SNOVA

Proposal for NIST PQC: Additional Digital Signature Schemes

Version 3.0

Lih-Chung Wang, Chun-Yen Chou, Jintai Ding, Hao Guo,
Yen-Liang Kuan, Jan Adriaan Leegwater,
Ming-Siou Li, Peigen Li, Bo-Shu Tseng, Po-En Tseng, Chia-Chun Wang

August 2026

https://snova.pqclab.org
pqclaborg@gmail.com
lcwang@gms.ndhu.edu.tw, briantseng0320@gmail.com

Changelog (Round 2 → Round 3)

In the following, we describe the changes between the Round 2 submission of SNOVA dated January 25, 2025, and the Round 3 submission, along with brief explanations of the motivations behind the changes.

Additional Submitters

We add two new members to our team: Hao Guo and Peigen Li.

Reformulation of the SNOVA Public Map

We reformulated the SNOVA scheme in Round 3 submission, a decision driven by the cryptanalysis occurred in the Round 2 evaluation. This reformulation leads to the following:

- We extend the signature to accommodate rectangular matrices. Therefore, the Round 3 SNOVA scheme is characterized by seven parameters: $(v, o, q, l, r, m_1, m_2)$.
- We introduce specifications for base fields of odd prime order.

The reformulation provides greater flexibility in parameter selection while also offering advantages in security. We would like to point out that the Round 3 SNOVA scheme is backwards compatible with Round 2 version. The Round 3 submission can reproduce the Round 2 KAT files by using an appropriate set of parameters.

Changed and Added Parameters

Update parameter sets:

- New parameter sets using the Round 3 possibilities of rectangular signatures and the parameter sets with odd prime q.
- Use a fixed set of ABQ matrices for *all* parameter sets. They are now parameter dependent scheme constants.
- Introduce the value of ν in the description of SNOVA scheme using matrix equations.
- We have removed the "Expanded Secret Key" option, we only offer "Seeded Secret Keys".

Other Changes

- We have made editorial changes to the specification.
- We update the algorithm specifications to reflect the expanded parameter space.
- We update the section of analysis of known attacks.
- We use the hash of the public key during signing and verification instead of only its seed.
- We simplified the generation of the T^{12} matrix as well as the ABQ matrices.

Contents

1 Algorithm Specification (2.B.1)

In this supporting document, we present a detailed specification of multivariate signature scheme SNOVA: **S**imple **N**oncommutative unbalanced **O**il and **V**inegar scheme with randomness **A**lignment. In the following sections, we adapt and modify the paragraphs from the Round 2 supporting document [56], the original SNOVA paper proposed by Wang *et al.* [57] and the reformulation proposd by Ding *et al.* [17].

1.1 Introduction

SNOVA is a variant of the Unbalanced Oil and Vinegar (UOV) signature scheme [30] designed to operate over noncommutative rings for enhanced efficiency and reduced public key size. While traditional UOV schemes suffer from excessively large public keys, SNOVA addresses this by utilizing a key-randomness alignment technique that successfully adapts to noncommutative rings without being negatively affected by their noncommutativity.

In response to cryptanalysis during the Round 1 and Round 2 evaluations, the SNOVA scheme has been reformulated for the Round 3 submission to provide a more flexible framework. Recent research shows that SNOVA can be characterized as applying a whipping transformation, similar to that of MAYO, to a structured UOV public key utilizing the $\mathbb{F}_q[S]$ structure. Because the ring $\mathbb{F}_q[S]$ is isomorphic to the extension field $\mathbb{F}_{q^l}$, this design is comparable to the polynomial matrix technique utilized in QR-UOV.

The Round 3 reformulation extends the signature to accommodate rectangular matrices, yielding a scheme defined by seven parameters: $(v, o, q, l, r, m_1, m_2)$. This allows the whipping dimension r to be an independent parameter distinct from l[1], and the number of equations m_2 to be chosen independently from the number of public key matrices m_1. Additionally, SNOVA now introduces specifications for base fields of odd prime order to address recent cryptanalysis on the UOV family. This reformulation offers advantages in security and much greater flexibility in parameter selection, while remaining fully backward compatible with the Round 2 version.

1.2 Preliminaries

1.2.1 Notations and Conventions

The notation conventions and symbols with fixed meanings used throughout this document are provided in Table 1.

[1] This aligns with Beullens' observation in [9] that the whipping dimension r need not be chosen equal to l.

Table 1. The table of symbols fixed with specific meaning in this document.

Symbol	Description
$\mathbb{F}_q$	the finite field of order q
$\mathbb{F}_q^c$	the space of c-dimensional column vectors over $\mathbb{F}_q$
$\mathbb{F}_q^{r\times c}$	the set of matrices of size $r \times c$ with entries in $\mathbb{F}_q$
$[c]$	the set $\{1, \ldots, c\}$
$\mathcal{R} = \mathbb{F}_q^{l\times l}$	the set of $l \times l$ matrices over $\mathbb{F}_q$
$S \in \mathcal{R}$	the symmetric matrix with irreducible characteristic polynomial
$\mathbb{F}_q[S]$	the set $\{a_0 I + a_1 S + \cdots + a_{l-1} S^{l-1} : \ a_0, \ldots, a_{l-1} \in \mathbb{F}_q\}$
$\otimes$	the Kronecker tensor product
$\tau : \mathbb{F}_{q^l} \to \mathbb{F}_{q^l}$	the Frobenius map defined by $z \mapsto z^q$
$\mathbf{Q}^{\otimes n}$	the block-diagonal matrix with n copies of matrix Q along the diagonal, $I_n \otimes Q$
v	the number of vinegar variables
o	the number of oil variables
q	the order of the underlying finite field
l	the size of the matrix ring
r	the dimension of the whipping transformation
m_1	the number of public key matrices
m_2	the number of equations
$n = v + o$	the number of (ring) variables
ν	the number of terms in the description via matrix equations
$\mathsf{MQ}(N, M, q)$	complexity of a homogeneous MQ system of M equations in N variables over $\mathbb{F}_q$
$g(q)$	$g(q) = 2(\log_2 q)^2 + \log_2 q$, the ♯gates required for one $\mathbb{F}_q$ multiplication

1.2.2 Basic Notions

MQ problem. Let $\mathbb{F}_q$ be a finite field of order q. Given M quadratic polynomials $\mathcal{P}(\mathbf{x}) = (p_1(\mathbf{x}), \ldots, p_M(\mathbf{x}))$ in N variables $\mathbf{x} = (x_1, \ldots, x_N)$, the (homogeneous) Multivariate Quadratic (MQ) problem is to find a vector $\mathbf{u} \in \mathbb{F}_q^N$ such that $\mathcal{P}(\mathbf{u}) = (p_1(\mathbf{u}), \ldots, p_M(\mathbf{u})) = \mathbf{0}$. This problem is known to be NP-hard when $N \sim M$ [25]. Note that it is generically expected to be exponentially hard in the case $N \sim M$ and it can be solved in polynomial time for $M \geq \frac{N(N+1)}{2}$ or $N \geq M(M+1)$ [19].

In this supporting document, we use $\mathsf{MQ}(N, M, q)$ to denote the complexity of solving such an MQ problem. There are several algorithms to solve a multivariate quadratic system of M equations in N variables over finite fields such as F_4 [20], F_5 [21] and XL variants [15].

Polar forms. For a homogeneous multivariate quadratic map $\mathcal{P}(\mathbf{x}) = (p_1(\mathbf{x}), \ldots, p_M(\mathbf{x})) : \mathbb{F}_q^N \to \mathbb{F}_q^M$, the polar form of $\mathcal{P}$ is defined to be the map

$$\mathcal{P}'(\mathbf{x}, \mathbf{y}) = (p_1'(\mathbf{x}, \mathbf{y}), \ldots, p_M'(\mathbf{x}, \mathbf{y})) : \mathbb{F}_q^N \times \mathbb{F}_q^N \to \mathbb{F}_q^M$$

where for each $i \in \{1, \ldots, M\}$, the polar form of $p_i(\mathbf{x})$ is defined by

$$p_i'(\mathbf{x}, \mathbf{y}) = p_i(\mathbf{x} + \mathbf{y}) - p_i(\mathbf{x}) - p_i(\mathbf{y}).$$

Note that each $p'_i(\mathbf{x}, \mathbf{y})$ is symmetric and bilinear. If we write the quadratic map into the quadratic form $p_i(\mathbf{x}) = \mathbf{x}^\top \cdot \mathbf{P}_i \cdot \mathbf{x}$ where $\mathbf{P}_i$ denotes the matrix representation of p_i then the matrix representation of $p'_i(\mathbf{x}, \mathbf{y})$ is $\mathbf{P}_i + \mathbf{P}_i^\top$.

1.2.3 NIST Security Level

In the new call for additional digital signature scheme project [40], NIST suggested several security levels for post-quantum cryptosystem design. Herein, we focus on levels I, III, and V. The NIST security levels I, III and V require that a classical attacker needs 2^{143}, 2^{207} and 2^{272} classical gates to break the scheme, and 2^{61}, 2^{125} and 2^{189} quantum gates for a quantum attacker, respectively.

The number of gates required for an attack against a digital signature scheme can be computed by

$$\sharp\text{gates} = \sharp\text{field multiplication} \cdot (2 \cdot (\log_2 q)^2 + \log_2 q)$$

with the assumption that one field multiplication in the field $\mathbb{F}_q$ needs about $(\log_2 q)^2$ bit multiplications and same for bit additions, and for each field multiplication in the computation, it also needs an addition of field elements, each takes $\log_2 q$ bit additions. For the sake of the convenience on the notation, we define $g(x) = (2 \cdot (\log_2 x)^2 + \log_2 x)$.

Table 2. NIST Security Level.

Security Level (SL)	Classical gates	Quantum gates
I	143	61
III	207	125
V	272	189

1.2.4 Unbalanced Oil and Vinegar Signature (UOV) Scheme

The Unbalanced Oil and Vinegar (UOV) signature scheme [30] signature scheme is a slight modification of the Oil and Vinegar (OV) [42] signature scheme, proposed by Patarin in 1997. This scheme is based on a trapdoor map $\mathcal{F}$ which is easily inverted and it also can resist the KS attack [31] on OV scheme. A (v, o, q, m) UOV signature scheme with $v > o$ is defined with a tuple of positive integers so that the number of variables $n = v + o$, the number of equations $m = o$, and over $\mathbb{F}_q$.

Central map. The central map of the UOV scheme is $\mathcal{F} = (f_1, \ldots, f_m) : \mathbb{F}_q^n \to \mathbb{F}_q^m$ where each f_i is of the form

$$f_i(\mathbf{x}) = \sum_{j=1}^{v} \sum_{k=j}^{n} a_{i,jk} x_j x_k.$$

The coefficients $a_{i,jk}$'s are chosen randomly from $\mathbb{F}_q$. Note that each f_i is a homogeneous quadratic polynomials in n variables which has no terms $x_j x_k$ for $v + 1 \leq j, k \leq n$ over $\mathbb{F}_q$. The variables $x_1, \ldots, x_v$ are called the vinegar variables and $x_{v+1}, \ldots, x_n$ are called the oil variables.

Secret key and public key. The secret key of UOV is the pair $(\mathcal{F}, \mathcal{T})$ where $\mathcal{T} : \mathbb{F}_q^n \to \mathbb{F}_q^n$ is an invertible linear map which is randomly chosen. The public map is $\mathcal{P} = \mathcal{F} \circ \mathcal{T} : \mathbb{F}_q^n \to \mathbb{F}_q^m$. The quadratic form of $p_i(\mathbf{u})$ is $p_i(\mathbf{u}) = \mathbf{u}^\top \mathbf{P}_i \mathbf{u}$ where $\mathbf{u} = (u_1, \ldots, u_n)^\top$. Note that $\mathbf{P}_i = \mathbf{T}^\top \cdot \mathbf{F}_i \cdot \mathbf{T}$, where $1 \leq i \leq m$, are the public key matrices and $\mathbf{T}$ denotes the matrix representation of $\mathcal{T}$.

Signature. Let **msg** be the message to be signed. To generate a signature for the hash value $\mathbf{y} = \mathsf{Hash}(\mathbf{msg})$, we compute the signature $\mathbf{u}$ as follows. First, we randomly assign values to the vinegar variables $x_1, \ldots, x_v$ in the system $\mathcal{F}(\mathbf{x}) = \mathbf{y}$. The resulting system then becomes a linear

system in the oil variables $x_{v+1}, \ldots, x_n$ which can be solved with high probability. If the system has no solution, then we reassign the values $x_1, \ldots, x_v$ randomly. Finally, the signature is computed as $\mathbf{u} = \mathcal{T}^{-1}(\mathbf{x})$.

Verification. If $\mathcal{P}(\mathbf{u}) = \mathsf{Hash}(\mathbf{msg})$, the signature is accepted, otherwise rejected.

Oil space, $\mathcal{O}$. The special structure of $\mathcal{F}$ in the UOV scheme indicates that $\mathcal{F}$ vanishes on the linear space $\mathcal{O} = \{\mathbf{x} = (x_1, \ldots, x_n)^\top \in \mathbb{F}_q^n : x_1 = \cdots = x_v = 0\}$. The space $\mathcal{O}$ is called the oil space of central map $\mathcal{F}$. And the oil space of public map $\mathcal{P}$ will be the space $\mathcal{T}^{-1}(\mathcal{O})$.

1.3 Parameter Space of the SNOVA Scheme

The parameter set of a SNOVA scheme is described by a tuple $(v, o, q, l, r, m_1, m_2)$ of positive integers.

- v, the number of vinegar variables over the $l \times l$ matrix ring $\mathbb{F}_q^{l\times l}$.
- o, the number of oil variables over the $l \times l$ matrix ring $\mathbb{F}_q^{l\times l}$. Note that the SNOVA scheme requires $v > o$.
- q, the order of the underlying finite field $\mathbb{F}_q$.
- l, the size of the matrix ring $\mathbb{F}_q^{l\times l}$ and the size of symmetric matrix $S \in \mathbb{F}_q^{l\times l}$ whose characteristic polynomial is irreducible over $\mathbb{F}_q$.
- r, the width of the matrices of the signature, which is also the dimension of the whipping transformation.
- m_1, the number of public key matrices over $\mathbb{F}_q^{ln\times ln}$. The public key matrices of SNOVA scheme are $\mathbf{P}_1, \ldots, \mathbf{P}_{m_1} \in \mathbb{F}_q^{ln\times ln}$.
- m_2, the number of equations in the public map $\mathcal{P}^*(\mathbf{U})$ of SNOVA.
- ν, the number of terms in the description via matrix equations (1.3). The current proposal incorporates the matrix equation description (1.3) with $\nu = lr + 2r$.

The number of ring variables in both the central map $\mathcal{F}$ and the public map $\mathcal{P}^*$ is $n = v + o$.

1.4 Design Rationale

We believe that multivariate signature schemes are useful in practice due to their compact signature sizes and efficient signature verification. However, the problem of suffering large public key size makes their application less practical. Therefore, we aim to design a multivariate signature scheme over noncommutative rings while addressing the issue of large public key sizes.

Due to its simplicity, UOV scheme [30] is an ideal test ground of these ideas. Although the idea of using noncommutative rings applies to general noncommutative rings, we decide to start from the matrix ring. It would be not wise to simply generalize UOV over finite fields to over noncommutative rings. Therefore, we adopt several techniques in our design.

SNOVA is a variant of the UOV scheme [30] that introduces additional matrix ring structure to minimize public-key size while preserving the performance. SNOVA is originally designed using concepts from non-commutative rings. The recent result [9] also shows that SNOVA can alternatively be characterized as applying a whipping transformation, which similar to that of MAYO [10], to a structured UOV public key that utilizing the $\mathbb{F}_q[S]$ structure. Note that the $\mathbb{F}_q[S]$ subring is isomorphic to the extension field $\mathbb{F}_{q^l}$. It is worth noting that this structure is similar to

the polynomial matrix technique in the QR-UOV [23]. Consequently, each public-key implicitly represents a family of l^2 corresponding to UOV polynomials linked by block-wise multiplication by the matrix $(\mathbf{S}^a)^{\otimes n}$ on the left hand side and $(\mathbf{S}^b)^{\otimes n}$ on the right hand side.

In the Round 1 and Round 2 evaluations, several attacks were proposed [27, 33, 9, 14, 55, 51, 36, 24, 47, 13, 52, 22, 18, 28]. We are very grateful for these analyses. On the other hand, in light of recent developments [17, 34], we observe that the core design of SNOVA and other multivariate signature schemes are closely related. Furthermore, we find that this core design of SNOVA can be expressed in three distinct languages: the ring equation, the whipping transformation and the tensor language.

The Round 3 SNOVA yields a more flexible framework that enables greater flexibility in parameter selection. In addition, the connections between SNOVA and other multivariate schemes become much clearer. The new framework allows us to enhance security through optimized parameter choices. Additionally, we provide a more comprehensive analysis in this supporting document. Based on the advantages of SNOVA, we believe that SNOVA is both highly competitive and practical.

1.5 SNOVA Signature Scheme

In this section, we present a detailed description of the SNOVA signature scheme.

1.5.1 Description

Let v, o, q, l, r, m_1, m_2 be positive integers with $v > o$ and let $\mathbb{F}_q$ be a field of order q. Let $n = v + o$ denotes the number of ring variables. A $(v, o, q, l, r, m_1, m_2)$ SNOVA scheme is described as follows.

Subring $\mathbb{F}_q[S]$ and elements in $\mathbb{F}_q[S]$. First, we define the subring $\mathbb{F}_q[S]$ of the matrix ring $\mathbb{F}_q^{l\times l}$ Let S be a $l \times l$ symmetric matrix with an irreducible characteristic polynomial. The subring $\mathbb{F}_q[S]$ of $\mathbb{F}_q^{l\times l}$ is defined as

$$\mathbb{F}_q[S] = \{a_0 I + a_1 S + \cdots + a_{l-1} S^{l-1} \mid a_0, a_1, \ldots, a_{l-1} \in \mathbb{F}_q\}.$$

Since the characteristic polynomial of S is irreducible, $\mathbb{F}_q[S]$ is isomorphic to the extension field $\mathbb{F}_{q^l}$. Thus, the elements in $\mathbb{F}_q[S]$ are symmetric and they all commute. It is worth noting that such matrix S is guaranteed to exist. For explicit constructions, we refer the reader to [7] and [32] for the even- and odd-characteristic cases, respectively. The selections of the matrix S of Round 3 SNOVA are specified in Section 1.7.

Secret key. The secret key consists of

- random matrices $\mathbf{F}_1, \ldots, \mathbf{F}_{m_1} \in \mathbb{F}_q^{ln\times ln}$, for each $i \in \{1, \ldots, m_1\}$, $\mathbf{F}_i$ is of the form

$$\mathbf{F}_i = \begin{bmatrix} \mathbf{F}_i^{11} & \mathbf{F}_i^{12} \\ \mathbf{F}_i^{21} & \mathbf{0} \end{bmatrix}$$

  where $\mathbf{F}_i^{11} \in \mathbb{F}_q^{lv\times lv}, \mathbf{F}_i^{12} \in \mathbb{F}_q^{lv\times lo}$ and $\mathbf{F}_i^{21} \in \mathbb{F}_q^{lo\times lv}$.

- invertible matrix $\mathbf{T} \in (\mathbb{F}_q[S])^{n\times n}$, more specifically, $\mathbf{T}$ is of the form

$$\mathbf{T} = \begin{bmatrix} \mathbf{I}^{11} & \mathbf{T}^{12} \\ \mathbf{0} & \mathbf{I}^{22} \end{bmatrix},$$

  where $\mathbf{T}^{12}$ is a $v \times o$ matrix whose entries are $l \times l$ matrices sampled uniformly at random from $\mathbb{F}_q[S]$, $\mathbf{I}^{11}$ is a $v \times v$ identity matrix and $\mathbf{I}^{22}$ is a $o \times o$ identity matrix. Note that the

corresponding matrix $\mathbf{T}^{-1}$ is of the form

$$\mathbf{T}^{-1} = \begin{bmatrix} \mathbf{I}^{11} & -\mathbf{T}^{12} \\ \mathbf{0} & \mathbf{I}^{22} \end{bmatrix}.$$

Note that we can use the secret key seed $\mathbf{s_{secret}}$ to generate $\mathbf{F}_1, \ldots, \mathbf{F}_{m_1}$ and $\mathbf{T}$.

Public key. The public key consists of $\mathbf{P}_1, \ldots, \mathbf{P}_{m_1} \in \mathbb{F}_q^{ln \times ln}$. Note that we construct the public key matrices by the congruence relation, i.e.,

$$\mathbf{P}_1 = \mathbf{T}^\top \mathbf{F}_1 \mathbf{T}, \ldots, \mathbf{P}_{m_1} = \mathbf{T}^\top \mathbf{F}_{m_1} \mathbf{T}.$$

A more detailed description of key generation algorithms is provided in the Section 1.8.1.

Bilinear map. For $i \in \{1, \ldots, m_1\}$ and $a, b \in \{0, \ldots, l-1\}$, we define $\mathbf{P}_{i,a,b} := (\mathbf{S}^a)^{\otimes n} \cdot \mathbf{P}_i \cdot (\mathbf{S}^b)^{\otimes n}$. Let $\mathcal{B} : \mathbb{F}_q^{ln} \times \mathbb{F}_q^{ln} \to \mathbb{F}_q^{l^2 m_1}$ be the bilinear map defined by

$$\mathcal{B}_{i,a,b}(\mathbf{u}, \mathbf{v}) := B_i^{(a,b)}(\mathbf{u}, \mathbf{v}),$$

where, for $i \in \{1, \ldots, m_1\}$ and $(a, b) \in \{0, \ldots, l-1\}^2$,

$$B_i^{(a,b)} : \mathbb{F}_q^{ln} \times \mathbb{F}_q^{ln} \to \mathbb{F}_q, \; B_i^{(a,b)}(\mathbf{u}, \mathbf{v}) := \mathbf{u}^\top \cdot \mathbf{P}_{i,a,b} \cdot \mathbf{v}.$$

Public map. The public map $\mathcal{P}^*(\mathbf{U}) : \mathbb{F}_q^{rln} \to \mathbb{F}_q^{m_2}$ of Round 3 SNOVA is defined by

$$\mathcal{P}^*(\mathbf{U}) = \sum_{j=0}^{r-1} \sum_{k=0}^{r-1} \mathbf{E}_{j,k} \cdot \mathcal{B}(\mathbf{u}_j, \mathbf{u}_k). \tag{1.1}$$

where $\mathbf{u}_j \in \mathbb{F}_q^{ln}$ is the $(j+1)$-th column of $\mathbf{U}$ for $j \in \{0, \ldots, r-1\}$ and for all $j, k \in \{0, \ldots, r-1\}$, the matrices $\mathbf{E}_{j,k} \in \mathbb{F}_q^{m_2 \times l^2 m_1}$ are the matrices we used to scramble the entries of the bilinear map $\mathcal{B}$. When m_2 is chosen to be different from $l^2 m_1$, the matrices $\mathbf{E}_{j,k}$ become rectangular[2]. As the result, the number of equations m_2 in the public map can be chosen independent from the parameters l and m_1. In our implementation, the matrices $\mathbf{E}_{jk}$ are parameter dependent scheme constants generated from a fixed seed. For more details, we refer to Section 1.8.1.

Remark 1.1. In [34], Li *et al.* have shown that the whipping form and tensor are equivalent. Therefore, one can also write the form (1.1) into the tensor form. Note that $\mathcal{P}^*(\mathbf{U})$ can be regarded as a quadratic map of m_2 equations in rln variables. By relabeling the index, for $s \in \{1, \ldots, m_2\}$, the matrix corresponding to the s-th entry of $\mathcal{P}^*(\mathbf{U})$ is

$$\mathbf{P}_s^* = \sum_{j=1}^{m_1} \sum_{a,b} E^{s,j,a,b} \otimes \mathbf{P}_{j,a,b}, \tag{1.2}$$

where $s \in \{1, \ldots, m_2\}$, $E^{s,j,a,b} \in \mathbb{F}_q^{r \times r}$ and $rlo \geq m_2$. Note that the matrices $\mathbf{E}_{j,k}$ or $E^{s,j,a,b}$ can be generated from a seed. Note that $\mathbf{E}_{j,k}$ and $E^{s,j,a,b}$ are just the rearrangement of each other.

Signature. Let **msg** be the message to be signed and $\mathsf{Hash}(\mathbf{msg}) \in \mathbb{F}_q^{m_2}$ be its hash value. We compute the signature $\mathbf{U} \in \mathbb{F}_q^{ln \times r}$ step by step with the common Oil-Vinegar approach: First, we assign values randomly to the vinegar variables. Second, we solve the resulting system in oil variables, which is linear due to the Oil-Vinegar structure. Note that r, m_2 are the number of columns of $\mathbf{U}$ and the number of equations in $\mathcal{P}^*(\mathbf{U})$, respectively. The signature $\mathbf{U}$ is the preimage of $\mathsf{Hash}(\mathbf{msg}) \in \mathbb{F}_q^{m_2}$ under the public map $\mathcal{P}^*$. It would be required that $rlo \geq m_2$ to ensure that $\mathcal{P}^*$ can efficiently find a preimage by utilizing the Oil-Vinegar structure. A more detailed description of signing process is provided in the Section 1.8.2.

[2] A similar idea can be found in the Appendix of [56] and [34].

Verification. Let $\mathbf{U} \in \mathbb{F}_q^{ln \times r}$ be the signature to be verified. If $\mathcal{P}^*(\mathbf{U}) = \mathsf{Hash}(\mathbf{msg})$, then the signature is accepted, otherwise rejected. A more detailed description of signing process is provided in the Section 1.8.3.

Description via the matrix equations. On the other hand, we can still use matrix equations to describe the general mapping[3]. In general, if we assume that $A_{i,\alpha} \in \mathbb{F}_q^{r_1 \times r}$ and $B_{i,\alpha} \in \mathbb{F}_q^{r \times r_2}$ for some positive integers r_1 and r_2, we can describe the public map $\mathcal{P}^*(\mathbf{U})$ via the matrix equations. That is, $\mathcal{P}^* = (\mathcal{P}_1^*, \ldots, \mathcal{P}_m^*) : \mathbb{F}_q^{rln} \to \mathbb{F}_q^{mr_1r_2}$, for $i \in \{1, \ldots, m\}$,

$$\mathcal{P}_i^*(\mathbf{U}) = \sum_{\alpha=0}^{\nu-1} A_{i,\alpha} \cdot \mathbf{U}^\top \cdot \mathbf{Q}_{i,\alpha,1}^{\otimes n} \cdot \mathbf{P}_{i'} \cdot \mathbf{Q}_{i,\alpha,2}^{\otimes n} \cdot \mathbf{U} \cdot B_{i,\alpha} \in \mathbb{F}_q^{r_1 \times r_2}. \tag{1.3}$$

where m is a positive integer, $\mathbf{U} = (U_1, \ldots, U_n) \in \mathbb{F}_q^{ln \times r}$ and $i' = (i + \alpha) \pmod{m_1}$. In this case, we require $rlo \geq mr_1r_2 = m_2$ to ensure that $\mathcal{P}^*$ can efficiently find a preimage by exploiting the Oil-Vinegar structure. The detailed explanations of the equivalence between the matrix formulation (1.3) and the whipping formulation (1.1) can be found in [9, 34, 17]. For our fixed set of ABQ matrices, the matrix formulation and the whipping formulation are essentially equivalent. It is worth noting that describing the scheme using matrix equations can lead to a more efficient implementation. For more details, we refer to Section 1.8.

Relation to the other UOV variants. Recall that the public map (1.1) is defined by

$$\mathcal{P}^*(\mathbf{U}) = \sum_{j=0}^{r-1} \sum_{k=0}^{r-1} \mathbf{E}_{j,k} \cdot \mathcal{B}(\mathbf{u}_j, \mathbf{u}_k).$$

where $\mathbf{u}_j$ is the $(j+1)$-th column of $\mathbf{U}$ for $j \in \{0, \ldots, r-1\}$ and for all $j, k \in \{0, \ldots, r-1\}$, the matrices $\mathbf{E}_{j,k} \in \mathbb{F}_q^{m_2 \times l^2 m_1}$.

- When $l = 1$ and $m_1 = m_2 = m$, the public map $\mathcal{P}^*(\mathbf{U})$ is equivalent to the public map of Round 2 MAYO [10] with some different choices of $\mathbf{E}_{j,k}$.
- When $l = 1$, $m_1 = m_2 = m = o$ and $r = 1$, the public map $\mathcal{P}^*(\mathbf{U})$ is equivalent to the public map of UOV [11].

Obviously there are differences in the details, e.g. MAYO uses an upper triangular matrix and MAYO also uses a different $\mathbf{E}_{jk}$ matrix and byte order. We have created a SageMath version that recreates Round 2 KAT files for both MAYO and SNOVA. For those interested, this SageMath implementation [50] can be used to find *all* similarities and differences between MAYO and SNOVA.

1.5.2 Key Generation

In this section, we describe the key generation processes of SNOVA.

Standard key generation process. For $i \in \{1, \ldots, m_1\}$, the matrix $\mathbf{P}_i$ is obtained by the relation

$$\mathbf{T}^\top \mathbf{F}_i \mathbf{T} = \mathbf{P}_i = \begin{bmatrix} \mathbf{P}_i^{11} & \mathbf{P}_i^{12} \\ \mathbf{P}_i^{21} & \mathbf{P}_i^{22} \end{bmatrix}.$$

For each $i \in \{1, \ldots, m_1\}$, $\mathbf{F}_1, \ldots, \mathbf{F}_{m_1} \in \mathbb{F}_q^{ln \times ln}$, $\mathbf{F}_i$ is of the form

$$\mathbf{F}_i = \begin{bmatrix} \mathbf{F}_i^{11} & \mathbf{F}_i^{12} \\ \mathbf{F}_i^{21} & \mathbf{0} \end{bmatrix}$$

[3] For some specific set of ABQ matrices $\{A_{i,\alpha}, B_{i,\alpha}, Q_{i,\alpha,1}, Q_{i,\alpha,2}\}$, we can construct the corresponding $\mathbf{E}_{j,k}$ matrices by assembling the coefficients and the terms of the ABQ matrices, and vice versa [54].

where $\mathbf{F}_i^{11} \in \mathbb{F}_q^{lv\times lv}, \mathbf{F}_i^{12} \in \mathbb{F}_q^{lv\times lo}$ and $\mathbf{F}_i^{21} \in \mathbb{F}_q^{lo\times lv}$. Therefore, due to the congruence relation $\mathbf{P}_i = \mathbf{T}^\top \mathbf{F}_i \mathbf{T}$, we have the following relations

$$\begin{aligned}
\mathbf{P}_i^{11} &= \mathbf{F}_i^{11} \\
\mathbf{P}_i^{12} &= \mathbf{F}_i^{12} + \mathbf{F}_i^{11}\mathbf{T}^{12} \\
\mathbf{P}_i^{21} &= \mathbf{F}_i^{21} + (\mathbf{T}^{12})^\top \mathbf{F}_i^{11} \\
\mathbf{P}_i^{22} &= (\mathbf{T}^{12})^\top \mathbf{F}_i^{11}\mathbf{T}^{12} + (\mathbf{T}^{12})^\top \mathbf{F}_i^{12} + \mathbf{F}_i^{21}\mathbf{T}^{12}.
\end{aligned}$$

Therefore, to generate the public key matrices $\mathbf{P}_1, \ldots, \mathbf{P}_{m_1}$ we generate the matrices $\mathbf{F}_1, \ldots, \mathbf{F}_{m_1}$ and $\mathbf{T}$ from the secret seed $\mathbf{s_{secret}}$ at first and then compute the public key $\mathbf{P}_1, \ldots, \mathbf{P}_{m_1}$ with the formulas above.

Key generation with randomness alignment. The following are steps of key generation process of SNOVA with key randomness alignment technique proposed by Albrecht Petzoldt [43].

First Step: Fix an $l \times l$ symmetric matrix S with irreducible characteristic polynomial. Generate $\mathbf{P}_i^{11}$, $\mathbf{P}_i^{12}$ and $\mathbf{P}_i^{21}$ for $i \in \{1, \ldots, m_1\}$ from public key seed $\mathbf{s_{public}}$. Generate $\mathbf{T}$ from secret key seed $\mathbf{s_{secret}}$.

Second Step: Compute the matrix $\mathbf{F}_i^{11}, \mathbf{F}_i^{12}, \mathbf{F}_i^{21}, \mathbf{P}_i^{22}$ for $i \in \{1, \ldots, m_1\}$ as below. Inverting the relation between $\mathbf{F}_i$ and $\mathbf{P}_i$ above, the following equations hold

$$\begin{aligned}
\mathbf{F}_i^{11} &= \mathbf{P}_i^{11} \\
\mathbf{F}_i^{12} &= \mathbf{P}_i^{12} - \mathbf{P}_i^{11}\mathbf{T}^{12} \\
\mathbf{F}_i^{21} &= \mathbf{P}_i^{21} - (\mathbf{T}^{12})^\top \mathbf{P}_i^{11}.
\end{aligned}$$

In other words, we have

$$\begin{aligned}
\mathbf{P}_i^{22} &= (\mathbf{T}^{12})^\top \mathbf{F}_i^{11}\mathbf{T}^{12} + (\mathbf{T}^{12})^\top \mathbf{F}_i^{12} + \mathbf{F}_i^{21}\mathbf{T}^{12} \\
&= (\mathbf{T}^{12})^\top \mathbf{P}_i^{11}\mathbf{T}^{12} + (\mathbf{T}^{12})^\top \left(\mathbf{P}_i^{12} - \mathbf{P}_i^{11}\mathbf{T}^{12}\right) + \left(\mathbf{P}_i^{21} - (\mathbf{T}^{12})^\top \mathbf{P}_i^{11}\right)\mathbf{T}^{12} \\
&= (\mathbf{T}^{12})^\top \left(\mathbf{P}_i^{12} - \mathbf{P}_i^{11}\mathbf{T}^{12}\right) + \mathbf{P}_i^{21}\mathbf{T}^{12}.
\end{aligned}$$

1.5.3 To Attain EUF-CMA Security

For practical considerations, we use a random binary vector, called **salt** in order to achieve Existential Unforgeability under Chosen Message Attack (EUF-CMA) Security [39, 48]. In addition, to protect against malicious public key attacks, we include a hash of the public key in the signing process. With this modification the arguments of Aulbach *et al.* [5] on PROV also apply to Round 3 SNOVA (Proposition 11 in [5]). Aulbach *et al.* suggested to use the complete public key alongside the message into the hash function, but this would substantially increase the size of the secret key. We use a hash of the public key instead. With this (modified) PS-3 transform [44] SNOVA achieves the BUFF (Beyond UnForgeability Features) security properties defined by Cremers *et al.* [16].

Signature. Let **msg** be the message to be signed and $\mathbf{m_d} = \mathsf{SHAKE256}(\mathbf{msg}, 64)$ its message digest. The public key **pk** is hashed into $\mathbf{pk_d} = \mathsf{SHAKE256}(\mathbf{pk}, 48)$. We randomly choose a **salt** and then generate a signature for the hash value $\mathsf{Hash}(\mathbf{pk_d}||\mathbf{m_d}||\mathbf{salt})$.

Therefore, the corresponding signature with salt is of the form $\mathbf{sig} = (\mathbf{U}||\mathbf{salt})$ where $\mathbf{U}$ is the signature of $\mathsf{Hash}(\mathbf{pk_d}||\mathbf{m_d}||\mathbf{salt})$ generated by the SNOVA signer. While there is no immediate security risk if salts are used more than once, each signature generated should use a different random value of **salt**. Therefore, the length of **salt** is chosen to be 16 bytes under the assumption of up to 2^{64} signatures are to be generated with the system [40].

Verification. If $\mathcal{P}^*(\mathbf{U}) = \mathsf{Hash}(\mathbf{pk_d}||\mathbf{m_d}||\mathbf{salt})$, the signature is accepted, otherwise rejected.

1.6 Implementation Details

In this section, we describe the details about the implementation.

Secret key. The secret key consists of the public and secret key seeds, and a hash of the public key. The secret key expansion and the expansion of the random part of the public key are included in both the key generation procedure and the signing procedure of the signer.

Symmetric public matrix. The public matrix can be taken to be symmetric when the characteristic of the field $\mathbb{F}_q$ is odd. This allows for a reduction of the public key size but the security analysis changes. We currently do not recommended parameter sets with a symmetric public matrix pending additional cryptanalysis.

Symmetric primitives. In the SNOVA scheme, multiple hash functions are used as well as two XOF (eXtendable Output Function). We categorize the parts that needed hash functions and XOF and explain which instance we take in each case:

- The length of secret key seed, $|\mathbf{s_{secret}}|$: 32 bytes.
- The length of public key seed, $|\mathbf{s_{public}}|$: 16 bytes.
- The length of public key hash, $|\mathbf{pk_d}|$: 48 bytes.
- The length of **salt**, $|\mathbf{salt}|$: 16 bytes.
- The XOF which is used to generate the secret matrix T: SHAKE256.
- The XOF which is used to generate the random part of public matrix: AES128CTR or SHAKE128.
- The hash function: SHAKE256. A hash is required on multiple occasions, for example to generate the hash value to be signed.

The secret key size, $|\mathbf{sk}|$, is 96 bytes for all parameter sets, it is the sum of the length of secret and public key seeds as well as the public key hash.

To generate the long pseudorandom part of the public key efficiently, we have adopted AES128CTR as XOF. A variant that uses SHAKE128 (Secure Hash Algorithm KECCAK) for the public key expansion is offered as an alternative to AES128CTR. This variant is specified below by **Algorithm** 5. The default is to use SHAKE128 as the public XOF.

An implementation may choose to introduce a function to convert the secret key into some conveniently expanded secret key when multiple signatures using the same secret key are to be created. We consider this an implementation choice and not a part of the specification of SNOVA. We found that such caching of the expanded secret key can result in a speed-up of the signing by about 20% to 60%.

1.7 Constants and Tables

The finite field $\mathbb{F}_q$. For prime q this field is defined by usual arithmetic modulo q.

The finite field $\mathbb{F}_{16}$. Fix an irreducible polynomial $f(x) = x^4 + x + 1$ over $\mathbb{F}_2$ and consider that the finite field $\mathbb{F}_{16}$ consists of the polynomials $ax^3 + bx^2 + cx + d \in \mathbb{F}_2[x] \bmod f(x)$. The elements of $\mathbb{F}_{16}$ are stored in 4 bits and then the addition of two elements in $\mathbb{F}_{16}$ is equal to the bitwise XOR of them.

The matrix S. For simplicity, we represent elements of $\mathbb{F}_{16}$ as decimal numbers. In particular, an element $ax^3 + bx^2 + cx + d$ of $\mathbb{F}_{16}$ is converted to an integer $2^3a + 2^2b + 2c + d$. We fix the

matrices S as follows:

$$S = \begin{bmatrix} 8 & 7 \\ 7 & 6 \end{bmatrix} \quad \text{if } q = 16 \text{ and } l = 2,$$

$$S = \begin{bmatrix} 8 & 7 & 6 & 5 \\ 7 & 6 & 5 & 4 \\ 6 & 5 & 4 & 3 \\ 5 & 4 & 3 & 2 \end{bmatrix} \quad \text{if } q = 16 \text{ and } l = 4.$$

$$S = \begin{bmatrix} 8 & 7 & 6 & 5 & 4 \\ 7 & 6 & 5 & 4 & 3 \\ 6 & 5 & 4 & 3 & 2 \\ 5 & 4 & 3 & 2 & 1 \\ 4 & 3 & 2 & 1 & 9 \end{bmatrix} \quad \text{if } q = 16 \text{ and } l = 5.$$

One can check that the characteristic polynomials of these matrices S are irreducible over $\mathbb{F}_{16}$.

The matrix S_q. For prime q we use the following sets of matrices:

$$\begin{array}{ll} S(i,j) = (q_a + i + j) \textbf{ and } q_b & (i \neq l-1) \text{ or } (j \neq l-1) \\ S(i,j) = q_c & (i = l-1) \text{ and } (j = l-1) \end{array}$$

where **and** is a bitwise and. For the parameters q_a, q_b, q_c we use the values in Table 3. It can be verified that for $l = 2, 4, 5$ all S matrices are irreducible.

Table 3. Parameters of the S matrix depending on the field order q.

q	q_a	q_b	q_c
11	0	3	6
13	2	11	3
17	1	11	10
19	1	3	15
23	1	11	22
29	3	12	11
31	2	5	8

This results in for example:

$$S = \begin{bmatrix} 1 & 2 & 3 & 0 \\ 2 & 3 & 0 & 1 \\ 3 & 0 & 1 & 2 \\ 0 & 1 & 2 & 15 \end{bmatrix} \quad \text{if } q = 19 \text{ and } l = 4.$$

$$S = \begin{bmatrix} 1 & 2 & 3 & 0 & 1 \\ 2 & 3 & 0 & 1 & 2 \\ 3 & 0 & 1 & 2 & 3 \\ 0 & 1 & 2 & 3 & 0 \\ 1 & 2 & 3 & 0 & 15 \end{bmatrix} \quad \text{if } q = 19 \text{ and } l = 5.$$

Fixed ABQ matrices. We use the ASCII string "SNOVA_ABQ" and SHAKE256 for the generation of the ABQ matrices.

1.8 Algorithms

In the algorithm descriptions we have chosen to use a different sign convention for the **T** matrix. We use the convention $T^{12} = -\mathbf{T}^{12}$ in this section. The **for** loops are up to and including. Therefore, a loop such as **for** $i_1 \leftarrow 0$ **to** $l-1$ **do** ... **end** will also process $i_1 = l-1$. The set indices start at 0, so a set containing for example r numbers will be $\{0, \ldots, r-1\}$. We set $r_1 = r$, $r_2 = l$ and $m_2 = o \cdot l \cdot r$ in the algorithm specifications. The number of matrix equation is denoted by m, but as $m = o$ we use o in the following to prevent confusion with m_1 and m_2.

1.8.1 Algorithms for Key Generation

First we describe algorithms that are used by all functions.

Conversion between field elements and bytes. To convert N_q elements of $\mathbb{F}_q$ with coefficients $c_{q,i}$ into N_b bytes $c_{b,i}$ we calculate the sum

$$s_q = \sum_{i=0}^{N_q-1} c_{q,i} \cdot q^i,$$

and store the result as bytes, least significant byte first. This relation can easily be inverted. The values of N_q and N_b are such that any sequence of $\mathbb{F}_q$ can be converted to bytes. The reverse does not hold. If the input provided is a public key or a signature, and the sum of bytes is too large

$$s_b = \sum_{i=0}^{N_b-1} c_{b,i} \cdot q^i \geq q^{N_q},$$

then the input provided is illegal input, and the calling algorithm will exit with an error.

SNOVA uses the following values for N_q and N_b:

Table 4. Parameters of conversion $\mathbb{F}_q \Leftrightarrow$ bytes. The last two columns present the minimal number of bits required to encode an $\mathbb{F}_q$ element, and the actual number of bits used.

q	N_q	N_b	$\log_2(q)$	$8N_b/N_q$
11	16	7	3.459	3.5
13	15	7	3.7	3.733
16	2	1	4	4
17	15	8	4.087	4.267
19	15	8	4.248	4.267
23	7	4	4.524	4.571
29	13	8	4.858	4.923
31	8	5	4.954	5

Note that for $q = 16$ this is the nibble ordering used in Rounds 1 and 2. The specified conversion allows for an efficient packing, reducing the public key and signature sizes.

Algorithm 1: Convert elements of $\mathbb{F}_q$ to bytes

input : n elements $a_0, \ldots, a_{n-1}$ of $\mathbb{F}_q$
output: a byte string

$\mathbf{x} \leftarrow 0^0$
for $j \leftarrow 0$ **to** $\lceil (n-1)/N_q \rceil$ **do**
  $s_q \leftarrow \sum_{i=0}^{N_q-1} q^i a_{i+jN_q}$
  for $k \leftarrow 1$ **to** N_b **do**
    $\mathbf{x} \leftarrow \mathbf{x} \,||\, (s_q \bmod 256)$
    $s_q \leftarrow s_q/256$
  end
end
return $\mathbf{x}$

Algorithm 2: Convert bytes to elements of $\mathbb{F}_q$

input : n bytes $b_0, \ldots, b_{n-1}$
output: A string of elements of $\mathbb{F}_q$ if the input is legal

$x \leftarrow 0^0$
for $j \leftarrow 0$ **to** $\lceil (n-1)/N_b \rceil$ **do**
  $s_b \leftarrow \sum_{i=0}^{N_b-1} (256)^i b_{i+jN_b}$
  for $k \leftarrow 1$ **to** N_q **do**
    $x \leftarrow x \,||\, (s_b \bmod q)$
    $s_b \leftarrow s_b/q$
  end
  Raise an error if the input is a public key or a signature, and $s_b \neq 0$
    ▷ This check for legal input only applies when verifying.
end
return x

For performance reasons, the output of the public XOF is processed by a slightly different algorithm when $q \neq 16$. This expansion does not use rejection sampling of the public key data. In [2], it was shown that skipping the rejection sampling for the public key data does not significantly affect the security of QR-UOV. The argument applies to SNOVA without modification.

Algorithm 3: Expand public XOF data

input : data d as bytes
output: data as elements of $\mathbb{F}_q$

$r \leftarrow \{\}$
for b *in* d **do**
  if *q=16* **then**
    $b \leftarrow b \,||\, (d \mod 16)$
    $b \leftarrow b \,||\, (d/16)$
  else
    $b \leftarrow b \,||\, (d \mod q)$
  end
end
return b

The key generation process with key-randomness alignment technique is as follows.

First Step: Fix an $l \times l$ symmetric matrix S as in Section 1.7. Generate $T^{12} \in \mathbb{F}_q^{v \cdot l \times o \cdot l}$ from secret key seed $\mathbf{s_{secret}}$.

Algorithm 4: Generate the secret matrix T^{12}

input : Secret key seed $\mathbf{s_{secret}}$
output: The matrix T^{12}

$\mathbf{t_d} \leftarrow \mathsf{SHAKE256}(\mathbf{s_{secret}})$
$b \leftarrow \{\}$
$i \leftarrow 0$
$j \leftarrow 0$
while $i < o \cdot v \cdot l$ **do**
  if *q=16* **then**
    $b \leftarrow b \,||\, (\mathbf{t_d}[j] \mod 16)$
    $b \leftarrow b \,||\, (\mathbf{t_d}[j]/16)$
    $i \leftarrow i + 2$
  else
    if $t_d[j] < q\lfloor 256/q \rfloor$ **then**
      $b \leftarrow b \,||\, (\mathbf{t_d}[j] \mod q)$
      $i \leftarrow i + 1$ ▷ Rejection sampling.
    end
  end
  $j \leftarrow j + 1$
end
for $i \leftarrow 0$ **to** $v - 1$ **do**
  for $j \leftarrow 0$ **to** $o - 1$ **do**
    $t_{12} \leftarrow \sum_{k=0}^{l-1} b[(i \cdot o + j) \cdot l + k] \cdot S^k$
    for $i_1 \leftarrow 0$ **to** $l - 1$ **do**
      for $j_1 \leftarrow 0$ **to** $l - 1$ **do**
        $T^{12}[i \cdot l + i_1, j \cdot l + j_1] \leftarrow t_{12}[i_1, j_1]$
      end
    end
  end
end
return T^{12}

To generate the longer random part of the public key efficiently, we have adopted $\mathsf{AES128CTR}$ as XOF. A variant that uses $\mathsf{SHAKE128}$ (Secure Hash Algorithm KECCAK) for the public key expansion has been added in the Round 2 submission as an alternative to $\mathsf{AES128CTR}$. This variant can be both vectorized and indexed. We denote the XOF of this variant as $\mathsf{XOF_P}$. The algorithm of $\mathsf{XOF_P}$ can be described by: extract the bytes from $\mathsf{SHAKE128}$ in 168 bytes blocks where a block index is appended to the seed. The block size 168 follows from the rate of $\mathsf{SHAKE128}$. A more precise definition of $\mathsf{XOF_P}$ is: Let $\mathsf{SHAKE128}(s, n)$ denote the n-th byte of $\mathsf{SHAKE128}$ when instantiated with seed s as input, and similarly $\mathsf{XOF_P}(s, n)$. Then for all required bytes:

$$\mathsf{XOF_P}(s, n) = \mathsf{SHAKE128}(s \,||\, \lfloor n/168 \rfloor, n \bmod 168) \tag{1.4}$$

where the block index $b = \lfloor n/168 \rfloor$ is the 8 bytes little-endian representation of $n/168$ rounded to below, $||$ represents concatenation of bytes and mod is the integer modulus operation. Note that the invocation of $\mathsf{SHAKE128}$ can be executed in parallel for all values of the block index b. The main advantage of using SHAKE this way as a public XOF is that it allows for arbitrary levels of parallelization in the squeeze phase. The library XKCP allows for up to an 8-fold parallelization of the KECCAK permutation on AVX-512 processors. The official SNOVA implementation [50] uses this level of parallelism if available.

The algorithm of $\mathsf{XOF_P}$ is included in the following algorithm:

Algorithm 5: Public key expansion

input : Seed $\mathbf{s_{public}}$
The requested number of bytes N
AES: a flag indicating whether to use AES of SHAKE
output: Pseudo-random bytes

if AES **then**
  return $\mathsf{AES128CTR}(\mathbf{s_{public}}, N)$
else
  $\mathbf{x} \leftarrow 0^0$ ▷ x is set to the empty string
  $b \leftarrow 0$
  $n \leftarrow 0$
  while $n < N$ **do**
    $d \leftarrow \mathtt{min}(168, N - n)$
    $\mathbf{x} \leftarrow \mathbf{x} \parallel \mathsf{SHAKE128}(\mathbf{s_{public}} \parallel b_{64}, d)$
      ▷ b_{64} is the 8 byte representation of b, least significant byte first
    $n \leftarrow n + d$
    $b \leftarrow b + 1$
  end
  return x
end

Generate invertible matrices. Let $l \in \{2, 4, 5\}$ and $M \in \mathbb{F}_q^{l \times l}$ any $l \times l$ matrix over $\mathbb{F}_q$. Since the polynomial $\det(M + xS)$ in the variable x has at most l roots, there exists an element a of $\mathbb{F}_q$ such that the matrix $M + aS$ is invertible. We use this property to generate invertible matrices, but only when $r = l$, as follows:

Algorithm 6: Improve public matrices

input : Elements of $\mathbb{F}_q : b[.]$
Desired matrix size $l_1 \times l_2$
output: a matrix M

$i \leftarrow 0$
for $i_1 \leftarrow 0$ **to** $l_1 - 1$ **do**
  for $j_1 \leftarrow 0$ **to** $l_2 - 1$ **do**
    $M[i_1, j_1] = b[i]$
    $i \leftarrow i + 1$
  end
end
if M is a $l \times l$ matrix **and** $\det(M) = 0$ **then**
  for $a \leftarrow 1$ **to** $q - 1$ **do**
    $M \leftarrow M + aS$
    if $\det(M) \neq 0$ **then**
      break
    end
  end
end
return M

Fixed ABQ **matrices.** Beullens has shown in [9] that some public keys of the Round 1 version of SNOVA are weak. This was resolved in our Round 2 update. As an additional countermeasure we fix the seed used to generate the ABQ matrices to some arbitrary but well-chosen value in Round 3.

Algorithm 7: Generate public ABQ

input : -

output: The public matrices $A_{i,\alpha}$, $B_{i,\alpha}$, $Q_{i,\alpha,1}$ and $Q_{i,\alpha,2}$ and the coefficients $q_{i,\alpha,1}, q_{i,\alpha,2}$

$\mathbf{d_q} \leftarrow$ SHAKE256("SNOVA_ABQ")
Convert $\mathbf{d_q}$ to $\mathbb{F}_q$ using **Algorithm** 3
for $i \leftarrow 0$ **to** $o-1$ **do**
  for $\alpha \leftarrow 0$ **to** $\nu - 1$ **do**
    $i_d \leftarrow \nu \cdot i + \alpha$
    Create matrices $A_{i,\alpha} \in \mathbb{F}_q^{r \times r}$ from $d_q[i_d \cdot r^2 \ ... \ i_d \cdot r^2 + r^2 - 1]$ using **Algorithm** 6
    Create matrices $B_{i,\alpha} \in \mathbb{F}_q^{r \times l}$ from $d_q[\nu \cdot o \cdot r^2 + i_d \cdot r \cdot l \ ... \ \nu \cdot o \cdot r^2 + i_d \cdot r \cdot l + r \cdot l - 1]$
    using **Algorithm** 6
    $q_{i,\alpha,1} \leftarrow d_q[\nu \cdot o(r^2 + rl) + i_d \cdot l \ ... \ \nu o(r^2 + r \cdot l) + (\nu i + \alpha)l + l - 1]$
    $q_{i,\alpha,2} \leftarrow d_q[\nu \cdot o(r^2 + rl + l) + i_d \cdot l \ ... \ \nu \cdot o(r^2 + r \cdot l + l) + i_d \cdot l + l - 1]$
    $Q_{i,\alpha,1} \leftarrow \sum_{j=0}^{l-1} q_{i,\alpha,1}[j] \cdot S^j$
    $Q_{i,\alpha,2} \leftarrow \sum_{j=0}^{l-1} q_{i,\alpha,2}[j] \cdot S^j$
  end
end
return $A_{i,\alpha}$, $B_{i,\alpha}$, $Q_{i,\alpha,1}$ and $Q_{i,\alpha,2}$ as well as the coefficients $q_{i,\alpha,1}, q_{i,\alpha,2}$

If all of the l components of $q_{i,\alpha,1}$ or $q_{i,\alpha,2}$ are zero, then the specific i, α combination does not contribute to the public map (1.3). This undesirable situation can be prevented by only considering seeds to SHAKE256 in **Algorithm** 7 that result in all $Q_{i,\alpha,1}$ and $Q_{i,\alpha,2}$ matrices being non-zero. We use the same ASCII string "SNOVA_ABQ" (as bytes) for the generation of the ABQ matrices for all recommended parameter sets. We verified that the matrices $Q_{i,\alpha,1}$ and $Q_{i,\alpha,2}$ are always non-zero with this choice of the seed.

Now we have the ingredients needed to generate $\mathbf{P}_i^{11}$, $\mathbf{P}_i^{12}$ and $\mathbf{P}_i^{21}$ from the public key seed $\mathbf{s_{public}}$. The details differ between a symmetric public matrix and an asymmetric matrix. For this reason two algorithms are provided.

The public key expansion algorithm is given in **Algorithm** 8 or **Algorithm** 9. The order in which the bytes from the XOF are converted to elements of $\mathbb{F}_q$ has not been changed for Round 3. A possible improvement could be to optimize this order. This may result in for instance a reduced memory use on constrained devices. For the Round 3 submission we have chosen to stick to the Round 2 order, but the potential created by other orders will be investigated during Round 3. Note that the changing the order changes the KAT files, but does not impact the security of SNOVA.

Algorithm 8: Generate the random part of public key for an asymmetric public matrix

input : Public key seed $\mathbf{s_{public}}$
output: The matrices $\mathbf{P}_i^{11}, \mathbf{P}_i^{12}, \mathbf{P}_i^{21}$

Generate bytes $\mathbf{b}$ from $\mathbf{s_{public}}$ using **Algorithm** 5
$b \leftarrow \mathbf{b}$ from bytes to $\mathbb{F}_q$ using **Algorithm** 3
$k \leftarrow 0$
for $i \leftarrow 0$ **to** $m_1 - 1$ **do**
  for $n_i \leftarrow 0$ **to** $v - 1$ **do**
    for $n_j \leftarrow 0$ **to** $v - 1$ **do**
      for $i_1 \leftarrow 0$ **to** $l - 1$ **do**
        for $j_1 \leftarrow 0$ **to** $l - 1$ **do**
          $\mathbf{P}_i^{11}[n_i \cdot l + i_1, n_j \cdot l + j_1] = b[k]$
          $k \leftarrow k + 1$
        end
      end
    end
  end
end
for $i \leftarrow 0$ **to** $m_1 - 1$ **do**
  for $n_i \leftarrow 0$ **to** $v - 1$ **do**
    for $n_j \leftarrow 0$ **to** $o - 1$ **do**
      for $i_1 \leftarrow 0$ **to** $l - 1$ **do**
        for $j_1 \leftarrow 0$ **to** $l - 1$ **do**
          $\mathbf{P}_i^{12}[n_i \cdot l + i_1, n_j \cdot l + j_1] = b[k]$
          $k \leftarrow k + 1$
        end
      end
    end
  end
end
for $i \leftarrow 0$ **to** $m_1 - 1$ **do**
  for $n_i \leftarrow 0$ **to** $o - 1$ **do**
    for $n_j \leftarrow 0$ **to** $v - 1$ **do**
      for $i_1 \leftarrow 0$ **to** $l - 1$ **do**
        for $j_1 \leftarrow 0$ **to** $l - 1$ **do**
          $\mathbf{P}_i^{21}[n_i \cdot l + i_1, n_j \cdot l + j_1] = b[k]$
          $k \leftarrow k + 1$
        end
      end
    end
  end
end
return $\mathbf{P}_i^{11}, \mathbf{P}_i^{12}, \mathbf{P}_i^{21}$

Algorithm 9: Generate the random part of public key for a symmetric public matrix

input : Public key seed $\mathbf{s_{public}}$
output: The matrices $\mathbf{P}_i^{11}, \mathbf{P}_i^{12}, \mathbf{P}_i^{21}$

Generate bytes $\mathbf{b}$ from $\mathbf{s_{public}}$ using **Algorithm** 5
$b \leftarrow \mathbf{b}$ from bytes to $\mathbb{F}_q$ using **Algorithm** 3
$k \leftarrow 0$
for $i \leftarrow 0$ **to** $m_1 - 1$ **do**
  for $n_i \leftarrow 0$ **to** $v - 1$ **do**
    for $i_1 \leftarrow 0$ **to** $l - 1$ **do**
      for $j_1 \leftarrow i_1$ **to** $l - 1$ **do**
        $\mathbf{P}_i^{11}[n_i \cdot l + j_1, n_i \cdot l + i_1] = \mathbf{P}_i^{11}[n_i \cdot l + i_1, n_i \cdot l + j_1] = b[k]$
        $k \leftarrow k + 1$
      end
    end
    for $n_j \leftarrow n_i + 1$ **to** $v - 1$ **do**
      for $i_1 \leftarrow 0$ **to** $l - 1$ **do**
        for $j_1 \leftarrow 0$ **to** $l - 1$ **do**
          $\mathbf{P}_i^{11}[n_j \cdot l + j_1, n_i \cdot l + i_1] = \mathbf{P}_i^{11}[n_i \cdot l + i_1, n_j \cdot l + j_1] = b[k]$
          $k \leftarrow k + 1$
        end
      end
    end
    for $n_j \leftarrow 0$ **to** $o - 1$ **do**
      for $i_1 \leftarrow 0$ **to** $l - 1$ **do**
        for $j_1 \leftarrow 0$ **to** $l - 1$ **do**
          $\mathbf{P}_i^{21}[n_j \cdot l + j_1, n_i \cdot l + i_1] = \mathbf{P}_i^{12}[n_i \cdot l + i_1, n_j \cdot l + j_1] = b[k]$
          $k \leftarrow k + 1$
        end
      end
    end
  end
end
return $\mathbf{P}_i^{11}, \mathbf{P}_i^{12}, \mathbf{P}_i^{21}$

Second Step: Compute the matrix $\mathbf{P}_i^{22}$ for $0 \leq i < m_1$. Using the sign convention of this section, $T^{12} = -\mathbf{T}^{12}$, we have

$$\mathbf{F}_i^{12} = \mathbf{P}_i^{12} + \mathbf{P}_i^{11} T^{12}$$
$$\mathbf{P}_i^{22} = -(T^{12})^\top \mathbf{F}_i^{12} - \mathbf{P}_i^{21} T^{12}$$

We include the hash of the public key into the secret key in order to avoid having to recalculate $\mathbf{P}_i^{22}$ when signing. This choice increases the secret key size by 48 bytes.

Algorithm 10: Generate Secret and Public keys

input : Public and secret key seeds ($\mathbf{s_{public}}, \mathbf{s_{secret}}$)
output: Secret key and Public key as bytes

Generate T^{12} using **Algorithm** 4
Generate ($\mathbf{P}_i^{11}, \mathbf{P}_i^{12}$, $\mathbf{P}_i^{21}$ for $0 \leq i < m_1$) using **Algorithm** 8 or 9
for $i \leftarrow 0$ **to** $m_1 - 1$ **do**
    $\mathbf{F}_i^{12} \leftarrow \mathbf{P}_i^{12} + \mathbf{P}_i^{11} T^{12}$
    $\mathbf{P}_i^{22} \leftarrow -\left(T^{12}\right)^{\top} \mathbf{F}_i^{12} - \mathbf{P}_i^{21} T^{12}$
end
$\mathbf{p_c} \leftarrow$ packed $\mathbf{P}_i^{22}$ using **Algorithm** 11
$\mathbf{pk} \leftarrow \mathbf{s_{public}} \,||\, \mathbf{p_c}$
$\mathbf{pk_d} \leftarrow \mathsf{SHAKE256}(\mathbf{pk}, \mathbf{48})$
$\mathbf{sk} \leftarrow \mathbf{s_{public}} \,||\, \mathbf{s_{secret}} || \mathbf{pk_d}$
return sk, **pk**

Algorithm 11: Pack the generated public key as bytes

input : The matrices $\mathbf{P}_i^{22}$
output: The byte representation of P_{22}

$b \leftarrow \{\}$
for $i \leftarrow 0$ **to** $m_1 - 1$ **do**
    for $n_i \leftarrow 0$ **to** $o - 1$ **do**
        if Asymmetric **then**
            ▷ Output the full $\mathbf{P}_i^{22}$
            for $n_j \leftarrow 0$ **to** $o - 1$ **do**
                for $i_1 \leftarrow 0$ **to** $l - 1$ **do**
                    for $j_1 \leftarrow 0$ **to** $l - 1$ **do**
                        $b = b \,||\, \mathbf{P}_i^{22}[n_i \cdot l + i_1, n_j \cdot l + j_1]$
                    end
                end
            end
        else
            ▷ Only the upper diagonal part of $\mathbf{P}_i^{22}$
            for $i_1 \leftarrow 0$ **to** $l - 1$ **do**
                for $j_1 \leftarrow i_1$ **to** $l - 1$ **do**
                    $b = b \,||\, \mathbf{P}_i^{22}[n_i \cdot l + i_1, n_i \cdot l + j_1]$
                end
                for $n_j \leftarrow 0$ **to** $o - 1$ **do**
                    for $j_1 \leftarrow i_1$ **to** $l - 1$ **do**
                        $b = b \,||\, \mathbf{P}_i^{22}[n_i \cdot l + i_1, n_j \cdot l + j_1]$
                    end
                end
            end
        end
    end
end
$\mathbf{p_b} \leftarrow b$ as bytes using **Algorithm** 1
return $\mathbf{p_b}$

1.8.2 Algorithms for Signature Generation

The message hash is computed by the algorithm below. The reordering from d to m_h (line 4) has been included in the Round 3 specification to result in KAT files that are identical to the SNOVA

Round 2 KATs when Round 2 parameters are used. The reordering is not necessary for a correct operation of the algorithms; the step on line 4 can be skipped when backward compatibility of KATs is not required.

Algorithm 12: Get message hash in $\mathbb{F}_q$

input : Message **msg**
Public key digest $\mathbf{pk_d}$
A random **salt**
output: data as elements of $\mathbb{F}_q$

1 $\mathbf{m_d} \leftarrow \mathsf{SHAKE256}(\mathbf{msg}, 64)$
2 $d \leftarrow \mathsf{SHAKE256}(\mathbf{pk_d}||\mathbf{m_d}||\mathbf{salt})$ as $\mathbb{F}_q$ using **Algorithm** 2
3 $m_h[i \cdot l \cdot r + i_1 \cdot r + j_1] \leftarrow d[i \cdot r \cdot l + j_1 \cdot l + i_1]$ ▷ Reorder
4 ▷ $i \in \{0 \ldots o-1\}$, $i_1 \in \{0 \ldots l-1\}, j_1 \in \{0 \ldots r-1\}$
5 Return m_h

Algorithm 13: Expand secret key

input : Public and secret key seeds $(\mathbf{s_{public}}, \mathbf{s_{secret}})$
output: T^{12}, $\mathbf{F}_i^{11}, \mathbf{F}_i^{12}, \mathbf{F}_i^{21}$

1 Generate T^{12} using **Algorithm** 4
2 Generate $(\mathbf{P}_i^{11}, \mathbf{P}_i^{12}, \mathbf{P}_i^{21}$ for $0 \leq i < m_1)$ using **Algorithm** 8 or 9
3 **for** $i \leftarrow 0$ **to** $m_1 - 1$ **do**
4 $\quad \mathbf{F}_i^{11} \leftarrow \mathbf{P}_i^{11}$
5 $\quad \mathbf{F}_i^{12} \leftarrow \mathbf{P}_i^{12} + \mathbf{P}_i^{11} T^{12}$
6 $\quad \mathbf{F}_i^{21} \leftarrow \mathbf{P}_i^{21} + (T^{12})^\top \mathbf{P}_i^{11}$
7 **end**
8 **return** $T^{12}, \mathbf{F}_i^{11}, \mathbf{F}_i^{12}, \mathbf{F}_i^{21}$

Let **msg** be the message to be signed, we randomly choose a **salt** and then generate a signature for the hash value using **Algorithm** 12. Then we randomly assign values to vinegar variables $V_0, \ldots, V_{v-1}$ depending on the number of the sign attempt $\mathbf{num_{sig}}$ as follows

Algorithm 14: Assign values to vinegar variables

input : Message **msg**
Secret key seed $\mathbf{s_{secret}}$
The number of sign attempt $\mathbf{num_{sig}}$ as a singly byte
A random **salt**
output: Vinegar matrix V

1 $\mathbf{m_d} \leftarrow \mathsf{SHAKE256}(\mathbf{msg}, 64)$
2 $\mathbf{v_b} \leftarrow \mathsf{SHAKE256}(\mathbf{s_{secret}} \,||\, \mathbf{m_d} \,||\, \mathbf{salt} \,||\, \mathbf{num_{sig}})$
3 $v_g \leftarrow \mathbf{v_b}$ as $\mathbb{F}_q$ using **Algorithm** 2
4 $V_i[j][k] = v_g[i \cdot l \cdot r + j \cdot r + k]$
5 **return** V

Second, we compute the vinegar part values $F^*_{i,VV}$ of the central map F^*_i for $0 \leq i < o$. Recall that the vinegar part values $F^*_{i,VV}$ of the central map F^*_i is

$$F^*_{i,VV} = \sum_{\alpha=0}^{\nu-1} A_{i,\alpha} \cdot V^\top \cdot \mathbf{Q}^{\otimes v}_{i,\alpha,1} \cdot \mathbf{F}^{11}_{i'} \cdot \mathbf{Q}^{\otimes v}_{i,\alpha,2} \cdot V \cdot B_{i,\alpha}$$

where $i' = (i + \alpha) \mod m_1$, $F_{i',jk}$ are elements from $\mathbb{F}_q^{l \times l}$, $V_j \in \mathbb{F}_q^{l \times r}$, $A_{i,\alpha}$ and $B_{i,\alpha}$ are matrices of $\mathbb{F}_q^{r \times r}$ and $\mathbb{F}_q^{r \times l}$, and $Q_{i,\alpha,1}$, $Q_{i,\alpha,2}$ are invertible matrices randomly chosen from $\mathbb{F}_q[S]$. We will consider V to be vectors in the following instead of using explicit indices. This turns $F^*_{i,VV}$ into a

matrix with respect to j, k. Note that we write the central map $\mathbf{F}_i$ in the form of

$$\mathbf{F}_{i'} = \begin{pmatrix} \mathbf{F}_{i'}^{11} & \mathbf{F}_{i'}^{12} \\ \mathbf{F}_{i'}^{21} & \mathbf{0} \end{pmatrix}.$$

We can now compute the vinegar part values $F_{i,VV}^*$ of the central map F_i^*. We evaluate $F_{i,VV}^*$ in two steps, using that

$$Q_{i,\alpha,1} = \sum_{j=0}^{l-1} q_{i,\alpha,1}[j] \cdot S^j$$

(Q_2 similar) as

$$\begin{aligned} F_{i,VV}^* &= \sum_{\alpha} A_{i,\alpha} \cdot V^\top \cdot \mathbf{Q}_{i,\alpha,1}^{\otimes v} \cdot \mathbf{F}_{i'}^{11} \cdot \mathbf{Q}_{i,\alpha,2}^{\otimes v} \cdot V \cdot B_{i,\alpha} \\ &= \sum_{\alpha,a,b} A_{i,\alpha} \cdot V^\top \cdot q_{i,\alpha,1}[a] \cdot (\mathbf{S}^{\otimes v})^a \cdot \mathbf{F}_{i'}^{11} \cdot (\mathbf{S}^{\otimes v})^b \cdot q_{i,\alpha,2}[b] \cdot V \cdot B_{i,\alpha} \\ &= \sum_{\alpha,a,b} A_{i,\alpha} \cdot q_{i,\alpha,1}[a] \cdot D_{i,a,b,VV}^* \cdot q_{i,\alpha,2}[b] \cdot B_{i,\alpha} \end{aligned}$$

where

$$D_{i,a,b,VV}^* = V^\top \cdot (\mathbf{S}^{\otimes v})^a \cdot \mathbf{F}_{i'}^{11} \cdot (\mathbf{S}^{\otimes v})^b \cdot V \tag{1.5}$$

Algorithm 15: Compute the vinegar part of the central map

input : Secret key $\mathbf{F}_i^{11}$
Matrices $A_{i,\alpha}, B_{i,\alpha}$, and the coefficients $q_{i,\alpha,1}, q_{i,\alpha,2}$
Vinegar matrix V
output: Vectorized vinegar part $f_{i,VV}$

$D_{i',a,b,VV}^* \leftarrow V^\top \cdot (\mathbf{S}^{\otimes v})^a \cdot \mathbf{F}_{i'}^{11} \cdot (\mathbf{S}^{\otimes v})^b \cdot V$ ▷ $i' \in \{1 \dots m_1\}$, $a, b \in \{0 \dots l-1\}$
$F_{i,VV}^* \leftarrow 0$
for $i \leftarrow 0$ **to** $o-1$ **do**
  for $\alpha \leftarrow 0$ **to** $\nu - 1$ **do**
    $i' \leftarrow (i + \alpha) \mod m_1$
    $F_{i,VV}^* \leftarrow F_{i,VV}^* + \sum_{a,b} A_{i,\alpha} \cdot q_{i,\alpha,1}[a] \cdot D_{i',a,b,VV}^* \cdot q_{i,\alpha,2}[b] \cdot B_{i,\alpha}$
  end
end
$f_{i,VV} \leftarrow 0$ ▷ Return matrix as vector
for $i \leftarrow 0$ **to** $o-1$ **do**
  for $j_1 \leftarrow 0$ **to** $l-1$ **do**
    for $i_1 \leftarrow 0$ **to** $r-1$ **do**
      $f_{i,VV} \leftarrow f_{i,VV} \mid\mid F_{i,VV}^*[i_1, j_1]$
    end
  end
end
return $f_{i,VV}$

The resulting system can be seen as a linear system of the oil variables over the finite field $\mathbb{F}_q$. In order to write down this linear system **Algorithm** 15 returns the vectorization of the $F_{i,VV}^*$ matrix. We are now ready to write down the linear system of the oil variables over $\mathbb{F}_q$.

Algorithm 16: Compute the coefficient matrix of the oil variable

input : Secret key parts $\mathbf{F}_{i'}^{12}, \mathbf{F}_{i'}^{21}$
Public matrices $A_{i,\alpha}, B_{i,\alpha}, Q_{i,\alpha,1}, Q_{i,\alpha,2}$
Vinegar values as matrix V
output: The vectorized coefficient matrix G_{ik}

$G \leftarrow \text{matrix}(l \cdot r \cdot o \times l \cdot r \cdot o)$
for $i \leftarrow 0$ **to** $o-1$ **do**
  for $\alpha \leftarrow 0$ **to** $\nu - 1$ **do**
    $i' \leftarrow (i+\alpha) \mod m_1$
    $H_1 \leftarrow \mathbf{Q}_{i,\alpha,1}^{\otimes o} \cdot \mathbf{F}_{i'}^{21} \cdot \mathbf{Q}_{i,\alpha,2}^{\otimes v} \cdot V \cdot B_{i,\alpha}$
    $H_2 \leftarrow A_{i,\alpha} \cdot V^\top \cdot \mathbf{Q}_{i,\alpha,1}^{\otimes v} \cdot \mathbf{F}_{i'}^{12} \cdot \mathbf{Q}_{i,\alpha,2}^{\otimes o}$
    $G[i \cdot r \cdot l + i_1 \cdot r + i_2][i_0 \cdot l \cdot r + j_1 \cdot r + j_2]$
      $\leftarrow G[i \cdot r \cdot l + i_1 \cdot r + i_2][i_0 \cdot l \cdot r + j_1 \cdot r + j_2]$
        $+H_1[i_0 \cdot l + j_1][i_1] \cdot A_{i,\alpha}[i_2][j_2] + H_2[i_2, i_0 \cdot l + j_1] \cdot B_{i,\alpha}[j_2][i_1]$
        $\triangleright\ i_0 \in \{0 \dots o-1\},\ i_1, j_1 \in \{0 \dots l-1\},\ i_2, j_2 \in \{0 \dots r-1\}$
  end
end
return G

Algorithm 17: Sign message

input : Secret key **sk**
Message **msg**
A random **salt**
output: the signature **sig**

$\mathbf{s_{public}} \leftarrow \mathbf{sk}[0, \dots, 15]$
$\mathbf{s_{secret}} \leftarrow \mathbf{sk}[16, \dots, 47]$
$\mathbf{pk_d} \leftarrow \mathbf{sk}[48, \dots, 95]$
Generate T^{12}, $\mathbf{F}_{i'}^{11}, \mathbf{F}_{i'}^{12}, \mathbf{F}_{i'}^{21}$ using **Algorithm** 13
$T^{-1} \leftarrow \begin{pmatrix} I^{11} & T^{12} \\ 0 & I^{22} \end{pmatrix}$
Generate $A_{i,\alpha}$, $B_{i,\alpha}$, $Q_{i,\alpha,1}$ and $Q_{i,\alpha,2}$ using **Algorithm** 7
Get h_{msg} from **msg**, $\mathbf{pk_d}$, and **salt** using **Algorithm** 12
$\mathbf{num_{sig}} \leftarrow 0$
repeat
  flag_redo $\leftarrow$ **FALSE**
  $\mathbf{num_{sig}} \leftarrow \mathbf{num_{sig}} + 1$
  if $\mathbf{num_{sig}} = 255$ **then**
    return FAIL
  end
  Assign vinegar values $(X_0, \dots, X_{v-1})$ using **Algorithm** 14
  Compute $f_{i,VV}$ using **Algorithm** 15
  $M_{vec} \leftarrow h_{msg} - f_{i,VV}$
  Compute G using **Algorithm** 16
  if $\det(G) = 0$ **then**
    flag_redo $\leftarrow$ **TRUE**
  else
    $(\widetilde{X}_0, \widetilde{X}_1, \dots, \widetilde{X}_{o-1}) \leftarrow G^{-1} M_{vec}$
    $(\widetilde{X}_0, \widetilde{X}_1, \dots, \widetilde{X}_{n-1}) \leftarrow T^{-1}(X_0, \dots, X_{v-1}, \widetilde{X}_0, \dots, \widetilde{X}_{o-1})^\top$
  end
until flag_redo = **FALSE**;
$\mathbf{sig}_b \leftarrow (\widetilde{X}_0, \widetilde{X}_1, \dots, \widetilde{X}_{n-1})$ as bytes using **Algorithm** 18
return $\mathbf{sig}_b \,||\, \mathbf{salt}$

Algorithm 18: Pack the generated signature as bytes

input : The matrices $(\widetilde{X}_0, \widetilde{X}_1, \ldots, \widetilde{X}_{n-1})$
output: The byte representation $\mathbf{sig}_b$

$b \leftarrow \{\}$
for $i \leftarrow 0$ **to** $n-1$ **do**
  for $i_1 \leftarrow 0$ **to** $l-1$ **do**
    for $j_1 \leftarrow 0$ **to** $r-1$ **do**
      $b \leftarrow b \mid\mid \widetilde{X}_i[i_1, j_1]$
    end
  end
end
$\mathbf{sig}_b \leftarrow b$ as bytes using **Algorithm** 1
return $\mathbf{sig}_b$

1.8.3 Algorithms for Signature Verification

Algorithm 19: Expand public key

input : Public key part $\mathbf{pk}[16, \ldots]$
output: Public matrix part $\mathbf{P}_i^{22}$

$b \leftarrow$ from $\mathbf{pk}[16, ...]$ using **Algorithm** 2
$k \leftarrow 0$
if Asymmetric **then**
  ▷ Convert the full $\mathbf{P}_i^{22}$
  for $i \leftarrow 0$ **to** $m_1 - 1$ **do**
    for $n_i \leftarrow 0$ **to** $o - 1$ **do**
      for $n_j \leftarrow 0$ **to** $o - 1$ **do**
        for $i_1 \leftarrow 0$ **to** $l - 1$ **do**
          for $j_1 \leftarrow 0$ **to** $l - 1$ **do**
            $\mathbf{P}_i^{22}[n_i \cdot l + i_1, n_j \cdot l + j_1] = b[k]$
            $k \leftarrow k + 1$
          end
        end
      end
    end
  end
else
  ▷ The upper diagonal part of $\mathbf{P}_i^{22}$ is provided, symmetrize
  for $m_i \leftarrow 0$ **to** $m_1 - 1$ **do**
    for $n_i \leftarrow 0$ **to** $o - 1$ **do**
      for $i_1 \leftarrow 0$ **to** $l - 1$ **do**
        for $j_1 \leftarrow i_1$ **to** $l - 1$ **do**
          $\mathbf{P}_i^{22}[n_i \cdot l + i_1, n_i \cdot l + j_1] = \mathbf{P}_i^{22}[n_i \cdot l + j_1, n_i \cdot l + i_1] = b[k]$
          $k \leftarrow k + 1$
        end
        for $n_j \leftarrow 0$ **to** $o - 1$ **do**
          for $j_1 \leftarrow i_1$ **to** $l - 1$ **do**
            $\mathbf{P}_i^{22}[n_i \cdot l + i_1, n_j \cdot l + j_1] = \mathbf{P}_i^{22}[n_j \cdot l + j_1, n_i \cdot l + i_1] = b[k]$
            $k \leftarrow k + 1$
          end
        end
      end
    end
  end
end
return $\mathbf{P}_i^{22}$

Recall that the public map $\mathcal{P}^* = \left(\mathcal{P}_0^*, \ldots, \mathcal{P}_{m-1}^*\right) : (\mathbb{F}_q^{l \times r})^n \to (\mathbb{F}_q^{r \times l})^o$, where $i' = (i + \alpha) \mod m_1$ and

$$\mathcal{P}_i^*(\mathbf{U}) = \sum_{\alpha=0}^{\nu-1} \sum_{d_j=0}^{n-1} \sum_{d_k=0}^{n-1} A_{i,\alpha} \cdot U_{d_j}^\top \cdot \left(Q_{i,\alpha,1} \cdot P_{i',d_j d_k} \cdot Q_{i,\alpha,2}\right) \cdot U_{d_k} \cdot B_{i,\alpha}$$

with the variable $\mathbf{U} = (U_0, \ldots, U_{n-1})^\top$. For the signature verification, we write the signature $\mathbf{sig} = (U_0, \ldots, U_{n-1})^\top \in \mathbb{F}_q^{l \times r}$. The algorithm for evaluating the public map at a signature $\mathbf{sig}$ follows below. In the verification we use a two-step evaluation of $\mathcal{P}_i^*$ similar to the two-step procedure described near equation 1.5.

Algorithm 20: Expand signature

input : Signature **sig**
output: signature matrices $(\widetilde{X}_0, \widetilde{X}_1, \ldots, \widetilde{X}_{n-1})$

$s \leftarrow$ Expand **sig**$[0, \ldots, |\mathbf{sig}| - 16]$ using **Algorithm** 2 while checking for illegal input.
for $i \leftarrow 0$ **to** $n-1$ **do**
  for $i_1 \leftarrow 0$ **to** $l-1$ **do**
    for $j_1 \leftarrow 0$ **to** $r-1$ **do**
      $\widetilde{X}_i[i_1, j_1] \leftarrow s[i \cdot l \cdot r + i_1 \cdot r + j_1]$
    end
  end
end
return $(\widetilde{X}_0, \widetilde{X}_1, \ldots, \widetilde{X}_{n-1})$

Algorithm 21: Verify signature of message

input : The public key **pk**
Message **msg**
The signature **sig**
output: **Accept** or **Reject**

$\mathbf{pk_d} \leftarrow \mathsf{SHAKE256}(\mathbf{pk}, \mathbf{48})$
Get the signature matrices $(\widetilde{X}_0, \widetilde{X}_1, \ldots, \widetilde{X}_{n-1})$ from **sig** using **Algorithm** 20
salt $\leftarrow$ **sig**$[|\mathbf{sig}| - 16, \ldots, |\mathbf{sig}|]$ ▷ Last 16 bytes of the signature **sig**
$\mathbf{s_{public}} \leftarrow \mathbf{pk}[0, \ldots, 15]$
Get $A_{i,\alpha}$, $B_{i,\alpha}$, $Q_{i,\alpha,1}$, $Q_{i,\alpha,2}$, using **Algorithm** 7
Get $\mathbf{P}^{11}_{i'}$, $\mathbf{P}^{12}_{i'}$, $\mathbf{P}^{21}_{i'}$ from $\mathbf{s_{public}}$ using **Algorithm** 8 or 9
Get $\mathbf{P}^{22}_{i'}$ from **pk**$[16, \ldots]$ using **Algorithm** 19
$\mathbf{P}_{i'} \leftarrow \begin{pmatrix} \mathbf{P}^{11}_{i'} & \mathbf{P}^{12}_{i'} \\ \mathbf{P}^{21}_{i'} & \mathbf{P}^{22}_{i'} \end{pmatrix}$
$D^*_{i',a,b} \leftarrow X^\top \cdot (\mathbf{S}^{\otimes n})^a \cdot \mathbf{P}_{i'} \cdot (\mathbf{S}^{\otimes n})^b \cdot X$ ▷ $i' \in \{0, \ldots, m_1 - 1\}$, $a, b \in \{0, \ldots, l-1\}$
$\mathcal{P}^*_i \leftarrow 0$
for $i \leftarrow 0$ **to** $o-1$ **do**
  for $\alpha \leftarrow 0$ **to** $\nu - 1$ **do**
    $i' \leftarrow (i + \alpha) \mod m_1$
    $\mathcal{P}^*_i \leftarrow \mathcal{P}^*_i + \sum_{a,b} A_{i,\alpha} \cdot q_{i,\alpha,1}[a] \cdot D^*_{i',a,b} \cdot q_{i,\alpha,2}[b] \cdot B_{i,\alpha}$
  end
end
$\mathbf{hash_s}[i \cdot l \cdot r + j \cdot r + k] \leftarrow \mathcal{P}^*_i[k, j]$
▷ $i \in \{0, \ldots, o-1\}$, $j \in \{0, \ldots, l-1\}, k \in \{0, \ldots, r-1\}$
Get hash $\mathbf{hash_d}$ from **msg**, $\mathbf{pk_d}$, and **salt** using **Algorithm** 12
return Accept if $\mathbf{hash_s} = \mathbf{hash_d}$ **else return Reject**

1.9 Parameters Settings

In this section, we propose our parameters aiming at three security levels in the new call of NIST PQC project [40] levels I, III and V, respectively.

1.9.1 List of Our Parameters

In Table 5 we present our recommended parameters. While parameter sets with a symmetric public matrix may be of interest, the cryptanalysis has not matured to a level where such sets could be proposed for standardization.

Table 5. Recommended parameter sets. The sizes are reported in bytes. All recommended parameter sets use asymmetric public matrices. K: smaller public Key, B: Balanced public key and signature, S: smaller Signature. The secret key size |**sk**| is 96 bytes for all parameter sets.

SL	Name	$(v, o, q, l, r, m_1, m_2)$	|**pk**|	|**sig**|	|**pk**|+|**sig**|
I	SNOVA_I_K	$(29, 3, 16, 4, 8, 5, 96)$	376	528	904
	SNOVA_I_B	$(27, 4, 16, 4, 6, 5, 96)$	656	388	1044
	SNOVA_I_S	$(27, 5, 16, 4, 4, 5, 80)$	1016	272	1288
III	SNOVA_III_K	$(38, 4, 16, 4, 8, 7, 128)$	912	688	1600
	SNOVA_III_B	$(38, 5, 16, 4, 6, 7, 120)$	1416	532	1948
	SNOVA_III_S	$(38, 6, 16, 4, 5, 7, 120)$	2032	456	2488
V	SNOVA_V_K	$(40, 4, 16, 5, 8, 6, 160)$	1216	896	2112
	SNOVA_V_B	$(40, 5, 16, 5, 6, 6, 150)$	1891	691	2582
	SNOVA_V_S	$(40, 6, 16, 5, 5, 6, 150)$	2716	591	3307

For Security Levels I and III, we consistently set $l = 4$ to maximize performance gains, as more effective optimization techniques are available for this parameter choice. For Security Level V, we consistently set $l = 5$ to minimize public-key and signature sizes while maintaining good computational performance. This choice achieves a favorable balance between efficiency and compactness.

1.9.2 Alternative Parameter Sets

SNOVA supports a broad range of parameter choices. Table 6 presents several possible alternatives to the proposed parameter sets. These alternatives are secure against currently known attacks, although their security margins are less conservative than those of the recommended parameter sets. If no new effective attacks emerge during Round 3 and further cryptanalytic study increases our confidence in SNOVA's security, some of these alternatives may offer more attractive trade-offs in terms of key and signature sizes and computational performance.

We also provide a number of alternative parameter sets with $l = 2$, which feature relatively large public keys but compact signatures. In certain deployment scenarios, a fixed key pair may be used to generate a very large number of signatures, making signature size substantially more important than key size. Traditionally, UOV-type schemes have been particularly well suited to such settings. However, with the proposed $l = 2$ parameter sets, SNOVA offers an even more attractive trade-off, outperforming UOV in some of these signature-intensive applications.

Table 6. Alternative SNOVA parameter sets. The sizes are reported in bytes. All alternative parameter sets use asymmetric public matrices. The secret key size |**sk**| is 96 bytes for all parameter sets.

SL	$(v, o, q, l, r, m_1, m_2)$	|**pk**|	|**sig**|	|**pk**|+|**sig**|	Security
I	$(27, 3, 16, 4, 7, 5, 84)$	376	436	812	169
	$(27, 4, 16, 4, 5, 5, 80)$	656	326	982	177
	$(27, 4, 19, 4, 6, 5, 96)$	699	413	1112	201
	$(27, 5, 13, 4, 4, 5, 80)$	950	255	1205	170
	$(50, 17, 16, 2, 2, 17, 68)$	9842	150	9992	158
III	$(27, 4, 16, 5, 5, 4, 100)$	816	404	1220	221
	$(33, 5, 16, 4, 5, 6, 100)$	1216	396	1612	221
	$(39, 5, 11, 4, 6, 7, 120)$	1241	478	1719	230
	$(38, 5, 19, 4, 6, 7, 120)$	1510	567	2077	269
	$(76, 25, 16, 2, 2, 25, 100)$	31266	218	31484	223
V	$(40, 5, 19, 5, 6, 6, 150)$	2016	736	2752	331
	$(47, 6, 19, 4, 6, 9, 144)$	2781	695	3476	318
	$(104, 33, 16, 2, 2, 33, 132)$	71890	290	72180	287

1.9.3 Note On the Parameters of SNOVA

Round 3 SNOVA has seven parameters: $(v, o, q, l, r, m_1, m_2)$.

In general, a smaller value of o tends to provide stronger security (against key recovery attacks) for UOV-like schemes. In Round 3 SNOVA, the parameter l represents the size of the matrix ring used in SNOVA, larger values of l allow for a higher degree of public key compression. In both Round 1 and Round 2, we proposed only parameter sets with $q = 16$. During the Round 2 evaluation, subsequent analyses [47, 29, 22] of the UOV family highlighted the importance of parameter sets defined over odd characteristics. Accordingly, in the Round 3 submission, we introduce (alternative) parameter sets over odd characteristic fields as a response to the suggestion in the NIST IR 8610 Round 2 Status Report on the Second Round of the Additional Digital Signature Schemes for the NIST Post-Quantum Cryptography Standardization Process [1].

In the Round 2 SNOVA, the parameter r was always chosen to be equal to the matrix ring size l. In this round, we introduce rectangular signatures, which are equivalent to allowing the whipping dimension r in the whipping transformation to be an independent parameter distinct from l. In our Round 3 submission, our parameter selection we always choose $m_2 = rlo$. However, it is important to note that m_2 does not necessarily need to equal rlo.

By separating o from m_1, and similarly distinguishing r from l, we obtain substantially greater flexibility in parameter selection. Under this framework, the flexibility and feasibility of resisting a wide range of attacks through parameter adjustments are significantly enhanced.

2 Performance Analysis (2.B.2)

2.1 Time

The official implementation of SNOVA [50] contains a reference, optimized, AVX2 optimized and GFNI optimized version. The GFNI version is the fastest. The optimized version is entirely hand-crafted. The AVX2 version is based on the optimized version and was further optimized with AI-assisted development (Claude Code).

Below we report performance numbers for both the AVX2 and the GFNI versions.

Table 7. AVX2. SNOVA performance numbers on a modern desktop system (Arrow Lake, Intel(R) Core(TM) Ultra 7 265K at 3.8GHz, compiler: gcc 16.1.1, Turbo Boost: disabled) for the AVX2 optimized version. The performance is the median number of CPU cycles over 2048 benchmark runs.

SL	Variant	|**pk**|	|**sk**|	|**sig**|	KeyGen	Sign	Verify
I	SNOVA_I_K	376	96	528	187,973	679,577	196,547
	SNOVA_I_K_AES	376	96	528	115,520	608,193	125,406
	SNOVA_I_B	656	96	388	201,138	644,571	178,763
	SNOVA_I_B_AES	656	96	388	133,370	576,904	111,339
	SNOVA_I_S	1016	96	272	243,117	525,243	176,473
	SNOVA_I_S_AES	1016	96	272	171,859	453,782	105,698
III	SNOVA_III_K	912	96	688	487,723	1,455,292	418,243
	SNOVA_III_K_AES	912	96	688	314,118	1,279,004	243,216
	SNOVA_III_B	1416	96	532	575,506	1,459,126	419,616
	SNOVA_III_B_AES	1416	96	532	394,486	1,266,776	238,574
	SNOVA_III_S	2032	96	456	679,299	1,557,053	433,881
	SNOVA_III_S_AES	2032	96	456	491,167	1,361,225	244,463
V	SNOVA_V_K	1216	96	896	1,246,700	3,491,655	679,375
	SNOVA_V_K_AES	1216	96	896	987,835	3,229,305	422,812
	SNOVA_V_B	1891	96	691	1,505,245	3,561,227	782,471
	SNOVA_V_B_AES	1891	96	691	1,238,628	3,290,536	514,064
	SNOVA_V_S	2716	96	591	1,788,492	3,864,611	859,675
	SNOVA_V_S_AES	2716	96	591	1,504,120	3,582,743	581,273

Table 8. GFNI. SNOVA performance numbers on a modern desktop system (Arrow Lake, Intel(R) Core(TM) Ultra 7 265K at 3.8GHz, compiler: gcc 16.1.1, Turbo Boost: disabled) for the GFNI optimized version. The performance is the median number of CPU cycles over 2048 benchmark runs.

SL	Variant	$\|\mathbf{pk}\|$	$\|\mathbf{sk}\|$	$\|\mathbf{sig}\|$	KeyGen	Sign	Verify
I	SNOVA_I_K	376	96	528	159,618	584,188	166,666
	SNOVA_I_K_AES	376	96	528	87,169	512,004	94,164
	SNOVA_I_B	656	96	388	170,468	557,746	141,919
	SNOVA_I_B_AES	656	96	388	102,661	489,983	74,376
	SNOVA_I_S	1016	96	272	204,195	397,921	149,024
	SNOVA_I_S_AES	1016	96	272	132,758	325,418	78,305
III	SNOVA_III_K	912	96	688	416,490	1,240,034	349,462
	SNOVA_III_K_AES	912	96	688	243,311	1,061,262	175,688
	SNOVA_III_B	1416	96	532	484,626	1,226,638	335,912
	SNOVA_III_B_AES	1416	96	532	303,622	1,045,116	156,118
	SNOVA_III_S	2032	96	456	562,069	1,333,170	356,953
	SNOVA_III_S_AES	2032	96	456	372,165	1,141,612	167,684
V	SNOVA_V_K	1216	96	896	811,680	2,573,021	684,735
	SNOVA_V_K_AES	1216	96	896	548,998	2,301,728	424,527
	SNOVA_V_B	1891	96	691	955,444	2,470,611	679,221
	SNOVA_V_B_AES	1891	96	691	682,823	2,192,284	409,444
	SNOVA_V_S	2716	96	591	1,101,953	2,619,986	749,582
	SNOVA_V_S_AES	2716	96	591	823,154	2,341,503	465,939

2.2 Space

The sizes for the SNOVA scheme are determined by the following expressions:

Public key size. The size of the public key of asymmetric SNOVA in bytes is

$$|\mathbf{pk}| = \left\lceil m_1 \cdot o^2 \cdot l^2 \cdot \frac{N_b}{N_q} \right\rceil + 16$$

Secret key size. The size of the secret key in bytes is

$$|\mathbf{sk}| = 96.$$

Signature size. The size of a signature of SNOVA scheme in bytes is

$$|\mathbf{sig}| = \left\lceil r \cdot l \cdot n \cdot \frac{N_b}{N_q} \right\rceil + 16$$

Length of seeds and salt. The length of the public key seed $|\mathbf{s_{public}}| = 16$, the secret key seed $|\mathbf{s_{secret}}| = 32$ and the salt $|\mathbf{salt}| = 16$ for all parameters, as specified in Section 1.6

3 Known Answer Test values (2.B.3)

The submission includes Known Answer Tests (KAT) files of the SNOVA signature scheme. We have provided scripts to generate the KAT files in the submission package. The Round 3 SNOVA KAT files can also be downloaded at https://github.com/PQCLAB-SNOVA/SNOVA_KAT.

The first 24 bytes of the SHAKE256 digests of the KAT files are:

```
SNOVA_I_B.req  c0481e4b408461581c1bc4b66958fafe3be7c0f8af500700
SNOVA_I_B.rsp  0ba77e8b6c1ef99ef1f83d2bd555340bbe77b728f61b4d92
SNOVA_I_B_AES.req  c0481e4b408461581c1bc4b66958fafe3be7c0f8af500700
SNOVA_I_B_AES.rsp  0ced987eb8e00f66cfa748330b4efd05459de690c4c33886
SNOVA_I_K.req  c0481e4b408461581c1bc4b66958fafe3be7c0f8af500700
SNOVA_I_K.rsp  95b82ec747f3079445669640cfde77494d79d8e16cd2d7da
SNOVA_I_K_AES.req  c0481e4b408461581c1bc4b66958fafe3be7c0f8af500700
SNOVA_I_K_AES.rsp  506fa91805b967758c5dc8a29dfb5c5c5073de22676022b6
SNOVA_I_S.req  c0481e4b408461581c1bc4b66958fafe3be7c0f8af500700
SNOVA_I_S.rsp  37b863b07e411417e86dc9c4f81bbff4521d0b20d6b56f57
SNOVA_I_S_AES.req  c0481e4b408461581c1bc4b66958fafe3be7c0f8af500700
SNOVA_I_S_AES.rsp  539eb29c73c5f0b173c8e77ece63e5857803e652f3c6532e

SNOVA_III_B.req  c0481e4b408461581c1bc4b66958fafe3be7c0f8af500700
SNOVA_III_B.rsp  92b2b898126365f1dc376da37b304348125779131ddee4a6
SNOVA_III_B_AES.req  c0481e4b408461581c1bc4b66958fafe3be7c0f8af500700
SNOVA_III_B_AES.rsp  5704ba832507f6614af6ada5901720f18c12aeb44dfd3da6
SNOVA_III_K.req  c0481e4b408461581c1bc4b66958fafe3be7c0f8af500700
SNOVA_III_K.rsp  3661057e284978c63f406a933d930e9b2a78271535f3091b
SNOVA_III_K_AES.req  c0481e4b408461581c1bc4b66958fafe3be7c0f8af500700
SNOVA_III_K_AES.rsp  8a39e4377dfc64d3265840341124a3a791810f50c5f72130
SNOVA_III_S.req  c0481e4b408461581c1bc4b66958fafe3be7c0f8af500700
SNOVA_III_S.rsp  fcec3ed2bf3e6038ae77792edfdabde1b15dad96c4ca971c
SNOVA_III_S_AES.req  c0481e4b408461581c1bc4b66958fafe3be7c0f8af500700
SNOVA_III_S_AES.rsp  6bd0787f2c2b8dce837471a4879e4619c621f6e288858ed0

SNOVA_V_B.req  c0481e4b408461581c1bc4b66958fafe3be7c0f8af500700
SNOVA_V_B.rsp  40ae65e72d84a4a62a1cc3d0f4b021923d06aadfb2d61bc5
SNOVA_V_B_AES.req  c0481e4b408461581c1bc4b66958fafe3be7c0f8af500700
SNOVA_V_B_AES.rsp  fc88cf1dc37d9a223aa6745f09b894c7e54cf50ba9bd0c47
SNOVA_V_K.req  c0481e4b408461581c1bc4b66958fafe3be7c0f8af500700
SNOVA_V_K.rsp  08cd3a932567ec362eb352c42fadb81e808c747b1f4d4fdd
SNOVA_V_K_AES.req  c0481e4b408461581c1bc4b66958fafe3be7c0f8af500700
SNOVA_V_K_AES.rsp  895c5ed8afc5e65d93c6d481115c938a6c66dcef2b7032fd
SNOVA_V_S.req  c0481e4b408461581c1bc4b66958fafe3be7c0f8af500700
SNOVA_V_S.rsp  a3ced007ba19d42abb0a5bf62ae7ff7527a1d669f8eb8aaf
SNOVA_V_S_AES.req  c0481e4b408461581c1bc4b66958fafe3be7c0f8af500700
SNOVA_V_S_AES.rsp  a0c25c165f0e4066510f0b669635ba86f47684599b290e23
```

While the SNOVA parameter space has been expanded and the recommended parameters have changed, the scheme itself has not changed between rounds 2 and 3. The Round 2 KAT files can still be created by using appropriate parameters. See the official SNOVA repository [50] for an example.

4 Expected Security Strength (2.B.4)

We give the expected security strength of our parameters aiming at three security levels in the new call of the NIST PQC project [40], security levels I, III and V, respectively.

4.1 Security Strength

The estimated complexities of the SNOVA parameter sets against the known attacks discussed in Section 5 are presented in the following tables. For each parameter set, the lowest complexity among all known attacks is highlighted in **bold** font. The dash symbol "-" indicates the attack is not applicable.

Table 9. Security estimates of SNOVA security level I parameter sets.

Complexity	SNOVA_I_K	SNOVA_I_B	SNOVA_I_S
Comp_{Dir}	**196**	214	183
Comp_{For}	197	**197**	193
Comp_{Col}	214	214	**182**
Comp_{KS}	421	373	357
Comp_{Rec}	332	300	300
Comp_{Int}	507	436	411
$\text{Comp}_{\text{RecMinRank}}$	-	-	-
$\text{Comp}_{\text{Wedge}}$	-	-	-
$\text{Comp}_{\text{TrunRec}}$	-	440	313
$\text{Comp}_{\text{TrunInt}}$	516	444	420

Table 10. Security estimates of SNOVA security level III parameter sets.

Complexity	SNOVA_III_K	SNOVA_III_B	SNOVA_III_S
Comp_{Dir}	272	**262**	267
Comp_{For}	**266**	263	266
Comp_{Col}	279	263	**263**
Comp_{KS}	549	533	517
Comp_{Rec}	414	414	414
Comp_{Int}	643	615	589
$\text{Comp}_{\text{RecMinRank}}$	-	-	-
$\text{Comp}_{\text{Wedge}}$	-	-	-
$\text{Comp}_{\text{TrunRec}}$	798	603	427
$\text{Comp}_{\text{TrunInt}}$	647	-	-

Table 11. Security estimates of SNOVA security level V parameter sets.

Complexity	SNOVA_V_K	SNOVA_V_B	SNOVA_V_S
$\text{Comp}_{\mathsf{Dir}}$	**341**	325	329
$\text{Comp}_{\mathsf{For}}$	344	338	340
$\text{Comp}_{\mathsf{Col}}$	344	**323**	**323**
$\text{Comp}_{\mathsf{KS}}$	725	705	685
$\text{Comp}_{\mathsf{Rec}}$	528	528	528
$\text{Comp}_{\mathsf{Int}}$	836	802	769
$\text{Comp}_{\mathsf{RecMinRank}}$	-	-	-
$\text{Comp}_{\mathsf{Wedge}}$	-	-	-
$\text{Comp}_{\mathsf{TrunRec}}$	1003	758	540
$\text{Comp}_{\mathsf{TrunInt}}$	-	-	-

5 Analysis of Known Attacks (2.B.5)

In this section, we analyze the security of the SNOVA scheme against known attacks and present the corresponding complexity estimates.

5.1 Preliminaries

5.1.1 MQ Solver and Complexity Estimation

We give a brief overview of the methods used to estimate the complexity of solving an MQ problem. In this specification, we use $\mathsf{MQ}(N, M, q)$ to denote the complexity of solving the MQ system of M equations in N variables over $\mathbb{F}_q$.

Overdetermined case ($N \leq M$). The complexity of solving M homogeneous quadratic equations in N variables can be estimated by

$$3 \cdot \binom{N-1+d_{reg}}{d_{reg}}^2 \cdot \binom{N+1}{2}$$

field multiplications. The term d_{reg} is determined by the smallest positive integer d such that the coefficient of t^d term in the series generated by

$$\frac{(1-t^2)^M}{(1-t)^N}$$

is non-positive. In the literature, several algebraic algorithms, such as F_4 [20], F_5 [21] and XL [15] have been proposed for solving MQ systems. The hybrid approach [8] randomly guesses k variables with $0 \leq k \leq N$ before solving the MQ system and the corresponding complexity is

$$\min_k \; q^k \cdot 3 \cdot \binom{N-k-1+d_{reg,k}}{d_{reg,k}}^2 \cdot \binom{N-k+1}{2}$$

field multiplications where $d_{reg,k}$ is the smallest positive integer d such that the coefficient of t^d term in the series generated by

$$\frac{(1-t^2)^M}{(1-t)^{N-k}}$$

is non-positive.

Underdetermined case ($N > M$). For the underdetermined case, several methods have been proposed to solve underdetermined MQ systems more efficiently. Among these, the method of [26] is generally the most effective. Accordingly, for attacks that involve underdetermined MQ systems, we have considered Hashimoto's algorithm in our complexity estimates. The complexity estimation formula of Hashimoto's algorithm [26] is

$$\min_{k,\alpha} \; (M-\alpha-k+1) \cdot \mathsf{MQ}(\alpha, \alpha, q) + q^k \Big(\mathsf{MQ}(\alpha-1, \alpha-1, q) + \mathsf{MQ}(M-\alpha-k, M-\alpha, q) \Big)$$

provided that $N \geq \max\{(\alpha+1)(M-k-\alpha+1, \alpha(M-k)-(\alpha-1)^2+k\}$ holds.

Recently, two algorithms for solving underdetermined MQ systems have been proposed by Asanuma *et al.* [3] and Ostuzzi [41], respectively. These algorithms build upon Hashimoto's algorithm, extending and further optimizing its framework. It is worth noting that, in essence, the resulting computational complexities of these two approaches are comparable to that of Hashimoto's original algorithm. In the complexity estimation of the direct attack, we consider all three methods of solving underdetermined MQ system [26, 3, 41] for complexity estimation and adopt the minimum complexity among them as the security estimate against direct attacks.

5.2 Forgery Attacks

5.2.1 Direct Attack

In the plain direct attack, the attacker attempts to forge a signature by solving the system

$$\mathcal{P}^*(\mathbf{U}) = \mathsf{Hash}(\mathbf{msg}\|\mathbf{salt}) = \mathbf{z} \in \mathbb{F}_q^{m_2}.$$

This yields a set of m_2 multivariate quadratic equations in $rln + 1$ variables. Our complexity estimates are based on the known algorithms for solving underdetermined MQ systems [26, 35, 3, 41]. The complexity of the plain direct attack is given by

$$\mathrm{Comp}_{\mathsf{Dir}} = \mathsf{MQ}(rln + 1, m_2, q) \cdot g(q)$$

gates.

Table 12. The complexity estimates of the direct attack.

SL	Name	Hashimoto's algorithm [26]	Just Guess algorithm [35]	Variable Partitioning [3]	Pseudo-Oil Subspaces [41]
I	SNOVA_I_K	201	196	198	201
	SNOVA_I_B	214	228	214	214
	SNOVA_I_S	183	192	183	183
III	SNOVA_III_K	272	296	272	272
	SNOVA_III_B	262	296	262	262
	SNOVA_III_S	267	308	267	267
V	SNOVA_V_K	341	393	341	341
	SNOVA_V_B	325	388	325	325
	SNOVA_V_S	329	404	329	329

5.2.2 Forgery attack with MinRank

Attack. In [9], Beullens proposed a forgery attack with MinRank to find a fake signature. This attack attempts to forge a signature by solving for $\mathbf{U}$ such that the columns $\mathbf{u}_j = \mathbf{R}_j^{\otimes n}\mathbf{u}_0 + \mathbf{v}_j$ where $\mathbf{v}_j \in \mathbb{F}_q^{ln}$ and $\mathbf{R}_j \in \mathbb{F}_q[S]$ are randomly chosen for all $j \in \{1, \ldots, r-1\}$. Then, the quadratic part of public map $\mathcal{P}^*(\mathbf{U})$ is

$$\mathbf{E}_{\mathbf{R}} \cdot \mathcal{B}(\mathbf{u}_0, \mathbf{u}_0),$$

where $\mathbf{E}_{\mathbf{R}} \in \mathbb{F}_q^{m_2 \times l^2 m_1}$. The attack is divided into two steps:

- Find a set of $\mathbf{R}_1, \ldots \mathbf{R}_{r-1}$ so that the matrix $\mathbf{E}_{\mathbf{R}}$ has rank drop. This involves a generalized MinRank problem with $l(r-1)$ variables (the coefficients of $\mathbf{R}_1, \ldots \mathbf{R}_{r-1}$). We approximate the distribution of $R := \mathrm{rank}(\mathbf{E}_{\mathbf{R}})$ by that of a uniformly random matrix in $\mathbb{F}_q^{m_2 \times l^2 m_1}$. At the end of this subsection, we provide experimental evidence to validate this approximation. Under the approximation, for a uniformly random matrix in $\mathbb{F}_q^{m_2 \times l^2 m_1}$, the probability of such a matrix has rank R with $1 \leq R < \min(m_2, l^2 m_1)$ is about

$$\frac{q+1}{q} \cdot q^{-(l^2 m_1 - R)(m_2 - R)}.$$

  Note that we are looking for a matrix with rank R in a family of $q^{l(r-1)}$ matrices. Therefore, if we model the matrix $\mathbf{E}_{\mathbf{R}}$ in the family as independent random matrices, then heuristically the expected number of matrices of rank R, is approximately

$$\frac{q+1}{q} \cdot q^{l(r-1)-(l^2 m_1 - R)(m_2 - R)}. \tag{5.1}$$

- Following the first step, an attacker aiming to forge a signature only needs to solve a smaller MQ system, rather than tackling the full signature forgery problem directly. Then for an attacker who wants to forge a fake signature, the remaining is to solving an MQ system of R equations in $ln - (m_2 - R)$ variables. Therefore, the complexity of this step is estimated by

$$\mathsf{MQ}(ln - (m_2 - R), R, q).$$

Complexity. The approximation allows us to obtain a value for the expected MinRank. If the expected number of matrices of a given rank is substantially below one, then it will likely be zero. Therefore, the MinRank of $\mathbf{E_R}$ in the first step is expected to be the smallest R for which $l(r-1) - (l^2m_1 - R)(m_2 - R) \geq 0$, i.e.,

$$R_{\min} = \min\{R : (l^2m_1 - R)(m_2 - R) \leq l(r-1)\}. \tag{5.2}$$

Note that a smaller R then makes the second step of the attack easier. Thus, in terms of $R_{\min}$, we estimate the complexity of this attack by

$$\mathrm{Comp}_{\mathsf{For}} = \mathsf{MQ}(ln - m_2 + R_{\min}, R_{\min}, q) \cdot g(q).$$

gates.

Heuristic approximation of $\mathrm{rank}(\mathbf{E_R})$**.** The estimation of the attack so far assume that the $\mathbf{E_R}$ is behaving as-if random. Here we present experimental evidence that this indeed is the case when $\nu = lr + 2r$. To this end we studied SNOVA instances with slightly reduced parameters, such as $l = 3$ instead of $l = 4$, or $q = 7$ instead of $q = 16$. We present a representative selection of our of results.

Since our proposal is based on the matrix equation formulation (1.3), for an attacker, the $\mathbf{E}_{jk}$ matrices are constructed by the coefficients of ABQ matrices. For $j \in \{1, \ldots, r-1\}$, if we write $\mathbf{R}_j = \sum_{k=0}^{l-1} a_{j,k} \cdot S^k$ then the entries of $\mathbf{E_R}$ are quadratic polynomials of $a_{j,k}$[4] with $j \in \{1, \ldots, r-1\}$ and $k \in \{0, \ldots, l-1\}$. We have created a program that exhausts the distinct values of $a_{j,k}$ to establish the actual rank distribution for a specific SNOVA instance. This software is available at https://github.com/PQCLAB-SNOVA/SNOVA_Analysis.

To present our experiment results, we define the rank drop Δ of a matrix of size $m_2 \times l^2m_1$ with rank R by

$$\Delta = \min(m_2, l^2m_1) - R.$$

There are $q^{l\cdot(r-1)}$ different values of $a_{j,k}$. For the SNOVA parameter set $(24, 5, 16, 4, 4, 5, 80)$ an exhaustive search requires the determination of the rank of 2^{48} matrices of dimension 80×80. We do not have the computational resources to perform an exhaustive search for this parameter set. Thus, as an illustration example, we considered the distribution of Δ for the SNOVA parameter set $(24, 5, 16, 3, 3, 5, 45)$ for a number of seeds, while varying ν. The results of this experiment are presented in Table 13. These results show that $\nu = l \cdot r = 9$ is too low, and that the distribution of Δ appears to have converged when $\nu = 10$ or 11.

Table 13. Distribution of the rank drop Δ for the $l = 3$ reduced parameter set $(24, 5, 16, 3, 3, 5, 45)$. We used 100 different seeds and varied ν. The proposed SNOVA parameter sets use $\nu = r \cdot (l + 2)$, which is 15 for this instance.

Δ	$\nu = 9$	$\nu = 10$	$\nu = 11$	$\nu = 12$	$\nu = 15$
0	1,566,315,264	1,566,323,730	1,566,317,539	1,566,304,515	1,566,302,842
1	111,376,808	111,370,292	111,376,707	111,389,776	111,391,565
2	29,437	27,578	27,354	27,309	27,193
3	86				
4	5				

[4] In the appendix of [17] we have presented an explicit expression for the matrix $\mathbf{E_R} \in \mathbb{F}_q^{m_2 \times l^2 m_1}$ parameterized by the variables $a_{j,k}$.

For a random matrix we can obtain expressions for the probability distribution of the rank drop. Rewrite the equation (5.1) in terms of the rank drop Δ. We have

$$\frac{q+1}{q} \cdot q^{l(r-1)-(l^2m_1-R)(m_2-R)} = \frac{q+1}{q} \cdot q^{l\cdot(r-1)-\Delta\cdot(|m_2-l^2m_1|+\Delta)}.$$

We denote the above number by

$$N_H(\Delta) = \frac{q+1}{q} \cdot q^{l\cdot(r-1)-\Delta\cdot(|m_2-l^2m_1|+\Delta)}. \tag{5.3}$$

Note that, as the sum over all $N_H(\Delta)$ must be $q^{l\cdot(r-1)}$, we also have

$$N_H(\Delta=0) = q^{l\cdot(r-1)} - \sum_{\Delta\geq 1} N_H(\Delta). \tag{5.4}$$

These expressions from random matrices can be compared to experimental result results. As an illustration example, we considered the distribution of Δ for a number of SNOVA parameter sets with $q = 7$. This value of q allows for an exhaustive analysis of the MinRank distribution. In Table 14, we compare the random matrix approximation to the actual experimental data with $\nu = l \cdot r + 2r$. From this, and similar experiments, we conclude that the behavior of $\mathbf{E_R}$ agrees with the heuristic of a random matrix when ν is chosen appropriately. These results provide strong evidence that our choice of $\nu = r\cdot(l+2)$ is sufficient to ensure that the distribution of the rank($\mathbf{E_R}$) matches the random case.

Table 14. Distribution of the rank drop Δ for reduced SNOVA parameter sets compared to the heuristic, equations (5.3) and (5.4). We set $\nu = l \cdot r + 2r$ here.

Δ	$(24,5,7,3,3,5,45)$		$(24,5,7,3,4,6,60)$		$(24,5,7,4,4,5,80)$	
	experiment	$N_H(\Delta)$	experiment	$N_H(\Delta)$	experiment	$N_H(\Delta)$
0	98,439	98,385	40,353,553	40,353,551	11,582,305,886	11,574,896,473
1	19,151	19,208	54	56	2,252,135,235	2,259,801,992
2	59	56	0	1.2×10^{-6}	6,845,654	6,588,344
3	0	3.3×10^{-3}			426	392
4					0	4.8×10^{-4}

Our security estimates from the MinRank attack are based on the expressions for $N_H(\Delta)$ given above. It is worth noting that our complexity estimate in (5.2) is a conservative one, as it is based on the minimal rank $R_{\min}$ for which the attack remains valid.

Forgery Attack with Stable Ideal. In [14], Cabarcas *et al.* proposed a forgery attack against Round 1 SNOVA that combining the block diagonal structure in the MinRank step discovered by Beullens [9] and the technique of the stable ideal. In their forgery attack, the authors construct a system of equations and aim to solve it using their multi-homogeneous XL-like algorithm. However, this multi-homogeneous XL-like algorithm cannot be directly applied to solve the constructed system, since the system itself may not be multi-homogeneous. To overcome this problem, the overall strategy is to first exploit the multi-homogeneous structure to perform the XL and to define a degree D_{md} via a generating function

$$H'(t_1,\ldots,t_l) = \frac{\prod_{i=1}^{l}(1-t_i^2)^{m_1} \prod_{1\leq i<j\leq l}(1-t_it_j)^{2m_1}}{\prod_{i=1}^{l}(1-t_i)^{lm_1-\frac{k}{l}+1}}.$$

The authors define $D_{md} := \sum_{i=1}^{l} d_i$ where $\mathbf{d} = (d_1,\ldots,d_l)$ is the multi-degree such that $H'[\mathbf{t}^{\mathbf{d}}] < 0$. In the most cases of the experiment, the authors observe that the solving degree is less or equal

than D_{md}. Therefore, the authors conclude that the rows and columns of the largest matrix in the Gröbner basis computation are bounded by

$$\binom{l^2o - k + p + D_{md}}{D_{md}} \tag{5.5}$$

where k and p which are the parameters of the algorithm. Based on these experimental results, the author therefore use D_{md} to estimate the complexity and obtain the following complexity estimate

$$\binom{l^2o - k + p + D_{md}}{D_{md}}^{\omega} \tag{5.6}$$

where ω is the linear algebra constant.

The strategy of the attack proposed by Cabarcas *et al.* [14] is based on the diagonal block structure in the MinRank step of Round 1 SNOVA and the observation that the solving degree is less or equal than D_{md}. However, the block diagonal structure no longer exists in the case Round 3 SNOVA. Moreover, the reformulations introduced in Round 3 SNOVA cause the rank found in the MinRank step to be significantly higher than the rank that could previously be obtained. As also noted in their paper, these changes make their attack strategy ineffective.

5.2.3 Collision Attack

To forge a fake signature, an attacker can also try to check M intended signatures $\mathbf{U}_j$ where $j = 1, \ldots, M$, and N hash values $\mathsf{Hash}(\mathbf{pk_d}||\mathbf{m_d}||\mathbf{salt}_k)$ where $k = 1, \ldots, N$, whether there exists a collision $\mathcal{P}^*(\mathbf{U}_j) = \mathsf{Hash}(\mathbf{pk_d}||\mathbf{m_d}||\mathbf{salt}_k)$. And if it does, then the attacker has a valid fake signature. Thus, M signature computations and N hash value computations are involved. Therefore, according to the estimation of [12], the cost of such a collision attack would be

$$M \cdot (l^2 m_1) \cdot \left(2(\log_2 q)^2 + 3 \cdot \log_2 q\right) + N \cdot 2^{17} \tag{5.7}$$

gates in the sense that regarding SNOVA as a UOV scheme over $\mathbb{F}_q$. A collision is to be expected when $MN \approx q^{m_2}$. In a straightforward optimization of equation 5.7, the optimal value occurs when $N \approx q^{m_2/2}$ (note that at optimal values $M > N$). At security level III this optimal value of N requires a storage capacity of the order of 2^{192} bits which is more than the number of atoms in the Earth ($\approx 2^{167}$). Limiting M to the number of atoms in the Earth gives a cost of

$$\frac{q^{m_2}}{2^{167}} \cdot 2^{17}. \tag{5.8}$$

This is an unavoidable lower bound for the complexity of the collision attack when $q^{\frac{m_2}{2}} > 2^{167}$, as is the case for all parameters at security level III and security level V.

Memory-Time tradeoff. In the call for proposals [40], NIST has indicated that a metric to consider is the cost of accessing extremely large amounts of memory. For the collision attack a more basic aspect has to be taken into account: the cost of actually obtaining the amount of storage needed by the attack. For the proposed SNOVA parameters, the lowest value of m_2 is for $(27, 5, 16, 4, 4, 5, 80)$ where $m_2 \log_2(q) = 320$. In this case the optimal value for equation (5.7) is attained when $M \approx 2^{164}$ and $N \approx 2^{156}$. These are not the cost-optimal values however. Suppose that the collision attack runs at 1 GHz and has a runtime of about a year. As there are about $3 \cdot 10^{16}$ cycles in a year, the cost of a $N \approx 2^{156}$ bits of storage has to be comparable to the cost of 2^{109} computational units for the given values of N and M to be cost-optimal. This is not realistic. For our estimate of the collision attack we have assumed a runtime of a year and we have assumed that the cost of a single bit of (persistent) storage is cheaper than a single logic gate by a factor of 10^6. When this cost factor is taken into account, the cost estimate of the collision attack becomes

$$q^{\frac{m_2}{2}} \cdot 2 \left(m_2 \left(2(\log_2 q)^2 + 3 \cdot \log_2 q\right) \cdot 2^{17} \cdot 2^{35}\right)^{\frac{1}{2}}, \tag{5.9}$$

gates. Equation (5.9) can only apply when $q^{\frac{m_2}{2}} < 2^{167}$ as explained above.

Therefore, we estimate the complexity by

$$\text{Comp}_{\mathsf{PlainCol}} = \begin{cases} \dfrac{q^{m_2}}{2^{167}} \cdot 2^{17} & \text{if } q^{\frac{m_2}{2}} > 2^{167}, \\ q^{\frac{m_2}{2}} \cdot 2 \cdot \left(m_2 \cdot (2(\log_2 q)^2 + 3\log_2 q) \cdot 2^{17} \cdot 2^{35}\right)^{\frac{1}{2}} & \text{if } q^{\frac{m_2}{2}} < 2^{167}. \end{cases}$$

Taking into account the cost of memory makes the traditional collision attack less economical than the memoryless collision attack described below. This conclusion is largely independent of the specifics of the assumptions. Changing the runtime to a weekend and changing the relative cost of a storage bit versus one unit of computation to a very unrealistic value of 10^9 does not affect the outcome: The memoryless collision attack is more economical than a traditional collision attack.

For security level I we have presented the numbers taking into account the estimated cost of storage, equation (5.9). At security level III and security level V we have used equation (5.8). Note that all SNOVA schemes satisfy the required complexities even if the cost of obtaining and accessing the memory is ignored as in equation (5.7).

Memoryless Collision Attack. The collision attack comes with a huge memory requirement. Memoryless collision-finding algorithm are considered to be more realistic than a traditional collision attack. As far as we know, for SNOVA the attack of Nikolic and Sasaki [37] is the most efficient memoryless collision attack currently available. This algorithm operates by producing a large number n_v of values v_i by iterative application of a projected public map[5] and doing the same for a large number n_w of iterations on the hash function, $w_i = \mathsf{Hash}(w_{i-1})$. The iterations over v_i and w_i will have a collision with high probability when $n_v n_w > q^{m_2}$. For a complete description, we refer to [37].

Let N_P denote the complexity of calculating the public map, and $N_H = 2^{17.5}$ denote the complexity of the hash ($\mathsf{SHAKE256}$). For an arbitrary value to be evaluated, N_P is slightly more than

$$N_P = m_1 \cdot n^2 \cdot l^3 \cdot r \cdot g(q).$$

The complexity of the algorithm of Nikolic and Sasaki [37] at time-optimal parameters is

$$\text{Comp}_{\mathsf{MLC}} = q^{\frac{m_2}{2}} \cdot 2 \cdot (N_P \cdot N_H)^{\frac{1}{2}}. \tag{5.10}$$

gates. The algorithm of Nikolic and Sasaki [37] has a small overhead that amounts to an additional factor between 1 and 2 in equation (5.10). We have ignored this overhead in our estimates. The number of hashes to be stored is

$$\frac{\max(N_P, N_H)}{\min(N_P, N_H)},$$

which we have also ignored as well as it is small. The Gray-code enumeration optimization [12] underlying the cost estimate (5.9) is very efficient for related inputs. It is not an efficient algorithm to produce the iterated values v_i, a direct evaluation of the public map $\mathcal{P}^*(v_{i-1})$ is faster. Evaluating $\mathcal{P}^*(v_{i-1})$ for arbitrary v_{i-1} has a complexity of N_P. The vinegar variables can be set to some fixed value in the iteration. This reduces the complexity of the evaluation of $\mathcal{P}^*(v_{i-1})$ by a factor n/o.

Therefore, we use the minimal value to estimate the complexity of the collision attack. i,e,

$$\text{Comp}_{\mathsf{Col}} = \min(\text{Comp}_{\mathsf{PlainCol}}, \text{Comp}_{\mathsf{MLC}}) \tag{5.11}$$

gates.

[5] We can use $v_i = f(\mathcal{P}^*(v_{i-1}))$ for the iterated public map where f is some mapping from the codomain of $\mathcal{P}^*$ to its domain.

Table 15. The complexity of the variants of the collision attack.

SL	Name	$\text{Comp}_{\mathsf{PlainCol}}$	$\text{Comp}_{\mathsf{MLC}}$	$\text{Comp}_{\mathsf{Col}}$
I	SNOVA_I_K	234	214	214
	SNOVA_I_B	234	214	214
	SNOVA_I_S	192	182	182
III	SNOVA_III_K	362	279	279
	SNOVA_III_B	330	263	263
	SNOVA_III_S	330	263	263
V	SNOVA_V_K	490	344	344
	SNOVA_V_B	450	323	323
	SNOVA_V_S	450	323	323

5.3 Key Recovery Attacks

SNOVA over ground field $\mathbb{F}_q$. For key recovery attacks against SNOVA, the most important task is to find the oil space $\mathbf{T}^{-1}(\mathcal{O})$. Here, the space $\mathcal{O}$ is defined by

$$\mathcal{O} = \{\mathbf{x} = (x_1, \ldots, x_{ln})^\top \in \mathbb{F}_q^{ln} : x_1 = \cdots = x_{lv} = 0\}.$$

First, we recall the observations discussed by Ikematsu *et al.* [27] and Li *et al.* [33]. Since the components of the secret key $\mathbf{T}$ are in $\mathbb{F}_q[S]$, the matrix $\mathbf{T}$ satisfying the following equation

$$\mathbf{T} \cdot \mathbf{S}^{\otimes n} = \mathbf{S}^{\otimes n} \cdot \mathbf{T}.$$

Then, we have

$$\mathbf{T}^{-1}(\mathcal{O}) = \mathbf{T}^{-1}\mathbf{S}^{\otimes n}(\mathcal{O}) = \mathbf{S}^{\otimes n}\mathbf{T}^{-1}(\mathcal{O}).$$

Therefore, for each oil vector $\mathbf{x} \in \mathbf{T}^{-1}(\mathcal{O})$, we have

$$\mathbf{S}^{\otimes n} \cdot \mathbf{x}, \ldots, (\mathbf{S}^{\otimes n})^{l-1} \cdot \mathbf{x} \in \mathbf{T}^{-1}(\mathcal{O}).$$

In particular, for any $\mathbf{x} \in \mathbf{T}^{-1}(\mathcal{O})$, $a, b \in \{0, \ldots, l-1\}$ and $i \in \{1, \ldots, m_1\}$, we then have

$$\mathbf{x}^\top \cdot \mathbf{P}_{i,a,b} \cdot \mathbf{x} = \mathbf{x}^\top \cdot (\mathbf{S}^{\otimes n})^a \cdot \mathbf{P}_i \cdot (\mathbf{S}^{\otimes n})^b \cdot \mathbf{x} = 0.$$

Therefore, the public key matrices $\mathbf{P}_1, \ldots, \mathbf{P}_{m_1} \in \mathbb{F}_q^{ln \times ln}$ of a $(v, o, q, l, r, m_1, m_2)$ SNOVA scheme can be regarded as the public key matrices of a $(lv, lo, q, l^2 m_1)$ UOV scheme.

SNOVA over the extension field $\mathbb{F}_{q^l}$. We first review the technique of lifting the SNOVA scheme to the extension field $\mathbb{F}_{q^l}$ introduced in [14, 24].

Let $\tau : \mathbb{F}_{q^l} \to \mathbb{F}_{q^l}$ denote the Frobenius map defined by $z \mapsto z^q$. By abuse of notation, we also use τ to represent its entry-wise extension to vectors and matrices. Let $\lambda \in \mathbb{F}_{q^l}$ be an eigenvalue of S and $\xi \in \mathbb{F}_{q^l}^l$ be the eigenvector of S corresponding to eigenvalue λ. Let $L = [\xi \ \tau(\xi) \ \cdots \ \tau^{l-1}(\xi)] \in \mathbb{F}_{q^l}^{l \times l}$. It can be checked that $\tau^i(\xi)$ is the eigenvector corresponding to $\tau^i(\lambda)$ and L is invertible due to the fact that the characteristic polynomial of S is irreducible. Then, we have

$$L^{-1} \cdot S \cdot L = \begin{bmatrix} \lambda & & & \\ & \lambda^q & & \\ & & \ddots & \\ & & & \lambda^{q^{l-1}} \end{bmatrix} = \begin{bmatrix} \lambda & & & \\ & \tau(\lambda) & & \\ & & \ddots & \\ & & & \tau^{l-1}(\lambda) \end{bmatrix}. \tag{5.12}$$

Let $\mathbf{\Pi}$ be the commutation matrix corresponding to index (n, l), i.e.,

$$\mathbf{\Pi} = \begin{bmatrix} E_{11} & E_{21} & \cdots & E_{l1} \\ E_{12} & E_{22} & \cdots & E_{l2} \\ \vdots & \vdots & \ddots & \vdots \\ E_{1l} & E_{2l} & \cdots & E_{nl} \end{bmatrix} \tag{5.13}$$

where E_{ij} is the matrix of size $l \times n$ whose (i, j)-entry is one and other entries are zero. Given the public key $\mathbf{P}_1, \ldots, \mathbf{P}_{m_1} \in \mathbb{F}_q^{ln \times ln}$, the Theorem 5.1 below demonstrates how to lift the SNOVA scheme to the extension field $\mathbb{F}_q^l$.

Theorem 5.1 ([24, Theorem 1]). *Let $\bar{\mathbf{T}} = \mathbf{\Pi}^\top \cdot (\mathbf{L}^{\otimes n})^{-1} \cdot \mathbf{T} \cdot \mathbf{L}^{\otimes n} \cdot \mathbf{\Pi}$. For $k \in \{1, \ldots, m_1\}$, let $\bar{\mathbf{F}}_k = \mathbf{\Pi}^\top \cdot (\mathbf{L}^{\otimes n})^\top \cdot \mathbf{F}_k \cdot \mathbf{L}^{\otimes n} \cdot \mathbf{\Pi}$ and $\bar{\mathbf{P}}_k = \mathbf{\Pi}^\top \cdot (\mathbf{L}^{\otimes n})^\top \cdot \mathbf{P}_k \cdot \mathbf{L}^{\otimes n} \cdot \mathbf{\Pi}$. For each $i, j \in \{1, \ldots, l\}$, we define $\bar{F}_{k,(i,j)} \in \mathbb{F}_{q^l}^{n \times n}$ and $\bar{P}_{k,(i,j)} \in \mathbb{F}_{q^l}^{n \times n}$ by the (i, j)-th submatrices of $\bar{\mathbf{F}}_k$ and $\bar{\mathbf{P}}_k$, respectively. Then, we have*

(i) For all k, i, j, the lower right o-by-o submatrix of $\bar{F}_{k,(i,j)}$ is the zero matrix, i.e., $\bar{F}_{k,(i,j)}$ is a UOV-type matrix, and we have

$$\tau(\bar{F}_{k,(i,j)}) = \bar{F}_{k,((i+1) \bmod l,\, (j+1) \bmod l)}. \tag{5.14}$$

More precisely, we have

$$\bar{\mathbf{F}}_k = \begin{bmatrix} \bar{F}_{k,(1,1)} & \bar{F}_{k,(1,2)} & \cdots & \bar{F}_{k,(1,l)} \\ \tau(\bar{F}_{k,(1,l)}) & \tau(\bar{F}_{k,(1,1)}) & \cdots & \tau(\bar{F}_{k,(1,l-1)}) \\ \vdots & \vdots & \ddots & \vdots \\ \tau^{l-1}(\bar{F}_{k,(1,2)}) & \tau^{l-1}(\bar{F}_{k,(1,3)}) & \cdots & \tau^{l-1}(\bar{F}_{k,(1,1)}) \end{bmatrix}, \tag{5.15}$$

(ii) For the matrix $\bar{\mathbf{T}} = \mathbf{\Pi}^\top \cdot (\mathbf{L}^{\otimes n})^{-1} \cdot \mathbf{T} \cdot \mathbf{L}^{\otimes n} \cdot \mathbf{\Pi}$, we have

$$\bar{\mathbf{T}} = \begin{bmatrix} \bar{T}_1 & & & \\ & \tau(\bar{T}_1) & & \\ & & \ddots & \\ & & & \tau^{l-1}(\bar{T}_1) \end{bmatrix}, \tag{5.16}$$

where $\bar{T}_1 \in \mathbb{F}_{q^l}^{n \times n}$ is of the form

$$\begin{bmatrix} I_{v \times v} & *_{v \times o} \\ 0 & I_{o \times o} \end{bmatrix}. \tag{5.17}$$

(iii) For all k, i, j, we have

$$\tau(\bar{P}_{k,(i,j)}) = \bar{P}_{k,((i+1) \bmod l,\, (j+1) \bmod l)}, \tag{5.18}$$

$$\bar{P}_{k,(i,j)} = (\tau^{i-1}(\bar{T}_1))^\top \cdot \bar{F}_{k,(i,j)} \cdot \tau^{j-1}(\bar{T}_1). \tag{5.19}$$

The oil space over the extension field $\mathbb{F}_{q^l}$ is the subspace $\bar{T}_1^{-1}(\bar{\mathcal{O}})$ of $\mathbb{F}_{q^l}^n$ where $\bar{\mathcal{O}} \subset \mathbb{F}_{q^l}^n$ is defined by

$$\bar{\mathcal{O}} = \left\{ \mathbf{x} = (x_1, \ldots, x_n)^\top \in \mathbb{F}_{q^l}^n : x_1 = \ldots = x_v = 0 \right\}.$$

Note that the equation

$$\bar{P}_{k,(i,j)} = (\tau^{i-1}(\bar{T}_1))^\top \cdot \bar{F}_{k,(i,j)} \cdot \tau^{j-1}(\bar{T}_1) \tag{5.20}$$

implies that

$$(\tau^{i-1}\left((\bar{T}_1)\right)^\top)^{-1} \cdot \bar{P}_{k,(i,j)} \cdot \left(\tau^{j-1}(\bar{T}_1)\right)^{-1} = \bar{F}_{k,(i,j)}. \tag{5.21}$$

Since the lower right $o \times o$ submatrix of $\bar{F}_{k,(i,j)}$ is the zero matrix, we have

$$\mathbf{x}_i{}^\top \cdot \bar{P}_{k,(i,j)} \cdot \mathbf{x}_j = 0, \quad k \in [m_1],\, i, j \in [l] \tag{5.22}$$

for any $\mathbf{x}_i \in \tau^{i-1}(\bar{T}_1{}^{-1}\bar{\mathcal{O}})$ and $\mathbf{x}_j \in \tau^{j-1}(\bar{T}_1{}^{-1}\bar{\mathcal{O}})$. It is worth noting that an attacker can recover the original oil space $\mathbf{T}^{-1}(\mathcal{O})$ from the above space $\bar{T}_1^{-1}(\bar{\mathcal{O}})$.

Remark 5.2. A second lifting approach is discussed in [33, 36], where SNOVA is simply regarded as a (v, o, q^l, m_1)-UOV over the extension field $\mathbb{F}_{q^l}$. The corresponding key-recovery attacks are less effective than those based on the lifting approach of [14, 24], as they do not exploit the underlying multi-homogeneous structure of the system.

5.3.1 KS Attack

Plain KS attack. The oil space corresponding to the public key matrices $\mathbf{P}_1, \ldots, \mathbf{P}_{m_1} \in \mathbb{F}_q^{ln \times ln}$ is of dimension lo. The KS attack first seeks two invertible linear combinations of $\mathbf{P}_1, \ldots, \mathbf{P}_{m_1}$, say $\mathbf{L}_i$ and $\mathbf{L}_j$. Then, the oil space $\mathbf{T}^{-1}(\mathcal{O})$ can be recovered by computing the invariant subspace of $\mathbf{L}_j^{-1}\mathbf{L}_i$. Therefore, a lower bound to the complexity can be estimated as

$$\mathrm{Comp}_{\mathsf{PlainKS}} = q^{lv-lo} \cdot g(q) \tag{5.23}$$

gates.

Lifting KS attack. Since the oil space $\bar{T}_1^{-1}(\bar{\mathcal{O}}) \subset \mathbb{F}_{q^l}^n$ over the extension field $\mathbb{F}_{q^l}$ has dimension o, a lower bound to the complexity can be estimated as

$$\mathrm{Comp}_{\mathsf{LiftingKS}} = (q^l)^{v-o} \cdot g(q^l) \tag{5.24}$$

gates.

By comparing the equation (5.23) and the equation (5.24), we use the minimal value to estimate the complexity of KS attack. i,e,

$$\mathrm{Comp}_{\mathsf{KS}} = \min(\mathrm{Comp}_{\mathsf{PlainKS}}, \mathrm{Comp}_{\mathsf{LiftingKS}}) = \mathrm{Comp}_{\mathsf{PlainKS}} \tag{5.25}$$

gates.

Table 16. The complexity of the variants of the KS attack.

SL	Name	$\mathrm{Comp}_{\mathsf{PlainKS}}$	$\mathrm{Comp}_{\mathsf{LiftingKS}}$	$\mathrm{Comp}_{\mathsf{KS}}$
I	SNOVA_I_K	421	425	421
	SNOVA_I_B	373	377	373
	SNOVA_I_S	357	361	357
III	SNOVA_III_K	549	553	549
	SNOVA_III_B	533	537	533
	SNOVA_III_S	517	521	517
V	SNOVA_V_K	725	729	725
	SNOVA_V_B	705	709	705
	SNOVA_V_S	685	689	685

5.3.2 Reconciliation Attack

Plain reconciliation attack. In [27] and [33], Ikematsu *et al.* and Li *et al.* respectively propose variants of the reconciliation attack. For $i \in \{1, \ldots, m_1\}$ and $a, b \in \{0, \ldots, l-1\}$, we define $\mathbf{P}_{i,a,b} := (\mathbf{S}^a)^{\otimes n}\mathbf{P}_i(\mathbf{S}^b)^{\otimes n}$. The reconciliation attack on SNOVA over the ground field $\mathbb{F}_q$ is carried out by finding a vector $\mathbf{x} \in \mathbf{T}^{-1}(\mathcal{O})$ via the system

$$\mathbf{x}^\top \cdot \mathbf{P}_{i,a,b} \cdot \mathbf{x} = \mathbf{x}^\top \cdot (\mathbf{S}^{\otimes n})^a \cdot \mathbf{P}_i \cdot (\mathbf{S}^{\otimes n})^b \cdot \mathbf{x} = 0. \tag{5.26}$$

Since $\mathbf{P}_1, \ldots, \mathbf{P}_{m_1}$ are not symmetric, the system (5.26) yields a MQ system consisting of $l^2 m_1$ equations in $ln - (lo - 1) = lv + 1$ variables. Hence the complexity of reconciliation attack is

estimated by

$$\mathrm{Comp}_{\mathsf{PlainRec}} = \min_{k} q^k \cdot \mathsf{MQ}(lv+1-k, l^2m_1, q) \cdot g(q) \tag{5.27}$$

gates where $\max\{0, lv+1-l^2 \cdot m_1\} \leq k \leq lv$.

Lifting reconciliation attack. In [14], the Cabarcas *et al.* proposed a variant of lifting reconciliation attack against SNOVA. The attack presented in this subsection is a reformulation of the reconciliation attack in [14]. The lifting reconciliation attack is carried out by solving the following system

$$\mathbf{x}_i^{\top} \cdot \bar{P}_{k,(i,j)} \cdot \mathbf{x}_j = 0, \quad k \in [m_1],\ i,j \in [l] \tag{5.28}$$

where $\mathbf{x}_i \in \tau^{i-1}(\bar{T_1}^{-1}\bar{\mathcal{O}})$ and $\mathbf{x}_j \in \tau^{j-1}(\bar{T_1}^{-1}\bar{\mathcal{O}})$. The system (5.28) has $N = lv$ variables in total. For each $i = j \in [l]$, there are m_1 independent homogeneous quadratic equations. For each $1 \leq i < j \leq l$, there are $2m_1$ homogeneous bilinear equations. Hence, we have $M = l(m_1) + \binom{l}{2}2m_1 = l^2m_1$ equations in total.

When $lv \geq l^2m_1$, one can first assign $N - M = lv - l^2m_1$ random values in the base field $\mathbb{F}_q$ (as mentioned in [14]) and then try to solve the system over the extension field $\mathbb{F}_{q^l}$ and seek for a solution that corresponds to a solution in the base field $\mathbb{F}_q$. After randomly assigning variables, the resulting system has l^2m_1 variables and l^2m_1 equations in total. Consequently, the attack can be summarized in the following steps:

- First, consider the system (5.28). Randomly assign $N - M = lv - l^2m_1$ values in the base field $\mathbb{F}_q$ to the system (5.28). Then, this results in a system of l^2m_1 variables and equations in total. The resulting system has l sets of variables and each set has lm_1 variables.
- Randomly guess $0 \leq k \leq lm_1$ entries in the extension field $\mathbb{F}_{q^l}$. This will contribute a factor $(q^l)^k$ in the complexity estimation.
- Solve the system over extension field and try to find a desired solution. If this succeeds, the procedure terminates. Otherwise, the attacker returns to the first step. This will contribute a factor $q^{N-M} = q^{lv-l^2m_1}$ in the complexity estimation.

Therefore, the complexity is given as

$$\mathrm{Comp}_{\mathsf{LiftingRec}} = \min_{k \in [lm_1]} q^{lv-l^2m_1} \cdot (q^l)^k \cdot 3 \cdot (lm_1-k)^2 \cdot \prod_{i=1}^{l} \binom{lm_1-k+d_i}{d_i}^2 \cdot g(q^l) \tag{5.29}$$

gates. The solving degree $\mathbf{d} = (d_1, \ldots, d_l)$ is the multi-degree so that the coefficient of $\prod_{i=1}^{l} t_i^{d_i}$ in the following series

$$\frac{\prod_{1 \leq i < j \leq l} (1-t_it_j)^{2m_1} \cdot \prod_{i \in [l]} (1-t_i^2)^{m_1}}{\prod_{i \in [l]} (1-t_i)^{lm_1-k+1}} \tag{5.30}$$

is non-positive.

We can only consider multi-degree $\mathbf{d}$ with $d_1 \geq d_2 \geq \cdots \geq d_l$ due to the symmetry in the generating function. Thus, we use the minimal value to estimate the complexity of reconciliation attack. i,e,

$$\mathrm{Comp}_{\mathsf{Rec}} = \min(\mathrm{Comp}_{\mathsf{PlainRec}}, \mathrm{Comp}_{\mathsf{LiftingRec}}) \tag{5.31}$$

gates.

Remark 5.3. Note that the reconciliation attack proposed by Nakamura *et al.* [36] is corresponding to the case of $l = 1$. In general, the lifting attacks proposed by Nakamura *et al.* [36] is not expected to be more effective than the lifting attacks proposed by Furue *et al.* [24], since it does not utilize the multi-homogeneous structure.

Table 17. The complexity of the variants of the reconciliation attack.

SL	Name	$\mathrm{Comp}_{\mathsf{PlainRec}}$	$\mathrm{Comp}_{\mathsf{LiftingRec}}$	$\mathrm{Comp}_{\mathsf{Rec}}$
I	SNOVA_I_K	347	332	332
	SNOVA_I_B	315	300	300
	SNOVA_I_S	315	300	300
III	SNOVA_III_K	432	414	414
	SNOVA_III_B	432	414	414
	SNOVA_III_S	432	414	414
V	SNOVA_V_K	552	528	528
	SNOVA_V_B	552	528	528
	SNOVA_V_S	552	528	528

5.3.3 Intersection Attack

Plain intersection attack. The intersection attack aims to find a vector

$$\mathbf{y} \in \mathbf{L}_1\big(\mathbf{T}^{-1}\mathcal{O}\big) \cap \mathbf{L}_2\big(\mathbf{T}^{-1}\mathcal{O}\big),$$

where $\mathbf{L}_1, \mathbf{L}_2 \in \mathbb{F}_q^{ln\times ln}$ are two invertible linear combinations of the matrices $\mathbf{P}_1, \ldots, \mathbf{P}_{m_1}$. On the other hand, besides treating $\mathbf{L}_1$ and $\mathbf{L}_2$ as linear combinations of the matrices $\mathbf{P}_i$, we can further regard them as linear combinations of the matrices $\mathbf{P}_{i,a,b} = (\mathbf{S}^{\otimes n})^a \cdot \mathbf{P}_i \cdot (\mathbf{S}^{\otimes n})^b$. This affects the system of equations relevant to the intersection attack and results in a different number of equations. In our security analysis of the intersection attack, we take both cases into account.

The attack is carried by considering the following system

$$\begin{cases} \big(\mathbf{L}_1^{-1}\mathbf{y}\big)^\top \cdot (\mathbf{S}^{\otimes n})^a \cdot \mathbf{P}_i \cdot (\mathbf{S}^{\otimes n})^b \cdot \big(\mathbf{L}_1^{-1}\mathbf{y}\big) = 0 \\ \big(\mathbf{L}_1^{-1}\mathbf{y}\big)^\top \cdot (\mathbf{S}^{\otimes n})^a \cdot \mathbf{P}_i \cdot (\mathbf{S}^{\otimes n})^b \cdot \big(\mathbf{L}_2^{-1}\mathbf{y}\big) = 0 \\ \big(\mathbf{L}_2^{-1}\mathbf{y}\big)^\top \cdot (\mathbf{S}^{\otimes n})^a \cdot \mathbf{P}_i \cdot (\mathbf{S}^{\otimes n})^b \cdot \big(\mathbf{L}_1^{-1}\mathbf{y}\big) = 0 \\ \big(\mathbf{L}_2^{-1}\mathbf{y}\big)^\top \cdot (\mathbf{S}^{\otimes n})^a \cdot \mathbf{P}_i \cdot (\mathbf{S}^{\otimes n})^b \cdot \big(\mathbf{L}_2^{-1}\mathbf{y}\big) = 0 \end{cases} \tag{5.32}$$

If $\mathbf{L}_1, \mathbf{L}_2$ are the linear combination of $\mathbf{P}_i$, we can write

$$\mathbf{L}_1 = \sum_{i=1}^{m_1} \alpha_i \cdot \mathbf{P}_i$$
$$\mathbf{L}_2 = \sum_{i=1}^{m_1} \beta_i \cdot \mathbf{P}_i$$

Then, we have

$$\begin{aligned} &\sum_{i=1}^{m_1} \beta_i \cdot \big(\mathbf{L}_1^{-1}\mathbf{y}\big)^\top \cdot (\mathbf{S}^{\otimes n})^a \cdot \mathbf{P}_i \cdot \big(\mathbf{L}_2^{-1}\mathbf{y}\big) \\ =& \big(\mathbf{L}_1^{-1}\mathbf{y}\big)^\top \cdot (\mathbf{S}^{\otimes n})^a \cdot \mathbf{L}_2 \cdot \big(\mathbf{L}_2^{-1}\mathbf{y}\big) \\ =& \big(\mathbf{L}_1^{-1}\mathbf{y}\big)^\top \cdot (\mathbf{S}^{\otimes n})^a \cdot \mathbf{L}_1 \cdot \big(\mathbf{L}_1^{-1}\mathbf{y}\big) \\ =& \sum_{i=1}^{m_1} \alpha_i \cdot \big(\mathbf{L}_1^{-1}\mathbf{y}\big)^\top \cdot (\mathbf{S}^{\otimes n})^a \cdot \mathbf{P}_i \cdot \big(\mathbf{L}_1^{-1}\mathbf{y}\big) \end{aligned}$$

and

$$\begin{aligned}
&\sum_{i=1}^{m_1} \alpha_i \cdot \left(\mathbf{L}_2^{-1}\mathbf{y}\right)^\top \cdot (\mathbf{S}^{\otimes n})^a \cdot \mathbf{P}_i \cdot \left(\mathbf{L}_1^{-1}\mathbf{y}\right) \\
=&\left(\mathbf{L}_2^{-1}\mathbf{y}\right)^\top \cdot (\mathbf{S}^{\otimes n})^a \cdot \mathbf{L}_1 \cdot \left(\mathbf{L}_1^{-1}\mathbf{y}\right) \\
=&\left(\mathbf{L}_2^{-1}\mathbf{y}\right)^\top \cdot (\mathbf{S}^{\otimes n})^a \cdot \mathbf{L}_2 \cdot \left(\mathbf{L}_2^{-1}\mathbf{y}\right) \\
=&\sum_{i=1}^{m_1} \beta_i \cdot \left(\mathbf{L}_2^{-1}\mathbf{y}\right)^\top \cdot (\mathbf{S}^{\otimes n})^a \cdot \mathbf{P}_i \cdot \left(\mathbf{L}_2^{-1}\mathbf{y}\right).
\end{aligned}$$

Hence there are also $2l$ redundant equations in the system (5.32).

If $\mathbf{L}_1, \mathbf{L}_2$ are the linear combination of $(\mathbf{S}^{\otimes n})^a \cdot \mathbf{P}_i \cdot (\mathbf{S}^{\otimes n})^b$, then we write

$$\begin{aligned}
\mathbf{L}_1 &= \sum_{i,a,b} \alpha_i^{ab} \cdot (\mathbf{S}^{\otimes n})^a \cdot \mathbf{P}_i \cdot (\mathbf{S}^{\otimes n})^b \\
\mathbf{L}_2 &= \sum_{i,a,b} \beta_i^{ab} \cdot (\mathbf{S}^{\otimes n})^a \cdot \mathbf{P}_i \cdot (\mathbf{S}^{\otimes n})^b.
\end{aligned}$$

Then, we have

$$\begin{aligned}
&\sum_{i,a,b} \beta_i^{ab} \cdot \left(\mathbf{L}_1^{-1}\mathbf{y}\right)^\top \cdot (\mathbf{S}^{\otimes n})^a \cdot \mathbf{P}_i \cdot (\mathbf{S}^{\otimes n})^b \cdot \left(\mathbf{L}_2^{-1}\mathbf{y}\right) \\
=&\left(\mathbf{L}_1^{-1}\mathbf{y}\right)^\top \cdot \mathbf{L}_2 \cdot \left(\mathbf{L}_2^{-1}\mathbf{y}\right) \\
=&\left(\mathbf{L}_1^{-1}\mathbf{y}\right)^\top \cdot \mathbf{L}_1 \cdot \left(\mathbf{L}_1^{-1}\mathbf{y}\right) \\
=&\sum_{i,a,b} \alpha_i^{ab} \cdot \left(\mathbf{L}_1^{-1}\mathbf{y}\right)^\top \cdot (\mathbf{S}^{\otimes n})^a \cdot \mathbf{P}_i \cdot (\mathbf{S}^{\otimes n})^b \cdot \left(\mathbf{L}_1^{-1}\mathbf{y}\right)
\end{aligned}$$

and

$$\begin{aligned}
&\sum_{i,a,b} \alpha_i^{ab} \cdot \left(\mathbf{L}_2^{-1}\mathbf{y}\right)^\top \cdot (\mathbf{S}^{\otimes n})^a \cdot \mathbf{P}_i \cdot (\mathbf{S}^{\otimes n})^b \cdot \left(\mathbf{L}_1^{-1}\mathbf{y}\right) \\
=&\left(\mathbf{L}_2^{-1}\mathbf{y}\right)^\top \cdot \mathbf{L}_1 \cdot \left(\mathbf{L}_1^{-1}\mathbf{y}\right) \\
=&\left(\mathbf{L}_2^{-1}\mathbf{y}\right)^\top \cdot \mathbf{L}_2 \cdot \left(\mathbf{L}_2^{-1}\mathbf{y}\right) \\
=&\sum_{i,a,b} \beta_i^{ab} \cdot \left(\mathbf{L}_2^{-1}\mathbf{y}\right)^\top \cdot (\mathbf{S}^{\otimes n})^a \cdot \mathbf{P}_i \cdot (\mathbf{S}^{\otimes n})^b \cdot \left(\mathbf{L}_2^{-1}\mathbf{y}\right)
\end{aligned}$$

Hence there are only 2 redundant equations in the system (5.32). It's same for the linear combinations of $(\mathbf{S}^{\otimes n})^a \mathbf{P}_i (\mathbf{S}^{\otimes n})^b + (\mathbf{S}^{\otimes n})^b \mathbf{P}_i^\top (\mathbf{S}^{\otimes n})^a$.

In concoulsion, if $\mathbf{L}_1$ and $\mathbf{L}_2$ are taken as linear combinations of the matrices $\mathbf{P}_i$, then the system of equations (5.32) contains $2l$ redundant equations. Consequently, this results in a system with $4l^2m_1 - 2l$ equations in ln variables. On the other hand, if $\mathbf{L}_1$ and $\mathbf{L}_2$ are considered as linear combinations of the matrices $\mathbf{P}_{i,a,b}$, then the system (5.32) contains only 2 redundant equations. In our complexity estimation, we consider both scenarios. Note that throughout our analysis we always assume $v \geq 2o$. Let

$$\begin{aligned}
\mathrm{Comp}_{\mathsf{PlainInta}} &= \min_{k_1} q^{lv-2lo+1} \cdot q^{k_1} \cdot \mathsf{MQ}(ln - k_1 + 1, 4l^2m_1 - 2l, q) \cdot g(q) \\
\mathrm{Comp}_{\mathsf{PlainIntb}} &= \min_{k_2} q^{lv-2lo+1} \cdot q^{k_2} \cdot \mathsf{MQ}(ln - k_2 + 1, 4l^2m_1 - 2, q) \cdot g(q)
\end{aligned}$$

where

$$\begin{aligned}
\max\{0, ln - (4l^2m_1 - 2l)\} \leq k_1 \leq ln - 1 \\
\max\{0, ln - (4l^2m_1 - 2)\} \leq k_2 \leq ln - 1.
\end{aligned}$$

Then, we estimate the complexity of intersection attack as

$$\mathrm{Comp}_{\mathsf{PlainInt}} = \min(\mathrm{Comp}_{\mathsf{PlainInta}}, \mathrm{Comp}_{\mathsf{PlainIntb}})$$

gates.

Lifting intersection attack. In [24], Furue *et al.* proposed a lifting intersection attack against SNOVA. Let W_1 and W_2 be two invertible linear combinations of $\bar{P}_{1,(1,1)}, \dots \bar{P}_{m_1,(1,1)}$. The lifting intersection attack is carried out by solving the following system

$$\begin{cases} \left((\tau^{i-1}W_1)^{-1} \cdot \mathbf{x}_i\right)^\top \cdot \bar{P}_{k,(i,j)} \cdot \left((\tau^{j-1}W_1)^{-1} \cdot \mathbf{x}_j\right) = 0, \\ \left((\tau^{i-1}W_1)^{-1} \cdot \mathbf{x}_i\right)^\top \cdot \bar{P}_{k,(i,j)} \cdot \left((\tau^{j-1}W_2)^{-1} \cdot \mathbf{x}_j\right) = 0, \\ \left((\tau^{i-1}W_2)^{-1} \cdot \mathbf{x}_i\right)^\top \cdot \bar{P}_{k,(i,j)} \cdot \left((\tau^{j-1}W_1)^{-1} \cdot \mathbf{x}_j\right) = 0, \\ \left((\tau^{i-1}W_2)^{-1} \cdot \mathbf{x}_i\right)^\top \cdot \bar{P}_{k,(i,j)} \cdot \left((\tau^{j-1}W_2)^{-1} \cdot \mathbf{x}_j\right) = 0, \end{cases} \tag{5.33}$$

where $k \in [m_1]$, $1 \le l' \le l$ and $i, j \in [l']$.

For each $i = j \in [l']$, there are $4m_1 - 2$ independent homogeneous quadratic equations. For each $1 \le i < j \le l'$, there are $4 \cdot 2m_1 = 8m_1$ homogeneous bilinear equations. Hence, $l'(4m_1 - 2) + \binom{l'}{2} 8m_1 = 4m_1 l'^2 - 2l'$ equations in total. The complexity is given as

$$\mathrm{Comp}_{\mathsf{LiftingInt}} = \min_{1 \le l' \le l} (q^l)^{v-2o+1} \cdot 3 \cdot \binom{n+1}{2} \cdot \prod_{i=1}^{l'} \binom{n+d_i-1}{d_i}^2 \cdot g(q^l) \tag{5.34}$$

where $d_1, \dots, d_{l'}$ denote the solving degree for the l' sets of variables $\mathbf{x}_1, \dots, \mathbf{x}_{l'}$. The solving degree is chosen such that the coefficient of $\prod_{i=1}^{l'} t_i^{d_i}$ in the following series

$$\frac{\prod_{1 \le i < j \le l'} (1 - t_i t_j)^{8m_1} \cdot \prod_{i \in [l']} (1 - t_i^2)^{4m_1 - 2}}{\prod_{i \in [l']} (1 - t_i)^n}. \tag{5.35}$$

We can only consider multi-degree $\mathbf{d} = (d_1, \dots, d_{l'})$ with $d_1 \ge d_2 \ge \cdots \ge d_{l'}$ due to the symmetry in the generating function and we only consider l' so that $nl' \le 4m_1 l'^2 - 2l'$.

Thus, we use the minimal value to estimate the complexity of intersection attack. i.e.,

$$\mathrm{Comp}_{\mathsf{Int}} = \min(\mathrm{Comp}_{\mathsf{PlainInt}}, \mathrm{Comp}_{\mathsf{LiftingInt}}) \tag{5.36}$$

gates.

Table 18. The complexity of the variants of the intersection attack.

SL	Name	$\mathrm{Comp}_{\mathsf{PlainInt}}$	$\mathrm{Comp}_{\mathsf{LiftingInt}}$	$\mathrm{Comp}_{\mathsf{Int}}$
I	SNOVA_I_K	510	507	507
	SNOVA_I_B	445	436	436
	SNOVA_I_S	414	411	411
III	SNOVA_III_K	647	643	643
	SNOVA_III_B	623	615	615
	SNOVA_III_S	600	589	589
V	SNOVA_V_K	842	836	836
	SNOVA_V_B	811	802	802
	SNOVA_V_S	779	769	769

Remark 5.4. Note that the intersection attack proposed by Nakamura *et al.* [36] is corresponding to the case of $l' = 1$. In general, the lifting attacks proposed by Nakamura *et al.* [36] is not expected to be more effective than the lifting attacks proposed by Furue *et al.* [24], since it does not utilize the multi-homogeneous structure.

5.3.4 Rectangular MinRank Attack

In [24], Furue *et al.* proposed a rectangular MinRank attack over extension field $\mathbb{F}_{q^l}$ against SNOVA. If there exists an $l' \in [l]$ such that $v < 2m_1 l'$, then an attacker can perform the rectangular MinRank attack on SNOVA over the extension field by solving the following systems for variables $\mathbf{x}_1, \ldots, \mathbf{x}_{l'} \in \mathbb{F}_{q^\ell}^n$:

$$\begin{cases} \operatorname{rank}\left(L_{l'}(\mathbf{x}_1, \ldots, \mathbf{x}_{l'})\right) \leq v \\ \mathbf{x}_i^\top \cdot \bar{P}_{k,(i,j)} \cdot \mathbf{x}_j = 0 \end{cases}$$

where $k \in [m_1]$, $i, j \in [l']$ and $L_{l'}(\mathbf{x}_1, \ldots, \mathbf{x}_{l'}) \in \mathbb{F}_{q^l}^{n \times 2l'm_1}$ is a matrix constructed from the matrices $\bar{P}_{k,(i,j)}$ via matrix deformation technique, for more details, we refer to [24]. The complexity of the attack is given by

$$\mathrm{Comp}_{\mathsf{RecMinRank}} = \min_{\substack{1 \leq l' \leq l \\ v < n' \leq n}} 3 \cdot (v+1)^2 \cdot \binom{n'}{v}^2 \cdot \prod_{i=1}^{l'} \binom{v+d_i}{d_i}^2 \cdot g(q^l)$$

gates where $\mathbf{d} = (d_1, \ldots, d_{l'})$ denote the solving degree for the l' set of variables $\mathbf{x}_1, \ldots, \mathbf{x}_{l'}$ and $v < n' \leq n$. The multi-degree $\mathbf{d} = (d_1, \ldots, d_{l'})$ is chosen by the condition that the coefficient of $\prod_{i=1}^{l'} t_i^{d_i}$ in $G(\mathbf{t})$ is non-positive. Where,

$$G(\mathbf{t}) = G'(\mathbf{t}) \cdot \prod_{1 \leq i < j \leq l'} (1 - t_i t_j)^{2m_1} \cdot \prod_{i \in [l']} (1 - t_i^2)^{m_1}$$

and the each coefficient of $\prod_{i=1}^{l'} t_i^{d_i}$ in $G'(t)$ is given as

$$\binom{n'}{v} \cdot \prod_{i \in [l']} \binom{v+d_i}{d_i}$$

$$- \sum_{\mathbf{b} \in A_{\mathbf{d}}} \left((-1)^{|\mathbf{b}|+1} \cdot \prod_{i \in [l']} \binom{2m_1 + b_i - 1}{b_i} \cdot \binom{n'}{v + |\mathbf{b}|} \cdot \prod_{i \in [l']} \binom{v + d_i - b_i}{d_i - b_i} \right),$$

and the set $A_{\mathbf{d}}$ is defined by

$$A_{\mathbf{d}} = \left\{ \mathbf{b} = (b_1, \ldots, b_{l'}) \in \mathbb{Z}_{\geq 0}^{l'} \;\middle|\; 0 \leq b_i \leq d_i \ (i \in [l']),\ |\mathbf{b}| \geq 1 \right\},$$

under the condition of $0 \leq |\mathbf{d}| \leq v + 1$.

Note that, in [24], the authors state that the complexity estimated by the above can be regarded as the lower bound for the concrete complexity. Therefore, from the scheme designing perspective, the complexity estimations in this subsection are conservative, the actual complexity of the attack is expected to be higher than the estimates.

Remark 5.5. The complexity of the rectangular MinRank attack proposed by Suzuki *et al.* in [51] is expected to be higher than the complexity in this subsection since the attack itself does not utilize the multi-homogeneous structure.

5.3.5 Wedge Attack

The wedge attack proposed by Lars Ran [45] offered the cryptographic community a new perspective on the security of the UOV family, inspiring researchers to revisit the UOV variants and leading to additional analysis on SNOVA [52, 13].

Even characteristic. The wedge map of SNOVA is defined as

$$\mathcal{M} : \bigwedge^{lv} \mathbb{F}_q^{ln} \to \left(\bigwedge^{lv+2} \mathbb{F}_q^{ln} \right)^{l^2 m_1}, \quad \omega \mapsto \left(\omega \wedge \widehat{Q}_i^{(a,b)} \right)_{\substack{i \in \{1, \ldots, m_1\} \\ a,b \in \{0, \ldots, l-1\}}}.$$

Here, the $\widehat{Q}_i^{(a,b)} \in \bigwedge^2 \mathbb{F}_q^{ln}$ is the exterior 2-form corresponding to the bilinear form

$$Q_i^{(a,b)}(\mathbf{x},\mathbf{y}) \ = \mathbf{x}^\top \cdot \left(\mathbf{P}_{i,a,b} + (\mathbf{P}_{i,a,b})^\top\right) \cdot \mathbf{y}$$

where $i \in \{1,\ldots,m_1\}$ and $a,b \in \{0,\ldots,l-1\}$. Recently, similar results were independently obtained by Tseng *et al.* [52] and Bros *et al.* [13] in their recent works. These two works indicate how to accurately estimate the behavior of the wedge map in the wedge attack on SNOVA, in particular how to reliably estimate the dimension of its kernel $\dim(\ker \mathcal{M})$ [52] or its rank $\operatorname{rank}(\mathcal{M})$ [13].

More specifically, the experiments on small parameter sets of Tseng *et al* [52] have shown that the Hilbert series

$$\frac{\prod_{i=1}^{l}(1+t_i)^n}{\prod_{i=1}^{l}(1+t_i^2)^{m_1}\prod_{1\le i<j\le l}(1+t_it_j)^{2m_1}}$$

can accurately predict the dimension of the kernel of the wedge map $\dim(\ker \mathcal{M})$. For more details, we refer to [52].

On the other hand, in [13], the authors proposed the unbalanced project-down technique. It is possible to use the framework of the generating function introduced in [52] to describe how to accurately estimate the dimension of the kernel of the wedge map of SNOVA with unbalanced project down. Utilizing the framework of [52], the dimension of the kernel of the wedge map can be estimated by the generating function:

$$H(\mathbf{t}) = \frac{\prod_{i=1}^{l}(1+t_i)^{n_i}}{\prod_{i=1}^{l}(1+t_i^2)^{m_1}\prod_{1\le i<j\le l}(1+t_it_j)^{2m_1}}.$$

where $n_i = v + o_i'$ and $\lceil \max(\frac{v}{m_1}, \frac{2v}{lm_1}+1)\rceil \le o_1' \le \cdots \le o_l' \le o$. Therefore, the complexity of the wedge attack is given by

$$\mathrm{Comp}_{\mathsf{Wedge}} = \min_{\mathbf{o}'\in A} \ 3\cdot(v+1)^2 \cdot \prod_{i=1}^{l}\binom{v+o_i'}{o_i'} \cdot g(q^l)$$

where $A = \{\mathbf{o}' = (o_1',\ldots,o_l') \in \mathbb{Z}_{\ge 0}^l \mid \lceil \max(\frac{v}{m_1}, \frac{2v}{lm_1}+1)\rceil \le o_1' \le \cdots \le o_l' \le o,\ H[\mathbf{t}^{\mathbf{o}'}] \le 1\}$.

Odd characteristic. In the case that $\mathbb{F}_q$ is of odd characteristic. The wedge map of SNOVA is defined as

$$\mathcal{M} : \bigwedge^{lv} \mathbb{F}_q^{ln} \to \left(\bigwedge^{lv+2} \mathbb{F}_q^{ln}\right)^{l^2m_1}, \quad \omega \mapsto \left(\omega \wedge \widehat{Q}_i^{(a,b)}\right)_{\substack{i\in\{1,\ldots,m_1\}\\ a,b\in\{0,\ldots,l-1\}}}.$$

Here, the $\widehat{Q}_i^{(a,b)} \in \bigwedge^2 \mathbb{F}_q^{ln}$ is the exterior 2-form corresponding to the bilinear form

$$Q_i^{(a,b)}(\mathbf{x},\mathbf{y}) \ = \mathbf{x}^\top \cdot \left(\mathbf{P}_{i,a,b} - (\mathbf{P}_{i,a,b})^\top\right) \cdot \mathbf{y}$$

where $i \in \{1,\ldots,m_1\}$ and $a,b \in \{0,\ldots,l-1\}$. In our private communication with Lars Ran [46], we arrived at the following Hilbert series

$$\prod_{i=1}^{l}(1+t_i)^n \prod_{1\le i,j\le l}(1-t_it_j)^{m_1}.$$

Based on our experimental results on small parameter sets, we found that the Hilbert series estimates the value of $\dim\ker(\mathcal{M})$ well and conservatively. Together with the unbalanced project

down technique, the dimension of the kernel of the wedge map can be estimated by the generating function:

$$H(\mathbf{t}) = \prod_{i=1}^{l}(1+t_i)^{n_i} \prod_{1\le i,j\le l}(1-t_it_j)^{m_1}.$$

where $n_i = v + o'_i$ and $\lceil \max(\frac{v}{m_1}, \frac{2v}{lm_1}+1)\rceil \le o'_1 \le \cdots \le o'_l \le o$. Therefore, the complexity of the wedge attack is given by

$$\mathrm{Comp}_{\mathsf{Wedge}} = \min_{\mathbf{o}'\in A}\ 3\cdot(v+1)^2 \cdot \prod_{i=1}^{l}\binom{v+o'_i}{o'_i}\cdot g(q^l)$$

where $A = \{\mathbf{o}' = (o'_1,\ldots,o'_l) \in \mathbb{Z}^l_{\ge 0} |\ \lceil \max(\frac{v}{m_1}, \frac{2v}{lm_1}+1)\rceil \le o'_1 \le \cdots \le o'_l \le o,\ H[\mathbf{t}^{\mathbf{o}'}] \le 1\}$. Note that the even-characteristic and odd-characteristic cases differ only in their generating functions.

Remark 5.6. In [18], Ding *et al.* proposed a reformulation of the wedge attack within a cleaner algebraic-geometric framework and applied the resulting multi-homogeneous wedge attack to the security analysis of SNOVA. Their security analysis of the wedge attack of even characteristic case is consistent with the wedge attack discussed in this section.

Remark 5.7. In [28], Ikematsu and Furue proposed a new method for recovering the oil vector of UOV variants in the context of the wedge attack when $v \ge 2m$. Their method is also applicable to lifted SNOVA. However, due to the constraints of their approach, its range of applicability to SNOVA is substantially narrower than that of the wedge attack discussed in this section. We confirm that none of our Round 3 recommended or alternative parameter sets fall within the applicable range of their method.

5.3.6 Key Recovery Attack Using p^ℓ-truncated Polynomial Rings

In [22], Furue *et al.* introduced a variant of the XL algorithm over finite fields $\mathbb{F}_q$ with $q = p^e$. Their approach leverages the p^ℓ-truncated polynomial ring $R^{(p^\ell)} = \mathbb{F}_q[x_1,\ldots,x_N]/\langle x_1^{p^\ell},\ldots,x_N^{p^\ell}\rangle$ for $1 \le \ell \le e$, enabling the construction of smaller Macaulay matrices in key-recovery attacks against UOV variants, such as, the reconciliation attack and the intersection attack. Consequently, the overall complexity of these attacks can be reduced.

We now summarize the complexity analysis of the algorithm. Let $R := \mathbb{F}_q[\mathbf{x}]$ be the polynomial ring in N variables $\mathbf{x} = (x_1,\ldots,x_N)^\top$ over $\mathbb{F}_q$, where $q = p^e$ and p is a prime. Let $f_1(\mathbf{x}),\ldots,f_M(\mathbf{x}) \in R$ be M homogeneous quadratic polynomials. Assume that the solution set of the system

$$f_1(\mathbf{x}) = \cdots = f_M(\mathbf{x}) = 0$$

contains a vector space $\mathcal{A} \subset \mathbb{F}_q^N$ over $\mathbb{F}_q$ of dimension α, i.e., $\dim \mathcal{A} = \alpha$. We denote the number of monomials of degree d in the p^ℓ-truncated polynomial ring in N variables by $T(N, p^\ell, d)$ where

$$T(N,p^\ell,d) := \sum_{i=0}^{\min\{N,\lfloor d/p^\ell\rfloor\}} (-1)^i \binom{N}{i}\binom{d-ip^\ell+N-1}{N-1}.$$

We estimate the complexity of the attack based on the *Conjuncture 1* in [22]: With very high probability, the quadratic polynomials $f_1,\ldots,f_M$ are semi-regular on the p^ℓ-truncated polynomial ring $R^{(p^\ell)}$. Then, the Hilbert series of $\mathbb{F}_q[\mathbf{x}]/\langle x_1^{p^\ell},\ldots,x_N^{p^\ell}, f_1,\ldots,f_M\rangle$ is given by

$$H_{N,M}(t) = \frac{(1+t+\cdots+t^{p^\ell-1})^N}{(1+t^2+\cdots+t^{2(p^\ell-1)})^M}.$$

Therefore, the solving degree D with $2 \le D \le \alpha(p^\ell-1)$ of the algorithm can be estimated by the minimum integer d satisfying

$$H_{N,M}[t^d] \le T(\alpha, p^\ell, d).$$

As a result, the number of operations required for the algorithm is estimated by

$$3 \cdot \binom{N+1}{2} \Big(T(N, p^\ell, D) - T(\alpha, p^\ell, D) + 1 \Big)^2.$$

For more details of the algorithm, we refer to [22]. In the rest of this subsection, we describe the analysis of SNOVA against the key recovery attacks using p^ℓ-truncated polynomial rings.

p^ℓ-truncated reconciliation attack. The p^ℓ-truncated reconciliation attack over the base field $\mathbb{F}_q$ solves the system

$$\mathbf{x}^\top \cdot \mathbf{P}_{i,a,b} \cdot \mathbf{x} = \mathbf{x}^\top \cdot (\mathbf{S}^{\otimes n})^a \cdot \mathbf{P}_i \cdot (\mathbf{S}^{\otimes n})^b \cdot \mathbf{x} = 0$$

with $ln = l(v+o)$ variables and $l^2 m_1$ equations, where the dimension of the solution space is lo. According to the analysis of [22], the algorithm chooses two parameters ℓ and k with $1 \le \ell \le e$ and $0 \le k < lo$. Here, we have $q = p^e$ and p is a prime. We take the solving degree D as the minimum integer $2 \le d \le (lo-k)(p^\ell - 1)$ satisfying

$$H_{N,M}[t^d] \;\le\; T(lo-k, p^\ell, d)$$

with $N = ln - k$ and $M = l^2 m_1$. Then, we estimate the complexity of p^ℓ-truncated reconciliation attack by

$$\mathrm{Comp}_{\mathsf{TrunRec}} = 3 \cdot \binom{ln-k+1}{2} \Big(T(ln-k, p^\ell, D) - T(lo-k, p^\ell, D) + 1 \Big)^2.$$

p^ℓ-truncated intersection attack. The p^ℓ-truncated intersection attack over the base field $\mathbb{F}_q$ solves the system

$$\begin{cases} \left(\mathbf{L}_1^{-1}\mathbf{y}\right)^\top \cdot (\mathbf{S}^{\otimes n})^a \cdot \mathbf{P}_i \cdot (\mathbf{S}^{\otimes n})^b \cdot \left(\mathbf{L}_1^{-1}\mathbf{y}\right) = 0 \\ \left(\mathbf{L}_1^{-1}\mathbf{y}\right)^\top \cdot (\mathbf{S}^{\otimes n})^a \cdot \mathbf{P}_i \cdot (\mathbf{S}^{\otimes n})^b \cdot \left(\mathbf{L}_2^{-1}\mathbf{y}\right) = 0 \\ \left(\mathbf{L}_2^{-1}\mathbf{y}\right)^\top \cdot (\mathbf{S}^{\otimes n})^a \cdot \mathbf{P}_i \cdot (\mathbf{S}^{\otimes n})^b \cdot \left(\mathbf{L}_1^{-1}\mathbf{y}\right) = 0 \\ \left(\mathbf{L}_2^{-1}\mathbf{y}\right)^\top \cdot (\mathbf{S}^{\otimes n})^a \cdot \mathbf{P}_i \cdot (\mathbf{S}^{\otimes n})^b \cdot \left(\mathbf{L}_2^{-1}\mathbf{y}\right) = 0 \end{cases}$$

with $ln = l(v+o)$ variables and $4l^2 m_1 - 2l$ equations, where the dimension of the solution space is 1 (since our parameter sets satisfy the relation $v > 2o$). Then, we estimate the solving degree D as the minimum integer $2 \le d \le p^\ell - 1$ satisfying

$$H_{N,M}[t^d] \le T(1, p^\ell, d) = 1,$$

with $N = ln$, $M = 4l^2 m_1 - 2l$ and the complexity is estimated as

$$\mathrm{Comp}_{\mathsf{TrunInt}} = 3 \cdot q^{v-2o+1} \cdot \binom{ln+1}{2} \cdot T(ln, p^\ell, D)^2.$$

p^ℓ-truncated attacks with lifted SNOVA. In [22], the authors pointed out that if $D \le (N - 2M)(p^\ell - 1)$ then the above estimation is not expected to be accurate. It is because of the occurrence of the unexpected elements. This case is corresponding to the key recovery attacks against lifted SNOVA. A similar discussion can be found in the wedge attack [47]. Our complexity estimations have taken this into account. Therefore, we only provide the estimation of the p^ℓ-truncated key recovery attacks over base field $\mathbb{F}_q$. On the other hand, as noted by the authors in [22], the key recovery attacks using p^ℓ-truncated polynomial ring are generally more efficient over smaller truncated polynomial rings, in particular, the case of $\ell = 1$. For the p^ℓ-truncated reconciliation attack against lifted SNOVA, the most efficient case, $\ell = 1$, is almost equivalent to the wedge attack with unbalanced project down technique over extension field $\mathbb{F}_{q^l}$. This case is already considered in Section 5.3.5.

In [18], Ding *et al.* demonstrated how to execute the p^ℓ-truncated key-recovery attacks over the extension field $\mathbb{F}_{q^\ell}$ against lifted SNOVA. Due to computational resource limitations, performing

an exhaustive search to obtain complete complexity estimates for the proposed attack is currently impractical. Based on our previous analysis, we believe that the $\ell = 1$ case is likely to represent the most efficient variant of the attack. We have evaluated this case for all of our recommended parameter sets and found no feasible solving degree that would enable a successful attack. For the cases with $\ell > 1$, compared to the p^ℓ-truncated attacks over the base field $\mathbb{F}_q$, we expect the resulting reduction in attack complexity to be minor. Given the substantial security margin of our recommended parameter sets against known key-recovery attacks and the relatively small number of oil variables they employ, we expect these parameter sets to remain secure against the p^ℓ-truncated attacks with $\ell > 1$.

5.4 Side-Channel Attacks

The optimized implementation of SNOVA is constant time. Valgrind reports secret-dependent behavior when generating the KAT files, but all of these messages are due to rejection sampling. Recently, a number of papers have appeared that study additional possibilities for side-channel attacks on SNOVA [4, 6, 38, 53]. Masking of the Gaussian elimination step of the sign function is discussed in [38] for multiple NIST Round 2 candidates. Hardware related attacks specifically for SNOVA are studied in [6] and for more UOV signature schemes in [4]. At this moment, we have not implemented measures specific to hardware attacks. The specification and the implementation of SNOVA will be updated whenever a proposed countermeasure is considered to be either necessary or cost-effective in terms of impact on the performance.

In [49] an attack is considered where the attacker has gained partial knowledge of the secret key through some side-channel attacks such as cold-boot attacks. To the best of our knowledge, this side-channel attack is not the most effective attack on SNOVA.

6 Advantages and Limitations (2.B.6)

6.1 Advantages

The main advantages of SNOVA are summarized as follows:

- **Compact public keys and signatures:** As a MQ-based signature scheme, SNOVA retains the characteristically short signatures of multivariate constructions while substantially mitigating the large-public-key problem encountered in traditional MQ schemes. Cloudflare's study [59, 58] of post-quantum authentication in TLS suggests that timely deployment would be facilitated if six signatures and two public keys could collectively fit within approximately 9 KB, corresponding to the practical size criterion

$$6 \cdot |\mathbf{sig}| + 2 \cdot |\mathbf{pk}| \leq 9\,\mathrm{KB}.$$

  According to the SNOVA Round 3 specification, all recommended parameter sets, including those targeting NIST security level V, satisfy this criterion. SNOVA is therefore a promising candidate for general-purpose applications, including bandwidth-sensitive protocols such as TLS.

- **Modest computational requirements and fast verification:** Signing and verification are implemented using relatively simple matrix and finite-field operations. These operations can be efficiently accelerated on modern processors using instruction sets such as AVX2 and GFNI. At the same time, the absence of computationally expensive integer, lattice, or transcendental arithmetic makes SNOVA suitable for implementation on mobile and other resource-constrained platforms.

- **Small stored secret key:** The secret key can be compactly represented using cryptographic seeds. In the Round 3 implementation, the stored secret key consists of a public-key seed, a secret-key seed, and a hash of the public key. Its total size is exactly 96 bytes for every proposed parameter set.

- **High parameter flexibility and a wide security margin:** The Round 3 SNOVA supports rectangular signatures and extends the specification to base fields of odd characteristic. The resulting framework is described by seven parameters,

$$(v, o, q, l, r, m_1, m_2).$$

  In particular, the auxiliary structural parameter r and the number m_2 of public equations can be selected independently. This structural decoupling provides considerable flexibility for balancing public-key size, signature size, computational performance, and resistance to known attacks. It also enables the selection of conservative parameter sets with security margins substantially exceeding the minimum requirements of their target NIST security levels.

- **Simple and transparent arithmetic structure:** Although SNOVA is conceptually constructed using noncommutative matrix rings, its concrete operations consist primarily of standard finite-field linear algebra. Its architecture may be viewed as a UOV-type construction augmented by carefully designed noncommutative matrix structure. Consequently, the scheme remains conceptually accessible and amenable to detailed algebraic and implementation-level analysis.

6.2 Limitations

The principal limitations of SNOVA are as follows:

- **Absence of a reductionist security proof:** As with most practical MQ-based signature schemes, the security of SNOVA is currently supported by cryptanalytic evidence rather than by a formal reduction to a standard, independently established hard problem. When the matrix-ring dimension is set to $l = 1$, the SNOVA construction reduces structurally to a standard UOV scheme. This provides a familiar and extensively studied baseline for understanding the construction. Nevertheless, this structural relationship does not by itself constitute a security proof for the noncommutative cases $l > 1$, which must be evaluated against both generic MQ attacks and SNOVA-specific structural attacks.
- **Parameter-dependent performance trade-offs:** SNOVA necessarily involves trade-offs among public-key size, signature size, signing speed, and verification speed. For example, increasing the matrix-ring dimension l can substantially reduce the public-key size, but generally increases the signature size and introduces additional matrix arithmetic. It may therefore result in a moderate reduction in signing and verification performance. Parameter selection must consequently be adapted to the intended deployment environment rather than optimized according to a single metric.
- **Further cryptanalysis required for symmetric public matrices:** The extension to fields of odd characteristic creates the possibility of representing certain public-key coefficient matrices symmetrically, potentially yielding additional public-key compression. However, imposing symmetry changes the algebraic distribution and structural properties of the public key and may invalidate assumptions used in the existing security analysis. Parameter sets employing symmetric public matrices are therefore not currently recommended for standardization and should remain experimental until they have undergone sufficiently broad and mature cryptanalysis. We leave this as our future work.
- **Potential for new structural attacks:** SNOVA is a relatively new signature scheme. Although it has already received considerable research attention, new attacks may still emerge. In particular, because SNOVA exploits additional algebraic structure to reduce public-key and signature sizes, this structure may provide avenues for previously unknown attacks. The Round 3 parameter sets were therefore selected conservatively, with substantial security margins to mitigate the impact of potential improvements in existing attacks or the discovery of new structural attacks.

Acknowledgment

We would like to express our sincere gratitude to Simona Samardjiska and Lars Ran for the many valuable discussions and exchanges regarding the wedge attack and their insight into the parameters and performances. Their insights and feedback greatly contributed to our understanding of the subject. We are also thankful for the constructive discussions on the security of SNOVA over odd-characteristic fields. We are very grateful for their suggestions and thoughtful comments.

The SNOVA Team would like to express our heartfelt gratitude to Ya-Yun Zheng for designing the SNOVA logo. We sincerely thank her for creating such an iconic, distinctive, and vibrant logo. We are especially grateful for the considerable time, creativity, and dedication she devoted to bringing this design to life.

References

[1] Gorjan Alagic, Pierre Ciadoux Maxime Bros, Quynh Dang, Thinh Hung Dang, John Kelsey, Jacob Lichtinger, Yi-Kai Liu, Carl Miller, Dustin Moody, Rene Peralta, Ray Perlner, Angela Robinson, Hamilton Silberg, Daniel Smith-Tone, and Noah Waller. Status report on the second round of the additional digital signature schemes for the nist post-quantum cryptography standardization process. (national institute of standards and technology, gaithersburg, md) nist internal report (ir) nist ir 8610, 2026.

[2] Hiroshi Amagasa, Hiroki Furue, Rei Ueno, and Naofumi Homma. QR-UOV without rejection sampling: Security analysis and high-speed implementation. Cryptology ePrint Archive, Paper 2026/527, 2026.

[3] Hideki Asanuma, Yilong Chen, Hiroki Furue, Kosuke Sakata, and Tsuyoshi Takagi. Improved complexity estimates for underdetermined MQ systems via generalized variable partitioning. Cryptology ePrint Archive, Paper 2026/1054, 2026.

[4] Thomas Aulbach, Fabio Campos, and Juliane Krämer. Sok: On the physical security of uov-based signature schemes. In Ruben Niederhagen and Markku-Juhani O. Saarinen, editors, *Post-Quantum Cryptography*, pages 199–231, Cham, 2025. Springer Nature Switzerland.

[5] Thomas Aulbach, Samed Düzlü, Michael Meyer, Patrick Struck, and Maximiliane Weishäupl. Hash your keys before signing. In Markku-Juhani Saarinen and Daniel Smith-Tone, editors, *Post-Quantum Cryptography*, pages 301–335, Cham, 2024. Springer Nature Switzerland.

[6] Gustavo Banegas and Ricardo Villanueva-Polanco. A fault analysis on SNOVA. Cryptology ePrint Archive, Paper 2024/1883, 2024.

[7] Grégory Berhuy. Minimal and characteristic polynomials of symmetric matrices in characteristic two. *Journal of Algebra*, 593:525–549, 2022.

[8] Luk Bettale, Jean-Charles Faugere, and Ludovic Perret. Hybrid approach for solving multivariate systems over finite fields. *Journal of Mathematical Cryptology*, 3(3):177–197, 2009.

[9] Ward Beullens. Improved cryptanalysis of SNOVA. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 277–293. Springer, 2025.

[10] Ward Beullens, Fabio Campos, Sofía Celi, Basil Hess, and Matthias J. Kannwischer. Mayo. *NIST Computer Security Resource Center (CSRC)*, 2025.

[11] Ward Beullens, Ming-Shing Chen, Jintai Ding, Boru Gong, Matthias J. Kannwischer, Jacques Patarin, Bo-Yuan Peng, Dieter Schmidt, Cheng-Jhih Shih, Chengdong Tao, and Bo-Yin Yang. Uov: Unbalanced oil and vinegar. *NIST Computer Security Resource Center (CSRC)*, 2025.

[12] Charles Bouillaguet, Hsieh-Chung Chen, Chen-Mou Cheng, Tung Chou, Ruben Niederhagen, Adi Shamir, and Bo-Yin Yang. Fast exhaustive search for polynomial systems in $\mathbb{F}_2$. In Stefan Mangard and François-Xavier Standaert, editors, *Cryptographic Hardware and Embedded Systems, CHES 2010*, pages 203–218, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[13] Maxime Bros, Thai Hung Le, Jacob Lichtinger, Brice Minaud, Ray Perlner, Daniel Smith-Tone, and Cristian Valenzuela. Exploiting SNOVA's structure in the wedge product attack. *Cryptology ePrint Archive, Paper 2026/237*, 2026.

[14] Daniel Cabarcas, Peigen Li, Javier Verbel, and Ricardo Villanueva-Polanco. Improved attacks for snova by exploiting stability under a group action. In Yael Tauman Kalai and Seny F. Kamara, editors, *Advances in Cryptology – CRYPTO 2025*, pages 358–389, Cham, 2025. Springer Nature Switzerland.

[15] Nicolas Courtois, Alexander Klimov, Jacques Patarin, and Adi Shamir. Efficient algorithms for solving overdefined systems of multivariate polynomial equations. In Bart Preneel, editor, *Advances in Cryptology — EUROCRYPT 2000*, pages 392–407, Berlin, Heidelberg, 2000. Springer Berlin Heidelberg.

[16] Cas Cremers, Samed Düzlü, Rune Fiedler, Marc Fischlin, and Christian Janson. BUFFing signature schemes beyond unforgeability and the case of post-quantum signatures. Cryptology ePrint Archive, Paper 2020/1525, 2020.

[17] Jintai Ding, Hao Guo, Yen-Liang Kuan, Jan Adriaan Leegwater, Peigen Li, Po-En Tseng, and Lih-Chung Wang. Reformulating the SNOVA signature scheme. Cryptology ePrint Archive, Paper 2026/659, 2026.

[18] Jintai Ding, Peigen Li, and Siyong Tao. Revisiting the wedge attack on UOV problem. Cryptology ePrint Archive, Paper 2026/1548, 2026.

[19] Jintai Ding and Bo-Yin Yang. Multivariate public key cryptography. In *Post-quantum cryptography*, pages 193–241. Springer, 2009.

[20] Jean-Charles Faugere. A new efficient algorithm for computing Gröbner bases (F4). *Journal of pure and applied algebra*, 139(1-3):61–88, 1999.

[21] Jean-Charles Faugere. A new efficient algorithm for computing Gröbner bases without reduction to zero (F5). In *Proceedings of the 2002 international symposium on Symbolic and algebraic computation*, pages 75–83, 2002.

[22] Hiroki Furue and Yasuhiko Ikematsu. Key recovery attacks on UOV using p^l-truncated polynomial rings. Cryptology ePrint Archive, Paper 2026/298, 2026.

[23] Hiroki Furue, Yasuhiko Ikematsu, Fumitaka Hoshino, Tsuyoshi Takagi, Haruhisa Kosuge, Kimihiro Yamakoshi, Rika Akiyama, Satoshi Nakamura, Shingo Orihara, and Koha Kinjo. QR-UOV. *NIST Computer Security Resource Center (CSRC)*, 2025.

[24] Hiroki Furue, Yasuhiko Ikematsu, Shuhei Nakamura, and Rika Akiyama. Improved cryptanalysis of snova by solving multi-homogeneous systems via matrix transformations. In Goichiro Hanaoka and Bo-Yin Yang, editors, *Advances in Cryptology – ASIACRYPT 2025*, pages 405–435, Singapore, 2026. Springer Nature Singapore.

[25] Juris Hartmanis. Computers and intractability: a guide to the theory of np-completeness (michael r. garey and david s. johnson). *Siam Review*, 24(1):90, 1982.

[26] Yasufumi Hashimoto. Minor improvements of algorithm to solve under-defined systems of multivariate quadratic equations. *IACR Cryptol. ePrint Arch.*, 2021:1045, 2021.

[27] Yasuhiko Ikematsu and Rika Akiyama. Revisiting the security analysis of SNOVA. In *Proceedings of the 11th ACM Asia Public-Key Cryptography Workshop*, pages 54–61, 2024.

[28] Yasuhiko Ikematsu and Hiroki Furue. Extending the applicability of algebraic key recovery attacks on the UOV signature scheme. Cryptology ePrint Archive, Paper 2026/1620, 2026.

[29] Yi Jin, Yuansheng Pan, Xiaoou He, Boru Gong, and Jintai Ding. Security analysis on UOV families with odd characteristics: Using symmetric algebra. Cryptology ePrint Archive, Paper 2025/1137, 2025.

[30] Aviad Kipnis, Jacques Patarin, and Louis Goubin. Unbalanced oil and vinegar signature schemes. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 206–222. Springer, 1999.

[31] Aviad Kipnis and Adi Shamir. Cryptanalysis of the oil and vinegar signature scheme. In *Annual international cryptology conference*, pages 257–266. Springer, 1998.

[32] Fred Krakowski. Eigenwerte und minimalpolynome symmetrischer matrizen in kommutativen körpern. *Commentarii Mathematici Helvetici*, 32:224–240, 1958.

[33] Peigen Li and Jintai Ding. Cryptanalysis of the SNOVA signature scheme. In *International Conference on Post-Quantum Cryptography*, pages 79–91. Springer, 2024.

[34] Peigen Li, Hao Guo, and Jintai Ding. Unified approach to UOV-like multivariate signature schemes. Cryptology ePrint Archive, Paper 2025/1778, 2025.

[35] Alexander May, Massimo Ostuzzi, and Henrik Ressler. Just guess: Improved (quantum) algorithm for the underdetermined MQ problem. Cryptology ePrint Archive, Paper 2025/1788, 2025.

[36] Shuhei Nakamura, Yusuke Tani, and Hiroki Furue. Lifting approach against the snova scheme. In *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, 2025, Volume E108.A, Issue 10*, pages 1373–1381, 2025.

[37] Ivica Nikolić and Yu Sasaki. A new algorithm for the unbalanced meet-in-the-middle problem. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *Advances in Cryptology – ASIACRYPT 2016*, pages 627–647, Berlin, Heidelberg, 2016. Springer Berlin Heidelberg.

[38] Quinten Norga, Suparna Kundu, Uttam Kumar Ojha, Anindya Ganguly, Angshuman Karmakar, and Ingrid Verbauwhede. Masking gaussian elimination at arbitrary order with application to multivariate-and code-based pqc. In Arpita Patra, editor, *Topics in Cryptology – CT-RSA 2025*, pages 249–272, Cham, 2025. Springer Nature Switzerland.

[39] National Institute of Standards and Technology (NIST). Nist: Submission requirements and evaluation criteria for the post-quantum cryptography standardization process.

[40] National Institute of Standards and Technology (NIST). Post-quantum cryptography: Digital signature schemes, 2023.

[41] Massimo Ostuzzi. Pseudo-oil subspaces and the geometry of underdetermined MQ problems. Cryptology ePrint Archive, Paper 2026/1122, 2026.

[42] Jacques Patarin. The oil and vinegar algorithm for signatures. In *Dagstuhl Workshop on Cryptography, 1997*, 1997.

[43] Albrecht Petzoldt. *Selecting and reducing key sizes for multivariate cryptography*. PhD thesis, Technische Universität Darmstadt, 2013.

[44] Thomas Pornin and Julien P. Stern. Digital signatures do not guarantee exclusive ownership. In John Ioannidis, Angelos Keromytis, and Moti Yung, editors, *Applied Cryptography and Network Security*, pages 138–150, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

[45] Lars Ran. Wedges, oil, and vinegar – an analysis of UOV in characteristic 2. Cryptology ePrint Archive, Paper 2025/1143, 2025.

[46] Lars Ran. Private communication, 2026.

[47] Lars Ran. Wedges, oil, and vinegar. In Joan Daemen and Emmanuel Thomé, editors, *Advances in Cryptology – EUROCRYPT 2026*, pages 363–388, Cham, 2026. Springer Nature Switzerland.

[48] Koichi Sakumoto, Taizo Shirai, and Harunaga Hiwatari. On provable security of uov and hfe signature schemes against chosen-message attack. In Bo-Yin Yang, editor, *Post-Quantum Cryptography*, pages 68–82, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[49] Yuki Seto, Hiroki Furue, and Atsushi Takayasu. Partial key exposure attacks on UOV and its variants. Cryptology ePrint Archive, Paper 2025/595, 2025.

[50] SNOVA Team. Official SNOVA reference and optimized implementation. https://github.com/PQCLAB-SNOVA/SNOVA, 2026.

[51] Toshihiro Suzuki, Hiroki Furue, Takuma Ito, Shuhei Nakamura, and Shigenori Uchiyama. An extended rectangular minrank attack against uov and its variants. In Carlos Cid and Naoto Yanai, editors, *Advances in Information and Computer Security*, pages 111–130, Singapore, 2026. Springer Nature Singapore.

[52] Po-En Tseng, Lih-Chung Wang, Peigen Li, and Yen-Liang Kuan. Investigating the wedge map on SNOVA. *Cryptology ePrint Archive, Paper 2026/260*, 2026.

[53] N.K. Vishwaajith, Anindya Ganguly, Debranjan Pal, Trevor Yap, Puja Mondal, Suparna Kundu, Sayandeep Saha, Shivam Bhasin, Ingrid Verbauwhede, and Angshuman Karmakar. SCA-MQDSA: Side-channel analysis of multivariate digital signature implementations. Cryptology ePrint Archive, Paper 2026/228, 2026.

[54] Lih-Chung Wang, Chun-Yen Chou, Jintai Ding, Yen-Liang Kuan, Jan Adriaan Leegwater, Ming-Siou Li, Bo-Shu Tseng, Po-En Tseng, and Chia-Chun Wang. A note on the snova security. *NIST Computer Security Resource Center (CSRC)*, 2025.

[55] Lih-Chung Wang, Chun-Yen Chou, Jintai Ding, Yen-Liang Kuan, Jan Adriaan Leegwater, Ming-Siou Li, Bo-Shu Tseng, Po-En Tseng, and Chia-Chun Wang. On the security of round 2 snova. *NIST Computer Security Resource Center (CSRC)*, 2025.

[56] Lih-Chung Wang, Chun-Yen Chou, Jintai Ding, Yen-Liang Kuan, Jan Adriaan Leegwater, Ming-Siou Li, Bo-Shu Tseng, Po-En Tseng, and Chia-Chun Wang. Snova. *NIST Computer Security Resource Center (CSRC)*, 2025.

[57] Lih-Chung Wang, Po-En Tseng, Yen-Liang Kuan, and Chun-Yen Chou. A Simple Noncommutative UOV Scheme. *Taiwanese Journal of Mathematics*, 30(3):445 – 479, 2026.

[58] Bas Westerbaan. Sizing up post-quantum signatures. Cloudflare blog, 2021.

[59] Thom Wiggers. Making protocols post-quantum. Cloudflare blog, 2022.