Springer Berlin Heidelberg.
18. Cyprien Delpech de Saint Guilhem, Emmanuela Orsini, and Titouan Tanguy. Limbo: Efficient Zero-knowledge MPCitH-based Arguments. In Yongdae Kim, Jong Kim, Giovanni Vigna, and Elaine Shi, editors, *CCS '21: 2021 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, Republic of Korea, November 15 - 19, 2021*, pages 3022–3036. ACM, 2021.
19. Thibauld Feneuil. Building MPCitH-based signatures from MQ, MinRank, Rank SD and PKP. In *International Conference on Applied Cryptography and Network Security (ACNS)*, 2024.
20. Thibauld Feneuil, Antoine Joux, and Matthieu Rivain. Shared permutation for syndrome decoding: new zero-knowledge protocol and code-based signature. *Designs, Codes and Cryptography*, 91:563–608, 2022.
21. Thibauld Feneuil and Matthieu Rivain. Threshold Computation in the Head: Improved Framework for Post-Quantum Signatures and Zero-Knowledge Arguments. *Cryptology ePrint Archive, Report 2023/1573*, 2023.
22. Thibauld Feneuil and Matthieu Rivain. Threshold Linear Secret Sharing to the Rescue of MPC-in-the-Head. In *International Conference on the Theory and Application of Cryptology and Information Security (Asiacrypt)*, 2023.

23. Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *Advances in Cryptology — CRYPTO' 86*, pages 186–194, Berlin, Heidelberg, 1987. Springer Berlin Heidelberg.
24. Philippe Gaborit, Olivier Ruatta, and Julien Schrek. On the complexity of the rank syndrome decoding problem. *IEEE Transactions on Information Theory*, 62(2):1006–1019, 2016.
25. Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *J. ACM*, 33(4):792–807, aug 1986.
26. Louis Goubin and Nicolas T. Courtois. Cryptanalysis of the TTM Cryptosystem. In *International Conference on the Theory and Application of Cryptology and Information Security*, 2000.
27. Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Zero-knowledge from secure multiparty computation. In *Proceedings of the Thirty-Ninth Annual ACM Symposium on Theory of Computing*, STOC '07, page 21–30, New York, NY, USA, 2007. Association for Computing Machinery.
28. Jonathan Katz, Vladimir Kolesnikov, and Xiao Wang. Improved Non-Interactive Zero Knowledge with Applications to Post-Quantum Signatures. In David Lie, Mohammad Mannan, Michael Backes, and XiaoFeng Wang, editors, *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS 2018, Toronto, ON, Canada, October 15-19, 2018*, pages 525–537. ACM, 2018.
29. Aviad Kipnis and Adi Shamir. Cryptanalysis of the HFE public key cryptosystem by relinearization. In *crypto '99*, volume 1666 of *LNCS*, pages 19–30, Santa Barbara, California, USA, August 1999. Springer.
30. P. Loidreau. Properties of codes in rank metric, 2006.
31. Carlos Aguilar Melchior, Nicolas Aragon, Slim Bettaieb, Loïc Bidoux, Olivier Blazy, Maxime Bros, Alain Couvreur, Jean-Christophe Deneuville, Philippe Gaborit, Adrien Hauteville, and Gilles Zémor. RQC. NIST's Post-Quantum Cryptography Standardization Process, https://pqc-rqc.org/, 2017.
32. Ralph C. Merkle. A digital signature based on a conventional encryption function. In Carl Pomerance, editor, *Advances in Cryptology — CRYPTO '87*, pages 369–378, Berlin, Heidelberg, 1988. Springer Berlin Heidelberg.
33. NIST. Call for Additional Digital Signature Schemes for the Post-Quantum Cryptography Standardization Process, 2022. https://csrc.nist.gov/csrc/media/Projects/pqc-dig-sig/documents/call-for-proposals-dig-sig-sept-2022.pdf.
34. A. V. Ourivski and T. Johansson. New Technique for Decoding Codes in the Rank Metric and Its Cryptography Applications. *Probl. Inf. Transm.*, 38(3):237–246, jul 2002.
35. Bagus Santoso, Yasuhiko Ikematsu, Shuhei Nakamura, and Takanori Yasuda. Three-pass identification scheme based on minrank problem with half cheating probability, 2022.
36. Adi Shamir. How to share a secret. *Commun. ACM*, 22(11):612–613, nov 1979.
37. Jacques Stern. A new identification scheme based on syndrome decoding. In *International Cryptology Conference (CRYPTO)*, 1993.
38. Pascal Véron. Improved Identification Schemes Based on Error-Correcting Codes. *Applicable Algebra in Engineering, Communication and Computing*, 8(1), January 1997.
39. Kang Yang, Pratik Sarkar, Chenkai Weng, and Xiao Wang. Quicksilver: Efficient and affordable zero-knowledge proofs for circuits and polynomials over any field. In

Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security, CCS '21, page 2986–3001, New York, NY, USA, 2021. Association for Computing Machinery.