NEV: Faster and Smaller NTRU Encryption Using Vector Decoding

Jiang Zhang(✉), Dengguo Feng, and Di Yan

State Key Laboratory of Cryptology, P.O. Box 5159, Beijing 100878, China
{zhangj,yand}@sklc.org, fengdg@263.net

Abstract. In this paper, we present NEV – a faster and smaller NTRU Encryption using Vector decoding, which is provably IND-CPA secure in the standard model under the decisional NTRU and RLWE assumptions over the cyclotomic ring $R_q = \mathbb{Z}_q[X]/(X^n + 1)$. Our main technique is a novel and non-trivial way to integrate a previously known plaintext encoding and decoding mechanism into the provably IND-CPA secure NTRU variant by Stehlé and Steinfeld (Eurocrypt 2011). Unlike the original NTRU encryption and its variants which encode the plaintext into the least significant bits of the coefficients of a message polynomial, we encode each plaintext bit into the most significant bits of multiple coefficients of the message polynomial, so that we can use a vector of noised coefficients to decode each plaintext bit in decryption, and significantly reduce the size of q with a reasonably negligible decryption failure.

Concretely, we can use $q = 769$ to obtain public keys and ciphertexts of 615 bytes with decryption failure $\leq 2^{-138}$ at NIST level 1 security, and 1229 bytes with decryption failure $\leq 2^{-152}$ at NIST level 5 security. By applying the Fujisaki-Okamoto transformation in a standard way, we obtain an IND-CCA secure KEM from our basic PKE scheme. Compared to NTRU and Kyber in the NIST Round 3 finalists at the same security levels, our KEM is 33–48% more compact and 5.03–29.94X faster than NTRU in the round-trip time of ephemeral key exchange, and is 21% more compact and 1.42–1.74X faster than Kyber.

We also give an optimized encryption scheme NEV′ with better noise tolerance (and slightly better efficiency) based on a variant of the RLWE problem, called Subset-Sum Parity RLWE problem, which we show is polynomially equivalent to the standard decisional RLWE problem (with different parameters), and maybe of independent interest.

1 Introduction

The NTRU encryption proposed by Hoffstein, Pipher and Silverman [24] is one of the first publicly known practical public key encryptions (PKEs) on lattices. The security of NTRU encryption was originally stated as its own assumption, but after more than 25 years of studies, there is no significant algorithmic progress against it (except for overstretched parameters [17,29]). Now, it is more natural

J. Guo and R. Steinfeld (Eds.): ASIACRYPT 2023, LNCS 14444, pp. 157–189, 2023.
https://doi.org/10.1007/978-981-99-8739-9_6

to view NTRU encryption as a cryptosystem based on two hardness assumptions [18,43]: the decisional NTRU assumption which roughly says that the quotient $h = g/f$ of two small polynomials g, f is pseudorandom, and the RLWE assumption [32,44] which says that it is hard to recover e from $(h, hr + e)$ when h is uniformly random, and r, e are randomly chosen small polynomials. It is worth to note that the first assumption can be removed for appropriately chosen (but very inefficient) parameters [43].

In NIST post-quantum cryptography (PQC) standardization process [36], NTRU was one of the four PKEs/KEMs in NIST Round 3 finalists [37], but it was not selected for standardization by NIST in the end [38]. One main reason is that it is neither the fastest nor the smallest among the lattice KEM finalists [38]. In particular, compared to Kyber which was selected as the NIST KEM standard, NTRU has 8.3–18.6% larger public key and ciphertext sizes (see Table 1) and 8.21–45.34X slower key generation (see Table 2). Several recent efforts [18,20,33] have been made to improve the performance of NTRU.

Lyubashevsky and Seiler [33] proposed a NTRU variant, called NTTRU, over the specific cyclotomic ring $\mathbb{Z}_{7681}[x]/(x^{768} - x^{384} + 1)$ that supports Number Theory Transform (NTT), and obtained significant speedup over the original NTRU that uses rings (e.g., $\mathbb{Z}_q[x]/(x^n - 1)$) do not support NTT. Later, Duman et al. [18] extended the idea of [33] to other NTT-friendly rings of the same form $\mathbb{Z}_q[x]/(x^n - x^{n/2} + 1)$, and obtained comparable efficiency improvement for flexible choices of parameters. Note that given an NTRU public key $h = pg/f$ for some plaintext modulus p, the message m in the original NTRU encryption $c = hr + m$ will be multiplied by the secret f in decryption. Thus, purposefully choosing a "bad" m can significantly increase the decryption failure (by more than 2^{100} times for standard parameter choices [18]), which might be utilized by the adversary in a decryption failure attack to obtain information of f. To resist this attack, the authors [18] also provide three transformations to detach the decryption failure from the message. One of their main transformation called NTRU-A (that is used in comparison with related works in [18, Table 3]) requires a new assumption called RLWE2, which is closely related to the RLWE problem, but the authors only provide heuristic arguments to the equivalence of RLWE2 and RLWE [18]. Despite of the efficiency improvement, the sizes of [18,33] are still larger than that of Kyber at the same security levels (see Table 3).

Fouque et al. [20] proposed another NTRU variant, called BAT, with a GGH-like encryption and decryption paradigm over the power of 2 cyclotomic ring $\mathbb{Z}_q[x]/(x^n + 1)$, which requires a very complex trapdoor inversion algorithm. Compared to other NTRU schemes, BAT has the smallest sizes (see Table 3). But it has a very slow key generation, which is 266-2131X slower than Kyber, and is even 7-104X slower than NTRU (see Tables 2 and 5). Moreover, BAT needs a strong RLWR with binary secret assumption.

1.1 Our Results

We present a faster and smaller NTRU-like Encryption using Vector decoding, called NEV-PKE, which is provably IND-CPA secure under the decisional NTRU

and RLWE assumptions over the cyclotomic ring $R_q = \mathbb{Z}_q[X]/(X^n + 1)$ in the standard model, and thus can be directly used as a passively secure key exchange without resorting to the (quantum) random oracle model. Our main technique is a novel way to non-trivially integrate a previously known plaintext encoding and decoding mechanism [4,41] into the provably secure NTRU variant [43], which allows us to use a very small modulus q and obtain smaller public key and ciphertext sizes with a reasonably negligible decryption failure (see Sect. 1.2).

Concretely, the small modulus $q = 769$ can be used to achieve a decryption failure $\leq 2^{-138}$ for NIST level 1 security and $\leq 2^{-152}$ for NIST level 5 security. With a compressed representation of R_q elements (see Sect. 6.5), we can obtain public keys and ciphertexts of 615 and 1229 bytes respectively at the two security levels, which is 33–48% more compact than NTRU, and is 21% more compact than Kyber (see Table 1). By applying the Fujisaki-Okamoto transformation to NEV-PKE, we obtain an IND-CCA secure KEM called NEV-KEM. We implement our schemes using reference C language and AVX2 instructions in experiment. Due to the use of (partial) NTT multiplications and inversions in R_q (see Sects. 6.1 and 6.2), our NEV-KEM is 5.03–29.94X faster than NTRU and 1.42–1.74X faster than Kyber in the round-trip time of ephemeral key exchange.

We also give an optimized NTRU encryption called NEV-PKE$'$ with better noise tolerance based on a variant of the RLWE problem, called Subset-Sum Parity RLWE (sspRLWE) problem, which can also be seen as a generalization of the RLWE2 problem in [18]. We show that the sspRLWE problem is polynomially equivalent to the decisional RLWE problem (with different parameters), which partially solves the problem of proving the equivalence of RLWE2 and RLWE in [18]. By assuming that the concrete hardness of sspRLWE is equal to RLWE with the same parameters as for RLWE2 in [18], NEV-PKE$'$ can achieve a smaller decryption failure and slightly better performance than NEV-PKE. Concretely, we can use the same modulus $q = 769$ to achieve a decryption failure $\leq 2^{-200}$ at both NIST levels 1 and 5 security.

One nice feature which is worth to mention is that our schemes NEV-PKE and NEV-PKE$'$ are more robust than NTRU to a decryption failure attack because the plaintext has little contribution to the decryption noise in NEV-PKE, and the plaintext in NEV-PKE$'$ will essentially be masked using a random secret share algorithm (see Sect. 1.2 below). Similar to Newhope [4] that uses the power of 2 cyclotomic ring $\mathbb{Z}_q[x]/(x^n + 1)$, one possible limitation for our schemes is that we cannot find a proper parameter set for NIST level 3 security, but since our performance at NIST level 5 security is already comparable with existing schemes at NIST level 3 security (see Tables 1 and 2), we believe this would not be a real problem in practice.

1.2 Technical Overview

We begin by first recalling the original NTRU encryption. Formally, let n, q, p be three positive integers, and p coprime to q. Let $R_q = \mathbb{Z}_q/(x^n - 1)$. The public key h and ciphertext c of NTRU has forms of:

$$h = pg/f, \qquad c = hr + m,$$

where g, f, r are polynomials with small coefficients, m is the message polynomial. The decryption is done by first computing $u = fc = pgr + fm \in R_q$, and then computing $m = f^{-1}u \in R_p$. The decryption requires the ℓ_∞ norm of $pgr + fm$ to be smaller than $\frac{q-1}{2}$ (i.e., $\|pgr + fm\|_\infty < \frac{q-1}{2}$), and f invertible in both R_q and R_p for correctness, where p is typically equal to 3 for ternary message polynomial m. To simplify the decryption, f is usually set to have the form of $f = pf' + 1$ such that $f^{-1} \bmod p = 1$. In this case, we have $u = pgr + pf'm + m$, where the decryption noise $pgr + pf'm$ essentially has the same form to that of RLWE-based encryptions (except that m in the term $pf'm$ is replaced with a random error polynomial). There are two main reasons why NTRU has larger public keys and ciphertexts sizes than its RLWE-based counterparts: 1) when fixing all other parameters, the decryption noise with $p = 3$ in NTRU is 1.5X larger than that of its RLWE counterparts where $p = 2$ is typically used; and 2) the decryption failure for NTRU is more subtle because the term $pf'm$ in the decryption noise usually has the same magnitude as pgr, which may be utilized by the adversary in a decryption failure attack with a purposefully chosen "bad" message m. This is why NTRU [11] submitted to NIST PQC standardization sets its parameters to have no decryption failure.

Our basic idea is to use the plaintext encoding and decoding mechanism in [4,41] to increase the noise tolerance of NTRU, which basically encodes each plaintext bit into the most significant bit of multiple coefficients of the message polynomial, so that a vector of noised coefficients can be used to decode each plaintext bit in decryption. We note that this mechanism was, to the best of our knowledge, not used in NTRU and its variants before, because it is not quite compatible with the central features of NTRU: 1) m is required to be a random polynomial for the security of the ciphertext $c = hr + m$ (since m is directly used as the RLWE error); and 2) fm is required to be small for decryption correctness. We solve the above two technical issues by slightly modifying the key generation and the plaintext encoding/decoding of the provably IND-CPA secure NTRU variant [43] (whose security is independent from the message polynomial) with a small polynomial $v = (1 - x^{n/k})$, where n/k is the plaintext length and is fixed to be 256 for our interest.[1] Our construction crucially relies on the power of 2 cyclotomic ring $R_q = \mathbb{Z}_q[X]/(X^n + 1)$. In particular, $v = (1 - x^{n/k})$ has a nice inverse $v^{-1} = \frac{q+1}{2}(1 + x^{n/k} + \cdots + x^{(k-1)n/k}) \in R_q$, which will serve as our plaintext encoding polynomial. The public key and ciphertext of our NEV-PKE has forms of:

$$h = g/(vf' + 1), \qquad c = hr + e + v^{-1}m,$$

where g, f', r, e are small polynomials, and m is the plaintext polynomial only having non-zero binary coefficients in the first 256 coordinates. For decryption, we first compute $u = (vf' + 1)c = gr + vf'e + f'm + e + v^{-1}m$. Since $v^{-1}m \in R_q$ essentially copies $k = n/256$ times the first 256 coefficients of m to obtain n coefficients, we can use k coefficients in u to decode each plaintext bit in decryption (if $\|gr + vf'e + f'm + e\|_\infty \leq \frac{q-1}{4}$ holds with high probability) as

[1] We note that a 256-bit session key is sufficient for most real applications, and that the NIST PQC standard Kyber also only supports a 256-bit plaintext [9].

in [4,41]. The major reason that we can obtain a reasonably negligible decryption failure with very small modulus is because: 1) the magnitude of the major noise term $vf'e$ in our NEV-PKE is at least $\sqrt{2}$ times smaller than that of using $p = 2, 3$ or $x + 2$ in NTRU and its provable version [43]; 2) m has at most 256 non-zero binary coefficients; and 3) the use of vector decoding will lower the decryption failure (using a single coefficient) by roughly k times in the exponent.

We clarify that the slight modification of the public key in NEV-PKE will not require a stronger NTRU assumption because 1) the use of a polynomial $v = x+2$ was recommended by the authors of NTRU as early as 2000 [25] (note that $vf'+1$ is small if f' is small) and was investigated in [6,22,23,27,35,43]; 2) by replacing $v = (1-x^{n/k})$ with $v = p$ we recover the provably IND-CPA secure NTRU in [43], and the proof for the public key uniformity in [43, Theorem 3] mainly depends on the properties of the distributions of g and f', which essentially applies to any invertible $v \in R_q$ (even without changing any other parameters); and 3) the currently concrete security estimation also only cares about the distributions of g and f', since $v = (1 - x^{n/k})$ (or $v = p$) is invertible and publicly known which can be somehow removed in lattice attacks (see Sect. 5.1).

One nice feature of our NEV-PKE is that the magnitude of $f'm$ is much smaller than that of $gr + vf'e + e$ because m only has non-zero binary coefficients in the first 256 coordinates. This means that our NEV-PKE is more robust than NTRU to a decryption failure attack with maliciously chosen bad messages in generating ciphertexts. Experimentally, the best choice for the adversary to obtain a failure decryption in NEV-PKE is to use a message polynomial with all ones in the first 256 coordinates, which will only increase the decryption failure by a factor of 2^{21} and 2^{14} for parameters NEV-512 and NEV-1024, respectively (in contrast, NTRU has a factor more than 2^{100} for standard parameter choices [18]), which means that the resulting decryption failure (i.e., 2^{-117} for NEV-512 and 2^{-138} for NEV-1024) is still sufficiently small for a common restriction of at most 2^{64} decryption queries. We note that one can further remove this dependence on m by using the generic transformation (say, NTRU-C) with a small price of an extra 32 bytes in ciphertexts in [18].

An Optimization Based on the sspRLWE Assumption. Based on the observation that in the application of using PKEs as KEMs, the session key is randomly chosen and not necessarily known in advance, we also provide an optimized construction NEV-PKE$'$ which essentially merges the sampling of the encryption noise and the random session key in a single step: one can roughly think that the encryption noise is a random secret share of a random session key. Specifically, the public key and ciphertext of NEV-PKE$'$ has forms of

$$h = vg/(vf' + 1) = g/(f' + v^{-1}), \qquad c = hr + e,$$

where g, f', r, e are randomly chosen small polynomials. Note that by setting $v = p$, the above construction is essentially the same as the original NTRU

encryption. For decryption, we first compute $u = (f' + v^{-1})c = gr + f'e + v^{-1}e$. Let $\bar{v} = 1 + x^{n/k} + \cdots + x^{(k-1)n/k}$, $e_0 = \bar{v}e \bmod 2$, and $2e_1 = \bar{v}e - e_0$, we have

$$v^{-1} = \frac{q+1}{2}\bar{v}, \quad v^{-1}e = e_1 + \frac{q+1}{2}e_0 \in R_q, \text{ and } u = gr + f'e + e_1 + \frac{q+1}{2}e_0 \in R_q.$$

Let m be a polynomial only having $n/k = 256$ non-zero coefficients that are equal to the first 256 coefficients of e_0. By the nice property of $R_q = \mathbb{Z}_q[x]/(x^n+1)$ and the choice of $\bar{v}$ (and v^{-1}), it is easy to check that e_0 is essentially a polynomial which copies $k = n/256$ times the first 256 coefficients of m (and thus itself) to obtain n coefficients. Hence, we can use the vector decoding technique [4,41] again to recover m from u, and output m as the session key. Clearly, the decryption noise $gr + f'e + e_1$ in NEV-PKE′ is much smaller than that of NEV-PKE.

To obtain an IND-CCA secure KEM, we have to convert NEV-PKE′ into a PKE where m (or equivalently $\bar{v}e \bmod 2$) is determined before e. Since $\bar{v}e$ essentially adds k coefficients (with $\pm$ signs) of e to a single coefficient, we can easily achieve the goal of "inverting $\bar{v}e \bmod 2$ to obtain e" by using binomial noise distribution B_η. Take $\eta = 1$ and $k = 2$ as an example, we can "invert" a plaintext bit $b^* \in \{0,1\}$ to 2 samples from B_1 as follows: randomly choose $b_1, b_2, b_3 \leftarrow \{0,1\}$, set $b_0 = b^* \oplus b_1 \oplus b_2 \oplus b_3$, and output $e_0 = b_0 - b_1, e_1 = b_2 - b_3$. It is easy to check that $e_0 \pm e_1 \bmod 2 = b^*$, and $e_0, e_1 \sim B_1$ if b^* is random.

One problem is that we do not know how to directly prove the IND-CPA, or even OW-CPA security of NEV-PKE′ under the RLWE assumption. For this, we introduce a variant of the RLWE problem, called subset-sum parity RLWE problem (sspRLWE), which basically says that it is hard to compute $\bar{v}e \bmod 2$ given an RLWE tuple $(h, hr + e)$ as input. We note that our sspRLWE can also be seen as a generalization of the RLWE2 problem in [18], which essentially asks to compute $\bar{v}e \bmod 2$ for $\bar{v} = 1$ (or equivalently $k = 1$). At first glance, one might think that sspRLWE is hard if its corresponding RLWE is hard. Unfortunately, even in the special RLWE2 setting, the authors [18] only provide heuristic arguments for its equivalence to RLWE.

In Sect. 4.3, we show that the sspRLWE problem with discrete Gaussian noise distribution is polynomially equivalent to the DRLWE problem (with different Gaussian parameters), which can be extended to the binomial distribution by a standard argument using Rényi divergence [5]. Our proof is based on a very simple observation: $\bar{v}(2e_1 + e_0) = \bar{v}e_0 \bmod 2$, and one can naturally convert a DRLWE instance $(h, b = hr + e_1)$ to an sspRLWE instance $(h' = 2h, b' = 2b + e_0)$ (note that when both e_1 and e_0 follow discrete Gaussian distributions, so does $2e_1 + e_0$ [39]). Then, if (h, b) is computationally indistinguishable from uniform, the adversary can obtain no information about $\bar{v}e_0 \bmod 2$ from (h', b'). Since this proof also applies to $\bar{v} = 1$, we partially solve the problem of connecting RLWE2 to RLWE (for sufficiently large parameters). We also provide two concrete theorems for basing sspRLWE with $k = 1$ (namely, RLWE2) and $k = 2$ on the RLWE problem with binomial noise distribution B_1 and uniform binary noise distribution, respectively. The two proofs are mainly based the fact that $e \bmod 2 = 0 \Leftrightarrow e = 0$ for any variable $e \in \{-1, 0, 1\}$. Note that our parameter set NEV′-512 exactly corresponds to the case of $k = 2$. We believe that those proofs

provide a good confidence to make the reasonable assumption: the concrete hardness of sspRLWE is equal to RLWE with the same parameters. For those who is unsatisfying with this assumption, we recommend to use NEV-PKE, which is provably IND-CPA secure under the standard NTRU and RLWE assumptions, and only has slightly worse decryption failure and performance.

Table 1. Comparison between our NEV-KEMs, NTRU and Kyber in sizes

Schemes	$\|pk\|$ (Bytes)	$\|sk\|$ (Bytes)	$\|C\|$ (Bytes)	Dec Failure	LWE Estimator	NIST Security	Improv. Ratio
Kyber-512	800	1632	768	2^{-178}	140	Level 1	21.56%
NTRU-HPS2048677	930	1234	930	–	170		33.87%
NTRU-HRSS701	1138	1450	1138	–	158		45.96%
Our NEV-512	615	1294	615	2^{-138}	141		–
Our NEV′-512	615	1294	615	2^{-200}	145		–
Kyber-768	1184	2400	1088	2^{-164}	201	Level 3	$-8.19\%^{\dagger}$
NTRU-HPS4096821	1230	1590	1230	–	199		$0.08\%^{\dagger}$
Kyber-1024	1568	3168	1568	2^{-174}	270	Level 5	21.62%
NTRU-HPS40961229	1842	2366	1842	–	296		33.28%
NTRU-HRSS1373	2401	2983	2401	–	300		48.81%
Our NEV-1024	1229	2522	1229	2^{-152}	281		–
Our NEV′-1024	1229	2522	1229	2^{-200}	292		–

1.3 Comparison to the State of the Art

We give a detailed comparison between our KEMs, NTRU and Kyber in Tables 1 and 2. The column "LWE estimator" in Table 1 presents the concrete security estimates obtained by using the LWE estimator script [1]. The columns "Improv. Ratio" in Table 1 and "Speedup" in Table 2 are obtained by dividing the total sizes/timings of the corresponding schemes in an ephemeral key exchange by that of our NEV-KEM (i.e., NEV-512 and NEV-1024) at the same security levels, except that we obtain the figures (marked with †) for Kyber768 and NTRU-HPS4096821 at NIST level 3 security by dividing that of our KEMs at NIST level 5 security (i.e., NEV-1024). One can see that our NEV-KEM using NEV-1024 has the same public key and ciphertext sizes as that of NTRU-HPS4096821, but is still 4.10–11.05X faster: because our ring allows (partial) NTT. Compared to Kyber768, our NEV-KEM using NEV-1024 has size 8.19% larger but is 1.2X faster: because we do not have to expand a seed to a random matrix.

In Table 3, we compare our KEMs with three recent NTRU variants in sizes, where the figures in the column "LWE estimator" for schemes based on RLWE2, RLWR and sspRLWE problems are all obtained by using the assumption that the concrete hardness of those problems are equal to their corresponding RLWE problems with the same parameters. In Sect. 7, we will also compare the concrete

Table 2. Comparison between our NEV-KEMs, NTRU and Kyber in efficiency

Schemes	KeyGen (Ref)	Encap (Ref)	Decap (Ref)	KeyGen (AVX2)	Encap (AVX2)	Decap (AVX2)	Speedup (Ref/AVX2)
Kyber-512	132 334	167 834	195 024	32 996	47 514	34 816	1.67/1.42X
NTRU-HPS2048677	4 957 166	220 554	293 126	320 234	82 991	62 907	18.46/5.74X
NTRU-HRSS701	5 469 959	125 559	309 743	287 524	54 270	66 801	19.92/5.03X
Our NEV-512	95 007	88 131	113 268	21 192	33 694	26 297	–
Our NEV′-512	89 154	83 978	110 463	20 620	30 787	23 841	–
Kyber-768	217 023	263 971	303 945	54 789	72 268	53 822	1.21/1.19X†
NTRU-HPS4096821	6 645 818	251 935	280 318	450 336	96 475	78 522	11.05/4.10X†
Kyber-1024	329 555	377 541	421 837	73 562	97 756	76 454	1.74/1.62X
NTRU-HPS40961229	14 944 617	484 755	654 931	–	–	–	24.76/-X
NTRU-HRSS1373	18 366 972	313 188	769 187	–	–	–	29.94/-X
Our NEV-1024	208 045	183 977	257 489	37 636	64 046	50 807	–
Our NEV′-1024	205 719	171 669	251 303	37 805	60 411	45 851	–

Table 3. Comparison between our NEV-KEMs and recent NTRU variants in Size

Schemes	$\|pk\|$ (Bytes)	$\|C\|$ (Bytes)	Dec Failure	Hardness Assumption	LWE Estimator
NTRU-A$^{576}_{2593}$ [18]	864	864	2^{-150}	NTRU + RLWE2	154
NTRU-A$^{648}_{2917}$ [18]	972	972	2^{-170}	$R_q = \mathbb{Z}_q[x]/(x^n - x^{n/2} + 1)$	171
NTRU-A$^{768}_{3457}$ [18]	1152	1152	2^{-202}		200
NTRU-A$^{864}_{3457}$ [18]	1296	1296	2^{-182}		225
NTRU-A$^{972}_{3889}$ [18]	1458	1458	2^{-206}		252
NTRU-A$^{1152}_{3457}$ [18]	1728	1728	2^{-140}		305
NTRU-A$^{1296}_{3889}$ [18]	1944	1944	2^{-158}		341
NTTRU-768 [33]	1248	1248	2^{-1217}	NTRU + RLWE $R_q = \mathbb{Z}_q[x]/(x^n - x^{n/2} + 1)$	170
BAT-512 [20]	521	473	2^{-146}	NTRU + RLWR	144
BAT-1024 [20]	1230	1006	2^{-166}	$R_q = \mathbb{Z}_q[x]/(x^n + 1)$	273
Our NEV-512	615	615	2^{-138}	NTRU + RLWE	141
Our NEV-1024	1229	1229	2^{-152}	$R_q = \mathbb{Z}_q[x]/(x^n + 1)$	281
Our NEV′-512	615	615	2^{-200}	NTRU + sspRLWE	145
Our NEV′-1024	1229	1229	2^{-200}	$R_q = \mathbb{Z}_q[x]/(x^n + 1)$	292

performance of our schemes with BAT in Table 5 and NTTRU in Table 6 (we do not have the source code of NTRU-A, but it was reported having comparable performance with NTTRU [18, Table 3]). In summary, our KEMs have comparable efficiency as NTRU-A, but have sizes at least 28% more compact. The sizes of BAT are 19.19% (resp., 9.03%) smaller than our Π_{KEM} at NIST level 1 (resp., 5) security (note that BAT uses a strong RLWR with binary secret assumption,

which allows to compress the ciphertexts almost for free), but our NEV-KEM is 140-973X (resp., 334-2648X) faster than BAT.

Most recently, Micciancio and Schultz [34] provide a framework to capture the encoding of the message and the compression/quantization of the ciphertext, which aims at improving the ratio of the size of a plaintext to the size of a LWE-based ciphertext. As a NTRU-like ciphertext only contains a single ring element which will be multiplied by the secret key (namely, f) in decryption, one cannot directly apply their framework to improve the encryption rate of our schemes.

2 Preliminaries

2.1 Notation

Let n be a power of 2, and q a prime. We denote by R the ring $R = \mathbb{Z}[X]/(X^n+1)$ and by R_q the ring $R_q = \mathbb{Z}_q[X]/(X^n+1)$. The regular font letters (e.g., a, b) represent elements in R or R_q (including elements in $\mathbb{Z}$ or $\mathbb{Z}_q$), and bold lowercase letters (e.g., $\mathbf{a}$, $\mathbf{b}$) denote vectors of R or $\mathbb{Z}$ elements. For a positive integer $\ell \in \mathbb{Z}$, by $[\ell]$ we denote the set $\{0, \ldots, \ell-1\}$. By $r' = r \bmod^{\pm} q$ we denote the unique element in the range $[-\frac{q-1}{2}, \frac{q-1}{2}]$ such that $r' = r \bmod q$. For an element $w \in \mathbb{Z}_q$, we write $\|w\|_\infty$ to mean $|w \bmod^{\pm} q|$. The ℓ_∞ and ℓ_2 norms of a ring element $w \in R_q$ is defined as that of its coefficient vector $\mathbf{w} \in \mathbb{Z}_q^n$.

By $x \leftarrow \mathcal{D}$ we denote sampling x according to a distribution $\mathcal{D}$ and by $\mathcal{U}(S)$ we denote the uniform distribution over a finite set S. When we write that sampling a polynomial $g \leftarrow \mathcal{D}$ from a distribution $\mathcal{D}$ over $\mathbb{Z}$, we mean that sampling each coefficient of g from $\mathcal{D}$ individually. We use $\log_b$ to denote the logarithm function in base b (e.g., 2 or natural constant e) and log to represent $\log_e$. We say that a function $f : \mathbb{N} \to [0, 1]$ is *negligible*, if for every positive c and all sufficiently large κ it holds that $f(\kappa) < 1/\kappa^c$. We denote by $\mathsf{negl} : \mathbb{N} \to [0, 1]$ an (unspecified) negligible function.

Binomial Distribution. The centered binomial distribution B_η with some positive $\eta \in \mathbb{Z}$ is defined as follows:

$$B_\eta = \left\{ \sum_{i=0}^{\eta-1} (a_i - b_i) : (a_0, \ldots, a_{\eta-1}, b_0, \ldots, b_{\eta-1}) \leftarrow \{0,1\}^{2\eta} \right\}$$

Ternary Distribution. The ternary distribution $\mathcal{T}_\sigma$ with some positive real $\sigma \in (0, 1/2)$ denotes the distribution of sampling a variable $x \in \{-1, 0, 1\}$ with $\Pr[x = 1] = \Pr[x = -1] = \sigma$, and $\Pr[x = 0] = 1 - 2\sigma$. By this notation, we have $\mathcal{T}_{1/3} = \mathcal{U}(\{-1, 0, 1\})$ is the uniform ternary distribution, and $\mathcal{T}_{1/4} = B_1$ is the centered binomial distribution with $\eta = 1$.

Gaussian Distribution. The Gaussian function $\rho_{s,\mathbf{c}}(\mathbf{x})$ over $\mathbb{R}^m$ centered at $\mathbf{c} \in \mathbb{R}^m$ with parameter $s > 0$ is defined as $\rho_{s,\mathbf{c}}(\mathbf{x}) = \exp(-\pi\|\mathbf{x} - \mathbf{c}\|^2/s^2)$. For lattice $\mathbf{\Lambda} \subseteq \mathbb{R}^m$, let $\rho_{s,\mathbf{c}}(\mathbf{\Lambda}) = \sum_{\mathbf{x}\in\mathbf{\Lambda}} \rho_{s,\mathbf{c}}(\mathbf{x})$, and define the discrete Gaussian distribution over $\mathbf{\Lambda}$ as $D_{\mathbf{\Lambda},s,\mathbf{c}}(\mathbf{y}) = \frac{\rho_{s,\mathbf{c}}(\mathbf{y})}{\rho_{s,\mathbf{c}}(\mathbf{\Lambda})}$, where $\mathbf{y} \in \mathbf{\Lambda}$. We omit the subscript $\mathbf{c}$ in the above notations if $\mathbf{c} = \mathbf{0}$.

Lemma 1 ([7,30]). *For any real $s, t > 0$, $c \geq 1, C = c \cdot \exp(\frac{1-c^2}{2}) < 1$, integer $m > 0$, and any $\mathbf{y} \in \mathbb{R}^m$ we have that $\Pr_{\mathbf{x} \leftarrow D_{\mathbb{Z}^m, s}}[\|\mathbf{x}\|_\infty > t \cdot s] \leq 2e^{-\pi t^2}$.*

Lemma 2 (Special case of [39, Theorem 3.1]). *Let $\alpha, \beta, \gamma > 0$ be reals such that $\alpha \geq \omega(\sqrt{\log n}), \gamma = \sqrt{\alpha^2 + \beta^2}$ and $\alpha\beta/\gamma > 2 \cdot \omega(\sqrt{\log n})$. Consider the following probabilistic experiment:*

$$\text{Choose } \mathbf{x}_2 \leftarrow D_{2\mathbb{Z}^n, \beta}, \text{ then choose } \mathbf{x}_1 \leftarrow \mathbf{x}_2 + D_{\mathbb{Z}^n, \alpha}.$$

Then, the marginal distribution of $\mathbf{x}_1$ is statistically close to $D_{\mathbb{Z}^n, \gamma}$.

2.2 Public-Key Encryption

A public-key encryption (PKE) Π_{PKE} with plaintext space $\mathcal{M}$ consists of three PPT algorithms $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$:

- $\mathsf{KeyGen}(1^\kappa)$: given a security parameter κ as input, output a pair of public and secret keys (pk, sk), denoted as $(pk, sk) = \mathsf{KeyGen}(1^\kappa)$.
- $\mathsf{Enc}(pk, M; r)$: given the public key pk, a plaintext $M \in \mathcal{M}$ and a randomness r (which might be an empty string) as inputs, output a ciphertext C, denoted as $C = \mathsf{Enc}(pk, M; r)$ or $C = \mathsf{Enc}(pk, M)$ in brief.
- $\mathsf{Dec}(sk, C)$: given the secret key sk and a ciphertext C as inputs, output a plaintext M' (which might be $\perp$), denoted as $M' = \mathsf{Dec}(sk, C)$.

We say that a PKE scheme $\Pi_{\mathrm{PKE}} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ is δ-correct, if for any $M \in \mathcal{M}$, $(pk, sk) = \mathsf{KeyGen}(1^\kappa)$ and $C = \mathsf{Enc}(pk, M)$, the probability that $\mathsf{Dec}(sk, C) \neq M$ is at most δ over the random coins used in KeyGen and Enc. For our interest, we recall the OW-CPA and IND-CPA security for PKEs from [8], which is modeled by games between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ in Fig. 1.

Definition 1 (OW-CPA PKE). *We say that a PKE scheme Π_{PKE} is* OW-CPA secure *if for any PPT adversary $\mathcal{A}$, its advantage*

$$\mathrm{Adv}^{\text{ow-cpa}}_{\Pi_{\mathrm{PKE}}, \mathcal{A}}(\kappa) = \Pr[M' = M^*]$$

in the OW-CPA security game in Fig. 1 is negligible in security parameter κ.

Definition 2 (IND-CPA PKE). *We say that a PKE scheme Π_{PKE} is* IND-CPA secure *if for any PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, its advantage*

$$\mathrm{Adv}^{\text{ind-cpa}}_{\Pi_{\mathrm{PKE}}, \mathcal{A}}(\kappa) = \left| \Pr[\mu' = \mu^*] - \frac{1}{2} \right|$$

in the IND-CPA security game in Fig. 1 is negligible in security parameter κ.

Algorithm OW-CPA:

$(pk, sk) = \Pi_{\text{PKE}}.\mathsf{KeyGen}(1^\kappa)$;
$M^* \leftarrow \mathcal{M}$;
$C^* = \Pi_{\text{PKE}}.\mathsf{Enc}(pk, M^*)$;
$M' = \mathcal{A}(pk, C^*)$;
return $M' = M^*$;

Algorithm IND-CPA:

$(pk, sk) = \Pi_{\text{PKE}}.\mathsf{KeyGen}(1^\kappa)$;
$(M_0, M_1, st) = \mathcal{A}_1(pk)$;
$\mu \leftarrow \{0, 1\}$;
$C^* = \Pi_{\text{PKE}}.\mathsf{Enc}(pk, M_\mu)$;
$\mu' = \mathcal{A}_2(C^*, st)$;
return $\mu' = \mu^*$;

Fig. 1. Games for OW-CPA and IND-CPA Security of PKEs

2.3 Key Encapsulation Mechanism

A key encapsulation mechanism (KEM) Π_{KEM} with session key space $\mathcal{K}$ consists of three PPT algorithms $(\mathsf{KeyGen}, \mathsf{Encap}, \mathsf{Decap})$:

- $\mathsf{KeyGen}(1^\kappa)$: given a security parameter κ as input, output a pair of public and secret keys (pk, sk), denoted as $(pk, sk) = \mathsf{KeyGen}(1^\kappa)$.
- $\mathsf{Encap}(pk; r)$: given the public key pk, and a randomness r as inputs, output a ciphertext C and a session key $K \in \mathcal{K}$, denoted as $(C, K) = \mathsf{Encap}(pk; r)$, or $(C, K) = \mathsf{Encap}(pk)$ in brief.
- $\mathsf{Decap}(sk, C)$: given a secret key sk and a ciphertext C as inputs, output a key K' (which might be a failure symbol $\perp$), denoted as $K' = \mathsf{Decap}(sk, C)$.

We say that a KEM scheme $\Pi_{\text{KEM}} = (\mathsf{KeyGen}, \mathsf{Encap}, \mathsf{Decap})$ is δ-correct, if for any $(pk, sk) = \mathsf{KeyGen}(1^\kappa)$ and $(C, K) = \mathsf{Encap}(pk)$, the probability that $\mathsf{Decap}(sk, C) \neq K$ is at most δ over the random coins used in KeyGen and Enc. We now recall the chosen-ciphertext security for KEMs from [12], which is modeled by the game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ in Fig. 2.

Algorithm IND-CCA:

$(pk, sk) = \Pi_{\text{KEM}}.\mathsf{KeyGen}(1^\kappa)$;
$\mu \leftarrow \{0, 1\}$;
$(C^*, K_0^*) = \Pi_{\text{KEM}}.\mathsf{Encap}(pk)$;
$K_1^* \leftarrow \mathcal{K}$;
$\mu' = \mathcal{A}^{\mathcal{O}_{\mathsf{Dec}}(\cdot)}(pk, C^*, K_\mu^*)$;
return $\mu' = \mu^*$;

Oracle $\mathcal{O}_{\mathsf{Dec}}(C)$**:**

if $C = C^*$ **then**
    return $\perp$;
end
$K = \mathsf{Decap}(sk, C)$;
return K;

Fig. 2. Game for IND-CCA Security of KEMs

Definition 3 (IND-CCA KEM). *We say that a KEM scheme* Π_{KEM} *is* IND-CCA secure *if for any PPT adversary* $\mathcal{A}$*, its advantage*

$$\text{Adv}^{\text{ind-cca}}_{\Pi_{\text{KEM}}, \mathcal{A}}(\kappa) = \left| \Pr[\mu' = \mu^*] - \frac{1}{2} \right|$$

in the IND-CCA security game in Fig. 2 is negligible in security parameter κ.

2.4 Hard Problems

Let n be a power of 2, and q a prime. Let $R_q = \mathbb{Z}_q[x]/(x^n+1)$. Let R_q^* denote all invertible ring elements in R_q. Let $\chi_f, \chi_g, \chi_r, \chi_e$ be four probability distributions over R. Let $v \in R_q^*$ be a publicly known small ring element.

The NTRU Assumption. The computational NTRU problem $\mathrm{NTRU}_{n,q,\chi_f,\chi_g,v}$ asks an algorithm, given $h = g/f \in R_q$ as input, to output f', where $f' \leftarrow \chi_f, g \leftarrow \chi_g$ and $f = vf' + 1 \in R_q^*$. The decisional NTRU problem $\mathrm{DNTRU}_{n,q,\chi_f,\chi_g,v}$ asks an algorithm to distinguish the following two distributions:

$$\{h = g/f \mid f' \leftarrow \chi_f, g \leftarrow \chi_g, \text{ and } f = vf' + 1 \in R_q^*\} \text{ and } \{u \mid u \leftarrow R_q\}.$$

The computational (resp., decisional) NTRU assumption says that it is hard for any PPT algorithms to solve $\mathrm{NTRU}_{n,q,\chi_f,\chi_g,v}$ (resp., $\mathrm{DNTRU}_{n,q,\chi_f,\chi_g,v}$) with non-negligible advantage over a random guess.

Remark 1. The above definition generalizes the common NTRU assumption with $v = p \in R_q^*$ for some integer p (e.g., $p = 3$ in [11,24,25,43]) with a publicly known ring element $v \in R_q^*$. We note that this generalization is mild up to the choices of the secret key distribution χ_f, because $\mathrm{NTRU}_{n,q,\chi_f,\chi_g,v}$ is essentially equivalent to the standard NTRU problem $\mathrm{NTRU}_{n,q,\chi'_f,\chi_g,p}$ with $\chi'_f = p^{-1}v\chi_f$ (or $\chi_f = pv^{-1}\chi'_f$). In fact, the polynomial $v = x + 2$ was recommended by the authors of the original NTRU cryptosystem as early as 2000 [25], and was investigated in [6,22,23,27,35,43].

Since its introduction [24], the NTRU problem has been studied more than 25 years, and there is no significant algorithmic progress. The decisional NTRU (DNTRU) assumption over the cyclotomic ring $R = \mathbb{Z}_q[x]/(x^n+1)$, which is also known as the decisional small polynomial ratio (DSPR) assumption, has been extensively used and investigated in [10,16,19,31,40,43]. Notably, Stehlé and Steinfeld [43] showed that the DNTRU assumption indeed holds unconditionally if χ_f, χ_g are discrete Gaussian distributions of standard deviation $\sigma = \omega(n\sqrt{q})$ (We note that their proof mainly focuses on the special case $v = 3$, but it essentially applies to any invertible $v \in R_q^*$). For small secret distributions, a variant of the NTRU problem over $R_q = \mathbb{Z}_q[x]/(x^n + 1)$ is also shown to be at least as hard as the worst-case approximate SVP problem on ideal lattices [40].

The RLWE Assumption. The computational RLWE problem $\mathrm{RLWE}_{n,q,\chi_r,\chi_e}$ asks an algorithm, given a polynomial number of samples from the distribution $\{(a, b = ar + e) \mid a \leftarrow R_q, e \leftarrow \chi_e\}$ as inputs, to output the secret $r \in R_q$, where $r \leftarrow \chi_r$. The decisional RLWE problem $\mathrm{DRLWE}_{n,q,\chi_r,\chi_e}$ asks an algorithm, given a polynomial number of samples to distinguish the following two distributions:

$$\{(a, b = ar + e) \mid a \leftarrow R_q, e \leftarrow \chi_e\} \text{ and } \{(a, u) \mid a \leftarrow R_q, u \leftarrow R_q\}.$$

The computational (resp., decisional) RLWE assumption says that it is hard for any PPT algorithms to solve $\mathrm{RLWE}_{n,q,\chi_r,\chi_e}$ (resp., $\mathrm{DRLWE}_{n,q,\chi_r,\chi_e}$) with non-negligible advantage over a random guess.

As an extension of the LWE problem [42], the RLWE problem was first considered in [32,44], and was provably as hard as some hard lattice problems such as the Shortest Vectors Problem (SVP) on ideal lattices.

The Subset-Sum Parity RLWE Assumption. We introduce a variant of the RLWE problem which we call subset-sum parity RLWE (sspRLWE) problem. Formally, the sspRLWE problem $\text{sspRLWE}_{n,q,\chi_r,\chi_e,v}$ asks an algorithm, given an RLWE instance $(a, b = ar + e) \in R_q$ as input, to output $ve \bmod 2 \in R_2$ for some fixed ring element $v \in R_2$. This name comes from the fact that for $R = \mathbb{Z}[X]/(x^n + 1)$, the i-th coefficient of $ve \bmod 2 \in R_2$ is essentially equal to the parity of the subset sum $\sum_{v_j=1} e_{(i-j) \bmod n}$ of the coefficient vector $\mathbf{e} = (e_0, \ldots, e_{n-1})$ of $e \in R_q$. The sspRLWE assumption says that it is hard for any PPT algorithms to solve $\text{sspRLWE}_{n,q,\chi_r,\chi_e,v}$ with non-negligible advantage over a random guess according to the distribution $\chi' = v\chi_e \bmod 2$.

Remark 2. Our sspRLWE problem can be seen as a generalization of the RLWE2 problem [18] from a special choice of $v = 1$ to a general chosen $v \in R_2$. On the first hand, the $\text{sspRLWE}_{n,q,\chi_r,\chi_e,v}$ problem is not harder than the corresponding RLWE problem $\text{RLWE}_{n,q,\chi_r,\chi_e}$. On the other hand, if the DRLWE problem $\text{DRLWE}_{n,q,\chi_r,\chi_e}$ is hard, it seems that a RLWE sample $(a, b = ar+e)$ essentially hides all the information about e, and that the best way for a PPT algorithm to solve the sspRLWE problem is to make a random guess on $ve \bmod 2$ according to the distribution $\chi' = v\chi_e \bmod 2$. Moreover, the problem of reducing $\text{DRLWE}_{n,q,\chi_r,\chi_e}$ to $\text{sspRLWE}_{n,q,\chi_r,\chi_e,v}$ can be seen as the problem of solving $\text{DRLWE}_{n,q,\chi_r,\chi_e}$ with modular hints $ve \bmod 2$, and an efficient algorithm to solve $\text{sspRLWE}_{n,q,\chi_r,\chi_e,v}$ may directly lead to a new and better algorithm to solve $\text{RLWE}_{n,q,\chi_r,\chi_e}$ according to the study in [13].

However, we cannot expect a general reduction that bases the hardness of $\text{sspRLWE}_{n,q,\chi_r,\chi_e,v}$ on that of $\text{DRLWE}_{n,q,\chi_r,\chi_e}$ for arbitrary choices of v and noise distribution χ_e, because $ve \bmod 2$ may loose too much information about e and may be of little help to solve $\text{DRLWE}_{n,q,\chi_r,\chi_e}$. Note that the authors [18] only present heuristic arguments for the equivalence of RLWE and $\text{sspRLWE}_{n,q,\chi_r,\chi_e,v}$ even for the special case $v = 1$. Moreover, it is easy to show that $\text{DRLWE}_{n,q,\chi_r,\chi'_e}$ for $\chi'_e = 2\chi_e$ is equivalent to $\text{DRLWE}_{n,q,\chi_r,\chi_e}$, but we always have $v\chi'_e = 0 \bmod 2$ for the $\text{sspRLWE}_{n,q,\chi_r,\chi'_e,v}$ problem. For our purpose, we are particularly interested in the sspRLWE problem $\text{sspRLWE}_{n,q,\chi_r,\chi_e,\bar{v}}$ satisfying the following two conditions:

- $\bar{v} = 1 + x^{n/k} + x^{2n/k} + \cdots + x^{(k-1)n/k} \in R_2$ for integers $n/k = 256$;
- χ_e is the binomial distribution.

Looking ahead, we will use this kind of sspRLWE assumption to construct a OW-CPA secure encryption $\mathsf{NEV\text{-}PKE}'$ with better noise tolerance in Sect. 4.2, and will show that for appropriate choices of parameters, the sspRLWE problem is at least as hard as the standard RLWE problem (with slightly different parameters) in Sect. 4.3 (and thus partially solves the problem of reducing the RLWE2 problem to the standard RLWE problem in [18]).

3 NTRU Encryption Using Vector Decoding

In this section, we first give a provably secure IND-CPA PKE scheme called NEV-PKE from the standard DNTRU and DRLWE assumptions, then we transform it into a IND-CCA KEM called NEV-KEM using the generic Fujisaki-Okamoto (FO) transformation [21]. We begin by describing our plaintext encoding and decoding algorithms.

3.1 Plaintext Encoding and Decoding

Our way of encoding and decoding plaintext is inspired by the method for RLWE-based encryption in [41], which essentially encodes a single plaintext bit into multiple coefficients of a ring element, and is also used in Newhope [2,4] submitted to NIST PQC competition. We adapted this idea to the NTRU setting. Formally, let n be a power of 2, and q be a prime. Let $R = \mathbb{Z}[x]/(x^n+1)$ and $R_q = \mathbb{Z}_q[x]/(x^n+1)$. Let $\mathcal{M} = \{0,1\}^\ell$ be the plaintext space. Let k be the largest integer satisfying $k|n$ and $n/k \geq \ell$. Let $v = (1 - x^{n/k}) \in R_q^*$ be a ring element, whose inverse is $v^{-1} = \frac{q+1}{2}(1 + x^{n/k} + \cdots + x^{(k-1)n/k}) \in R_q^*$. We define the following two algorithms Pt2poly and Poly2Pt for encoding and decoding:

- $\mathsf{Pt2poly}(M)$: given a plaintext $M \in \{0,1\}^\ell$ as input, return a polynomial $m = M_0 + M_1 x + \cdots + M_{\ell-1}x^{\ell-1} \in R_q$, where $M_i \in \{0,1\}$ is the i-th bit of M, denoted as $m = \mathsf{Pt2poly}(M)$.
- $\mathsf{Poly2Pt}(w)$: given a polynomial $w = w_0 + w_1 x + \cdots + w_{n-1}x^{n-1} \in R_q$ as input, first compute $\tilde{w}_i = w_i - \frac{q+1}{2} \bmod^{\pm} q$ for all $i \in [n]$. Then, compute $t_j = \sum_{i=j \bmod n/k} |\tilde{w}_i|$ for all $j \in [\ell]$. Finally, set

$$M_j = \begin{cases} 1, & \text{if } t_j < \frac{k\cdot(q-1)}{4}; \\ 0, & \text{otherwise,} \end{cases}$$

  and return the plaintext $M = (M_0, \ldots, M_{\ell-1}) \in \{0,1\}^\ell$.

We have the following lemma for the above two algorithms.

Lemma 3. *Let $n, q, k, \ell \in \mathbb{Z}$ and $v \in R_q^*$ be defined as above. Then, for any $M \in \{0,1\}^\ell$, $m = \mathsf{Pt2poly}(M) \in R_q$ and any polynomial $e = e_0 + e_1 x + \cdots + e_{n-1}x^{n-1} \in R_q$ satisfying the following condition*

$$\left(\sum_{i=j \bmod n/k} \left| e_i \bmod^{\pm} q \right| \right) < \frac{k\cdot(q-1)}{4} \text{ for } i \in [n] \text{ and } j \in [\ell] \qquad (1)$$

we always have $\mathsf{Poly2Pt}(v^{-1}m + e) = M$.

Proof. Let $m = \mathsf{Pt2poly}(M) \in R_q$. By the definition of $\mathsf{Pt2poly}(M)$, we have that m only has non-zero binary coefficients at the first $\ell \leq n/k$ coordinates. Thus, multiplying m with $v^{-1} = \frac{q+1}{2}(1 + x^{n/k} + \cdots + x^{(k-1)n/k})$ is essentially equal to first multiply m by $\frac{q+1}{2}$ and then copy $k-1$ times the first n/k coefficients as a block to the next $(k-1)n/k$ coordinates. In other words, for all $u = v^{-1}m \in R_q$, we always have $u_i = M_j \frac{q+1}{2}$ for all $i = j \bmod n/k$ for $i \in [n]$ and $j \in [\ell]$, where $u = u_0 + u_1 x + \cdots + u_{n-1}x^{n-1}$ and $M = (M_0, \ldots, M_{\ell-1})$. Let $w = u + e = v^{-1}m + e \in R_q$, it suffices to show that $\mathsf{Poly2Pt}(w)$ will always correctly recover each bit of M. Formally, let $w = w_0 + w_1 x + \cdots + w_{n-1}x^{n-1}$, we continue the proof by considering the value of each $M_j \in \{0, 1\}$ for $j \in [\ell]$:

- $M_j = 1$: we have that $w_i = u_i + e_i = \frac{q+1}{2} + e_i$ for all $i = j \bmod n/k$, and that $\tilde{w}_i = w_i - \frac{q+1}{2} = e_i \bmod^{\pm} q$. This means that

$$t_j = \sum_{i=j \bmod n/k} |\tilde{w}_i| = \sum_{i=j \bmod n/k} |e_i \bmod^{\pm} q| < \frac{k \cdot (q-1)}{4},$$

  and that $\mathsf{Poly2Pt}(w)$ will output $M_j = 1$;
- $M_j = 0$: we have that $w_i = e_i$ for all $i = j \bmod n/k$, and that $\tilde{w}_i = w_i - \frac{q+1}{2} = e_i - \frac{q+1}{2} \bmod^{\pm} q$. Since we have either $e_i = |e_i \bmod^{\pm} q|$ or $e_i = q - |e_i \bmod^{\pm} q|$, it is easy to check that $|\tilde{w}_i| \geq \frac{q-1}{2} - |e_i \bmod^{\pm} q|$. This means that

$$t_j = \sum_{i=j \bmod n/k} |\tilde{w}_i| \geq \sum_{i=j \bmod n/k} \left(\frac{q-1}{2} - |e_i \bmod^{\pm} q| \right) > \frac{k \cdot (q-1)}{4},$$

  and that $\mathsf{Poly2Pt}(w)$ will output $M_j = 0$.

This completes the proof.

Remark 3. There is a tradeoff between the plaintext length ℓ and the decoding capacity. A smaller k (e.g., $k = 1$) allows to support longer plaintext length (as we require $\ell \leq n/k$) but has worse noise tolerance. In particular, if each coefficient of e is chosen from a distribution such that the probability of $|e_i \bmod^{\pm} q| < \frac{q-1}{4}$ for all $i \in [n]$ is $1 - p$, then the probability that $\mathsf{Poly2Pt}(v^{-1}m + e) = M$ is roughly about $1 - p^k$. This is why we prefer to choose the largest integer k such that $n/k \geq \ell$. For the typical application of PKE in encrypting a session key $\ell = 128$ or 256, one could fix $k = n/\ell$ to obtain the best noise tolerance.

3.2 A Provably Secure IND-CPA NTRU Encryption

Let $n, q, k, \ell \in \mathbb{Z}$ and $v \in R_q^*$ be defined as above. Let $\chi_f, \chi_g, \chi_r, \chi_e$ be four probability distributions over R. Our PKE scheme $\mathsf{NEV\text{-}PKE}$ consists of the following three algorithms $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$:

- $\mathsf{NEV\text{-}PKE.KeyGen}(\kappa)$: given the security parameter κ as input, randomly choose $f' \leftarrow \chi_f$ and $g \leftarrow \chi_g$ such that $f = vf' + 1 \in R_q^*$ is invertible. Then, return the public and secret key pair $(pk, sk) = (h = g/f, f) \in R_q \times R_q$.

- $\mathsf{NEV\text{-}PKE.Enc}(pk, M)$: given the public key $pk = h \in R_q$ and a plaintext $M \in \{0,1\}^\ell$ as inputs, randomly choose $r \leftarrow \chi_r, e \leftarrow \chi_e$, compute $m = \mathsf{Pt2poly}(M) \in R_q$ and $c = hr + e + v^{-1}m$. Return the ciphertext $c \in R_q$.
- $\mathsf{NEV\text{-}PKE.Dec}(sk, c)$: given the secret key $sk = f = vf' + 1 \in R_q^*$ and a ciphertext $c \in R_q$ as inputs, compute $w = fc$, and $M' = \mathsf{Poly2Pt}(w)$. Finally, return the message $M' \in \{0,1\}^\ell$.

Remark 4. Our above PKE scheme can be easily adapted to support other choices of $v \in R_q^*$, e.g., $v = 3$, but it seems that $v = (1 - x^{n/k})$ might be the optimal one in reducing the decryption failure (see below).

Since we have the following decryption formula

$$w = fc = gr + (vf' + 1)(e + v^{-1}m) = \underbrace{gr + vf'e + f'm + e}_{=\tilde{e}} + v^{-1}m = \tilde{e} + v^{-1}m.$$

the decryption is correct as long as we set the parameters such that $\tilde{e}$ satisfies the condition (1) in Lemma 3. It is worth to note the following three nice properties about our decryption formula, which are very important for our scheme to choose practical (and small) parameters:

1. Multiplying $v = (1 - x^{n/k})$ will only increase the size of $vf'e$ from that of $f'e$ in a very mild way when taking account of the distributions of f' and e: the standard deviation of $vf'e$ is about $\sqrt{2}$ times larger than that of $f'e$;
2. The size of $f'm$ is far smaller than that of gr and $vf'e$ because m only has non-zero binary coefficients at the first $\ell \leq n/k$ coefficients.
3. The contribution of g, r to the size of $\tilde{e}$ is much less than that of (f', e), and we can utilize this asymmetric property to obtain a better balance between security and decryption failure as in [45].

In Sect. 5.2, we will choose concrete parameters such that the decryption failure is negligibly small. For security, we have the following theorem.

Theorem 1. *Let $n, q \in \mathbb{Z}$, $v \in R_q^*$ and distributions $\chi_f, \chi_g, \chi_r, \chi_e$ be defined as above. Then, under the* $\mathrm{DNTRU}_{n,q,\chi_f,\chi_g,v}$ *and* $\mathrm{DRLWE}_{n,q,\chi_r,\chi_e}$ *assumption, our PKE scheme* NEV-PKE *is provably IND-CPA secure in the standard model.*

Proof. We prove Theorem 1 by using a sequence of games $G_0 \sim G_2$, where G_0 is the standard IND-CPA game, and G_2 is a random one. The security is established by showing that G_0 and G_2 are computationally indistinguishable in the adversary's view. Let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be an adversary which can break the IND-CPA security of our PKE with advantage ϵ. Let F_i be the event that $\mathcal{A}$ correctly guesses $\mu' = \mu^*$ in game $i \in \{0, \ldots, 2\}$. By definition, the adversary's advantage $\mathrm{Adv}^{\text{ind-cpa}}_{\mathsf{NEV\text{-}PKE},\mathcal{A}}(\kappa)$ in game i is exactly $|\Pr[F_i] - 1/2|$.

Game G_0. This game is the real IND-CPA security game defined in Fig. 1. Formally, the challenger $\mathcal{C}$ works as follows:

KeyGen. randomly choose $f' \leftarrow \chi_f$ and $g \leftarrow \chi_g$ such that $f = vf' + 1 \in R_q^*$, compute $h = g/f$. Then, return the public key $pk = h$ to the adversary $\mathcal{A}_1$, and keep the secret key f private.

Challenge. Upon receiving two challenge plaintexts $(M_0, M_1) \in \{0,1\}^\ell \times \{0,1\}^\ell$ from the adversary $\mathcal{A}_1$, first randomly choose $\mu^* \leftarrow \{0,1\}, r^* \leftarrow \chi_r, e^* \leftarrow \chi_e$, compute $m^* = \mathsf{Pt2poly}(M_{\mu^*}) \in R_q$ and $c^* = hr^* + e^* + v^{-1}m^*$. Finally, return the challenge ciphertext c^* to $\mathcal{A}_2$.

Finalize. Upon receiving a guess $\mu' \in \{0,1\}$ from $\mathcal{A}_2$, return 1 if $\mu' = \mu^*$, otherwise return 0.

By definition, we have the following lemma.

Lemma 4. $|\Pr[F_0] - 1/2| = \epsilon$.

Game G_1. This game is similar to game G_0 except that the challenger $\mathcal{C}$ changes the KeyGen phase as follows:

KeyGen. randomly choose $h \leftarrow R_q$, and return the public key $pk = h$ to the adversary $\mathcal{A}_1$.

Lemma 5. *Under the* $\mathrm{DNTRU}_{n,q,\chi_f,\chi_g}$ *assumption, we have that Games G_1 and G_0 are computationally indistinguishable in the adversary's view. Moreover,* $|\Pr[F_1] - \Pr[F_0]| \leq \mathsf{negl}(\kappa)$.

Proof. This lemma directly follows from that the only difference between Games G_0 and G_1 is that $\mathcal{C}$ replaces $h = g/f$ in G_0 with a random one $h \leftarrow R_q$ in G_1.

Game G_2. This game is similar to game G_1 except that the challenger $\mathcal{C}$ changes the Challenge phase as follows:

Challenge. Upon receiving two challenge plaintexts $(M_0, M_1) \in \{0,1\}^\ell \times \{0,1\}^\ell$ from the adversary $\mathcal{A}_1$, first randomly choose $\mu^* \leftarrow \{0,1\}$ and $b \leftarrow R_q$, compute $m^* = \mathsf{Pt2poly}(M_{\mu^*}) \in R_q$ and $c^* = b + v^{-1}m^*$. Finally, return the challenge ciphertext c^* to $\mathcal{A}_2$.

Lemma 6. *Under the* $\mathrm{DRLWE}_{n,q,\chi_r,\chi_e}$ *assumption, we have that Games G_2 and G_1 are computationally indistinguishable in the adversary's view. Moreover,* $|\Pr[F_2] - \Pr[F_1]| \leq \mathsf{negl}(\kappa)$.

Proof. This lemma follows from that the only difference between Games G_1 and G_2 is that $\mathcal{C}$ replaces $b = hr^* + e^*$ in G_1 with a random one $b \leftarrow R_q$ in G_2.

Lemma 7. $|\Pr[F_2] - \frac{1}{2}| \leq \mathsf{negl}(\kappa)$.

Proof. This lemma directly follows from that b in Game G_2 is uniformly random, and statistically hides the information of m^* in $c^* = b + v^{-1}m^*$.

By Lemmas 4–7, we have that $\epsilon = |\Pr[F_0] - \frac{1}{2}| \leq \mathsf{negl}(\kappa)$. This completes the proof of Theorem 1.

3.3 An IND-CCA NTRU KEM from FO-Transformation

Let $\mathsf{NEV\text{-}PKE} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ be defined in the above subsection. Let $H_1 : \{0,1\}^* \to \{0,1\}^\kappa$, $H_2 : \{0,1\}^{\ell+\kappa} \to \{0,1\}^\kappa \times \{0,1\}^\kappa$ and $H_3 : \{0,1\}^* \to \{0,1\}^\kappa$ be three hash functions, which will be modeled as random oracles in the security proof. We now transform $\mathsf{NEV\text{-}PKE}$ into a IND-CCA secure KEM $\mathsf{NEV\text{-}KEM} = (\mathsf{KeyGen}, \mathsf{Encap}, \mathsf{Decap})$ following the generic FO-transformation.

- $\mathsf{NEV\text{-}KEM.KeyGen}(\kappa)$: given the security parameter κ as input, compute $(pk', sk') = \mathsf{NEV\text{-}PKE.KeyGen}(1^\kappa)$ and randomly choose $s \leftarrow \{0,1\}^\kappa$. Then, return the public key $pk = pk'$, and secret key $sk = (sk', pk, H_1(pk), s)$.
- $\mathsf{NEV\text{-}KEM.Encap}(pk, M)$: given the public key pk as input, randomly choose $M \leftarrow \{0,1\}^\ell$, and compute

$$(\bar{K}, \rho) = H_2(M, H_1(pk)), c = \mathsf{NEV\text{-}PKE.Enc}(pk, M; \rho) \text{ and } K = H_3(\bar{K}, c).$$

  Then, return the ciphertext and session key pair (c, K).
- $\mathsf{NEV\text{-}KEM.Decap}(sk, c)$: given the secret key $sk = (sk', pk, H_1(pk), s)$ and a ciphertext c as inputs, compute $M' = \mathsf{NEV\text{-}PKE.Dec}(sk', c)$, $(\bar{K}', \rho') = H_2(M', H_1(pk))$ and $c' = \mathsf{NEV\text{-}PKE.Enc}(pk, M', \rho')$. If $c' = c$, return $K = H_3(\bar{K}', c)$, otherwise, return $K = H_3(s, c)$.

Since $\mathsf{NEV\text{-}KEM}$ is obtained by a standard application of the FO transformation (with implicit rejection) to $\mathsf{NEV\text{-}PKE}$, the correctness of $\mathsf{NEV\text{-}KEM}$ directly follows from that of $\mathsf{NEV\text{-}PKE}$. Moreover, we have the following security theorem for $\mathsf{NEV\text{-}KEM}$ according to the studies in [15,18,26,28].

Theorem 2. *Let $n, q \in \mathbb{Z}$, $v \in R_q^*$ and distributions $\chi_f, \chi_g, \chi_r, \chi_e$ be defined as in Theorem 1. Then, under the* $\mathrm{DNTRU}_{n,q,\chi_f,\chi_g,v}$ *and* $\mathrm{DRLWE}_{n,q,\chi_r,\chi_e}$ *assumption, our KEM scheme* $\mathsf{NEV\text{-}KEM}$ *is provably IND-CCA secure in the (quantum) random oracle model.*

4 An Optimized NTRU Encryption from sspRLWE

Since in the typical application of using PKEs as KEMs, the session key is randomly chosen and not necessarily known in advance, one might wonder if we can somehow simplify the construction of $\mathsf{NEV\text{-}PKE}$ based on the assumption that the plaintext is random. In this section, we give an optimized NTRU encryption called $\mathsf{NEV\text{-}PKE'}$, which essentially merges the sampling of the noise and the plaintext in a single step: one can roughly think that the noise is the output of a random secret share algorithm with a random plaintext as input.

4.1 Randomized Plaintext Encoding and Decoding

Let n be a power of 2, and q be a prime. Let $R = \mathbb{Z}[x]/(x^n + 1)$ and $R_q = \mathbb{Z}_q[x]/(x^n+1)$. Let $\mathcal{M} = \{0,1\}^{n/k}$ be the plaintext space. Let $v = (1-x^{n/k}) \in R_q^*$ be a ring element, whose inverse is $v^{-1} = \frac{q+1}{2}(1 + x^{n/k} + \cdots + x^{(k-1)n/k}) \in R_q^*$. Let B_η be the binomial distribution with parameter $\eta \in \mathbb{Z}$. We define a pair of encoding and decoding algorithm $(\mathsf{Pt2noise}, \mathsf{Noise2Pt})$ as follows:

– $\mathsf{Pt2noise}(M, \eta)$: given a plaintext $M \in \{0,1\}^{n/k}$ and an integer η as inputs, first randomly choose $s \leftarrow \{0,1\}^{2n\eta - n/k}$, and parse $s = (s_0, \ldots, s_{2k\eta-2})$ as $(2k\eta - 1)$ blocks of n/k bits (i.e., $s_i \in \{0,1\}^{n/k}$ for all $i \in [2k\eta - 1]$). Then, set $s_{2k\eta-1} = M \oplus (\oplus_{i=0}^{2k\eta-2} s_i) \in \{0,1\}^{n/k}$, arrange the bit string $(s_0, \ldots, s_{2k\eta-1}) \in \{0,1\}^{2n\eta}$ as a bit array with 2η rows and n columns, and use the 2η bits in the i-th column as the randomness to sample the i-th coefficient of a polynomial $m \in R_q$ from B_η, as depicted in Fig. 3. Finally, return $m = m_0 + m_1 x + \cdots + m_{n-1} x^{n-1} \in R_q$, where

$$m_{in/k+j} = \sum_{t=0}^{\eta-1} (s_{2i\eta+t,j} - s_{2i\eta+\eta+t,j}) \text{ for } i \in [k], j \in [n/k].$$

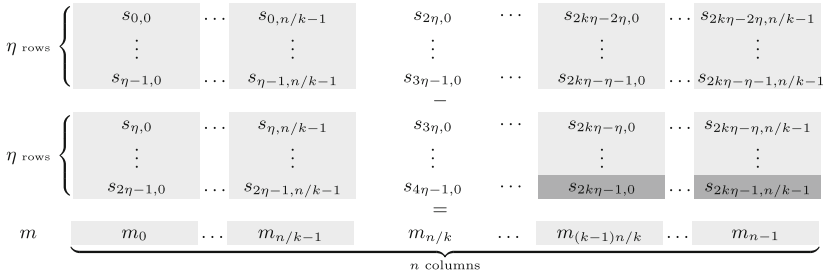

Fig. 3. The bit array for randomized encoding of a plaintext

– $\mathsf{Noise2Pt}(w)$: given a ring element $w \in R_q$ as input, compute and return $M = \mathsf{Poly2Pt}(w)$.

We have the following lemma for the above two algorithms.

Lemma 8. *Let $n, q, k, \eta \in \mathbb{Z}$ and $v \in R_q^*$ be defined as above. If M is uniformly chosen from $\{0,1\}^{n/k}$, then the coefficient distribution of $m = \mathsf{Pt2noise}(M, \eta)$ is identical to the binomial distribution B_η. Moreover, if $k\eta < \frac{q}{2}$, then for any $m = \mathsf{Pt2noise}(M, \eta)$ and any polynomial $e = e_0 + e_1 x + \cdots + e_{n-1} x^{n-1} \in R_q$ satisfying the following condition*

$$\left(\sum_{i=j \bmod n/k} \left| e_i \bmod^{\pm} q \right| \right) < \frac{k \cdot (q-1)}{4} - k\frac{k\eta + 1}{2} \text{ for } i \in [n] \text{ and } j \in [n/k] \tag{2}$$

we always have $\mathsf{Noise2Pt}(v^{-1} m + e) = M$.

Proof. The first claim directly follows from the fact that $(s_0, \ldots, s_{2k\eta-2})$ are uniformly chosen from $\{0,1\}^{2n\eta - n/k}$, and given $(s_0, \ldots, s_{2k\eta-2})$, $s_{2k\eta-1}$ is also uniformly distributed over $\{0,1\}^{n/k}$. Let $\bar{m} = \mathsf{Pt2poly}(M)$. By Lemma 3, it

suffices to show that $v^{-1}m = v^{-1}\bar{m} + e' \in R_q$ for some $\|e'\|_\infty \leq \frac{k\eta+1}{2}$. Formally, let $\bar{v} = (1 + x^{n/k} + \cdots + x^{(k-1)n/k})$, and $u = u_0 + u_1 x + \cdots + u_{n-1}x^{n-1} = \bar{v}m$, we have that $u_{in/k+j} = \sum_{t \leq i} m_{tn/k+j} - \sum_{t>i}^{k-1} m_{tn/k+j}$ for all $i \in [k], j \in [n/k]$. By the assumption that $k\eta < \frac{q}{2}$, we have that $u_{in/k+j} \in [-\frac{q-1}{2}, \frac{q-1}{2}]$ for all $i \in [k], j \in [n/k]$. Moreover, using a routine calculation one can check that $u_{in/k+j} = \sum_{t=0}^{k-1} m_{tn/k+j} = M_j \bmod 2$ for all $i \in [k], j \in [n/k]$ by the definition of m, and that there exists a polynomial e' such that $u = 2e' + \bar{v}\bar{m}$ and $\|e'\|_\infty \leq \frac{k\eta+1}{2}$ by the definition of $\bar{m}$. We immediately have $v^{-1}m = v^{-1}\bar{m} + e'$ using the fact that $v^{-1} = \frac{q+1}{2}\bar{v}$. This completes the proof.

Remark 5. Since $v^{-1}m + e = v^{-1}\bar{m} + e + e'$, the condition (2) in Lemma 8 can actually be relaxed to the following condition:

$$\left(\sum_{i=j \bmod n/k} \left| e_i + e'_i \bmod^{\pm} q \right| \right) < \frac{k \cdot (q-1)}{4} \text{ for } i \in [n] \text{ and } j \in [n/k]. \quad (3)$$

4.2 A OW-CPA Secure NTRU Encryption from sspRLWE

Let $n, q, k, \eta \in \mathbb{Z}$ and $v \in R_q^*$ be defined as above. Let χ_f, χ_g, χ_r be three distributions over R. We now give our PKE scheme NEV-PKE′, which consists of the following three algorithms (KeyGen, Enc, Dec):

- NEV-PKE′.KeyGen(κ): given the security parameter κ as inputs, randomly choose $f' \leftarrow \chi_f$ and $g \leftarrow \chi_g$ such that $f = f' + v^{-1} \in R_q^*$ is invertible. Then, return the public key and secret key pair $(pk, sk) = (h = g/f, f) \in R_q \times R_q$.
- NEV-PKE′.Enc(pk, M): given the public key $pk = h \in R_q$ and a plaintext $M \in \{0,1\}^{n/k}$ as inputs, sample $r \leftarrow \chi_r$ and $m \leftarrow \mathsf{Pt2noise}(M, \eta) \in R_q$. Then, compute and return the ciphertext $c = hr + m$.
- NEV-PKE′.Dec(sk, C): given the secret key $sk = f = f' + v^{-1} \in R_q^*$ and a ciphertext $c \in R_q$ as inputs, compute $u = fc$, and $M' = \mathsf{Noise2Pt}(u)$. Finally, return the plaintext $M' \in \{0,1\}^{n/k}$.

Remark 6. Note that if one wants to use NEV-PKE′ as a passively secure KEM, the encryption algorithm can be further simplified to directly sample a noise m from the binomial distribution B_η, and then derive a pre-session key $\bar{K}$ from the first n/k coefficients of $\bar{v}m \bmod 2$. By Lemma 8, this is actually equivalent to first randomly choose a prekey $\bar{K} \leftarrow \{0,1\}^{n/k}$ and then compute $m = \mathsf{Pt2noise}(\bar{K})$. We prefer to describe it as a PKE scheme because it supports the generic FO transformation in Sect. 3.3 to obtain an IND-CCA secure KEM.

Since we have the following decryption formula

$$w = fc = gr + (f' + v^{-1})m = \underbrace{gr + f'm}_{=\tilde{e}} + v^{-1}m = \tilde{e} + v^{-1}m.$$

the decryption is correct as long as $\tilde{e}$ satisfies the condition (2) in Lemma 8. We will choose concrete parameters such that the decryption failure is negligibly small in Sect. 5.2. For security, we have the following theorem.

Theorem 3. *Let $n, q, k, \eta \in \mathbb{Z}$ and $v = 1-x^{n/k}, \bar{v} = (1+x^{n/k}+\cdots+x^{(k-1)n/k}) \in R_q$ be defined as above. Let χ_f, χ_g, χ_r be three probability distributions over R_q. Then, under the* $\mathrm{DNTRU}_{n,q,\chi_f,\chi_g,v}$ *and* $\mathrm{sspRLWE}_{n,q,\chi_r,B_\eta,\bar{v}}$ *assumption, the above PKE scheme* NEV-PKE′ *is provably OW-CPA secure in the standard model.*

This proof is very similar to that of Theorem 1, we omit the details. By applying the same FO transformation in Sect. 3.3 to NEV-PKE′, we can obtain an IND-CCA secure KEM NEV-KEM′ in the (quantum) random oracle model.

4.3 On the Hardness of the SspRLWE Problem

In this subsection, we provide more evidences on the hardness of the problem $\mathrm{sspRLWE}_{n,q,\chi_r,B_\eta,\bar{v}}$ for binomial distribution B_η and $\bar{v} = (1 + x^{n/k} + \cdots + x^{(k-1)n/k}) \in R_2$. Specifically, we will first show that for discrete Gaussian noise distributions, the $\mathrm{sspRLWE}_{n,q,\chi_r,D_{\mathbb{Z}^n,\gamma},\bar{v}}$ problem is at least as hard as its standard decisional RLWE problem $\mathrm{DRLWE}_{n,q,\chi_r,D_{\mathbb{Z}^n,\beta}}$ for sufficiently large parameters $\gamma > \beta$, which can be extended to binomial distributions (with sufficiently large η) by a standard argument using Rényi divergence [5]. We will also prove two theorems for special cases of $\mathrm{sspRLWE}_{n,q,\chi_r,B_\eta,\bar{v}}$, which apply to η that is as small as 1. Formally, we have that following three theorems. A high-level intuition of the proofs for the theorems is already given in Sect. 1.2.

Theorem 4. *Let n, q, k, χ_r and $\bar{v}$ be defined as above. Let α, β, γ be three positive reals satisfying $\alpha \geq \omega(\sqrt{\log n}), \gamma = \sqrt{\alpha^2 + 4\beta^2}, 2\alpha\beta/\gamma \geq \sqrt{2} \cdot \omega(\sqrt{\log n})$ and $\gamma\sqrt{n} < q/2$. Let $D_{\mathbb{Z}^n,\beta}, D_{\mathbb{Z}^n,\gamma}$ be two discrete Gaussian distributions with parameter β and γ, respectively. If there is a PPT algorithm $\mathcal{A}$ solving the* $\mathrm{sspRLWE}_{n,q,\chi_r,D_{\mathbb{Z}^n,\gamma},\bar{v}}$ *problem (with probability negligibly close to 1), then there is another PPT algorithm $\mathcal{B}$ solving the* $\mathrm{DRLWE}_{n,q,\chi_r,D_{\mathbb{Z}^n,\beta}}$ *problem.*

Proof. It is sufficient to give the description of $\mathcal{B}$. Formally, given a DRLWE tuple $(a, b) \in R_q \times R_q$ as input, $\mathcal{B}$ first randomly chooses a polynomial $e' \in R_q$ from the distribution $D_{\mathbb{Z}^n,\alpha}$, and sets $(a', b') = (2a, 2b + e') \in R_q \times R_q$. Then, it runs algorithm $\mathcal{A}$ with input (a', b'), and obtains $w \in R_2$ from $\mathcal{A}$. Finally, $\mathcal{B}$ returns 1 if $w = \bar{v}e' \bmod 2$, otherwise returns 0.

We now analyze the behavior of algorithm $\mathcal{B}$. First, if $(a, b = ar + e)$ is a real $\mathrm{DRLWE}_{n,q,\chi_r,D_{\mathbb{Z}^n,\beta}}$ tuple, then we have that the coefficients of e are chosen from $D_{\mathbb{Z}^n,\beta}$, which means that the coefficient distribution of $2e$ follows the distribution of $D_{2\mathbb{Z}^n,2\beta}$. By Lemma 2, we have that the distribution of $\hat{e} = 2e + e'$ is statistically close to $D_{\mathbb{Z}^n,\gamma}$. Since $\gamma\sqrt{n} < q/2$, we have that $\|\hat{e}\|_\infty < q/2$ with probability negligibly close to 1 by Lemma 1, which means that $\hat{e} \bmod q = \hat{e}$ holds with probability negligibly close to 1. Thus, the distribution of $(a' = 2a, b' = 2ar + \hat{e}) \in R_q \times R_q$ is statistically close to an $\mathrm{sspRLWE}_{n,q,\chi_r,D_{\mathbb{Z}^n,\gamma},\bar{v}}$ tuple. Using the fact that $w = \bar{v}\hat{e} = \bar{v}e' \bmod 2$, we have that $\mathcal{B}$ will return 1 with probability negligibly close to 1. Second, if (a, b) is randomly chosen from $R_q \times R_q$, we have that $(a' = 2a, b' = 2b + e')$ is also randomly distributed over $R_q \times R_q$. This

means that the probability for any $\mathcal{A}$ to output $w \in R_2$ such that $w = \bar{v}e' \bmod 2$ is negligible in n/k by our choice of $e' \leftarrow D_{\mathbb{Z}^n,\alpha}$ with $\alpha \geq \omega(\sqrt{\log n})$. In all, we have shown that $\mathcal{B}$ is a valid distinguisher for $\mathrm{DRLWE}_{n,q,\chi_r,D_{\mathbb{Z}^n,\beta}}$ problem. This completes the proof.

Remark 7. As commonly seen in lattice-based cryptography, Theorem 4 does not provide concrete guarantee for practical parameters with typically small η. In the following, we show that for any $\eta \geq 1$, the $\mathrm{sspRLWE}_{n,q,\chi_r,B_\eta,\bar{v}}$ problem for $k = 1$ (resp., $k = 2$) is at least as hard as the standard $\mathrm{RLWE}_{n,q,\chi_r,\chi_e}$ problem with binomial distribution $\chi_e = B_1$ (resp., uniform binary distribution $\chi_e = U(R_2)$), where the case $k = 2$ essentially corresponds to our concrete parameter set NEV'-512.

Theorem 5. *Let $n, q, k, \chi_r, \eta, \bar{v}$ be defined as above, and $\eta < \frac{q}{2}$. If there is a PPT algorithm $\mathcal{A}$ solving the $\mathrm{sspRLWE}_{n,q,\chi_r,B_\eta,\bar{v}}$ problem for $k = 1$ (with probability negligibly close to 1), then there is another PPT algorithm $\mathcal{B}$ solving the $\mathrm{RLWE}_{n,q,\chi_r,B_1}$ problem.*

Proof. We now give the description of $\mathcal{B}$. Formally, given an $\mathrm{RLWE}_{n,q,\chi_r,B_1}$ instance $(a, b = ar + e)$ as input, $\mathcal{B}$ first randomly chooses a polynomial $e' \in R_q$ with coefficients sampling from the distribution $B_{\eta-1}$, and sets $b' = b + e' \in R_q$. Since $\eta \leq \frac{q-1}{2}$, it is easy to check that the coefficients of $\hat{e} = e + e' \bmod q = e + e'$ follows the distribution B_η, and that $(a, b' = ar + \hat{e})$ is an $\mathrm{sspRLWE}_{n,q,\chi_r,B_\eta,\bar{v}}$ instance. Then, it runs algorithm $\mathcal{A}$ with input (a, b'), which is expected to return $\bar{v}\hat{e} \bmod 2$ in polynomial time. Next, $\mathcal{B}$ computes $\bar{v}\hat{e} + \bar{v}e' = \bar{v}e \bmod 2$. Note that $\bar{v} = 1$ for $k = 1$. Let $u = \bar{v}e = e$, where $u = u_0 + u_1 + \cdots + u_{n-1}x^{n-1}$ and $e = e_0 + e_1 x + \cdots + e_{n-1}x^{n-1}$. Since $e_i \in \{-1, 0, 1\}$, we have that $u_i \bmod 2 = 0$ if and only if $e_i = 0$. Thus, $\mathcal{B}$ can expect to obtain $n/2$ equations on the n variables consisting of the coefficients of the secret r from $(a, b = ar + e)$. Let d be the order of q modulo $2n$, we have that $x^n + 1$ modulo q factors into n/d irreducible polynomials of the same degree d, the probability that a random $a \leftarrow R_q$ is invertible is $(1 - \frac{1}{q^d})^{n/d} \geq 1/2$. Thus, with probability greater than $1/2$ we have that those obtained equations are linearly independent. By repeating the above process using fresh $\mathrm{RLWE}_{n,q,\chi_r,B_1}$ instances at most a polynomial number of times, $\mathcal{B}$ can collect n linearly independent equations to recover all the n coefficients of r by using Gaussian elimination. In all, $\mathcal{B}$ can solve the $\mathrm{RLWE}_{n,q,\chi_r,B_1}$ problem in polynomial time. This completes the proof.

Theorem 6. *Let $n, q, k, \chi_r, \eta, \bar{v}$ be defined as above, and $\eta < \frac{q}{2}$. If there is a PPT algorithm $\mathcal{A}$ solving the $\mathrm{sspRLWE}_{n,q,\chi_r,B_\eta,\bar{v}}$ problem for $k = 2$ (with probability negligibly close to 1), then there is another PPT algorithm $\mathcal{B}$ solving the $\mathrm{RLWE}_{n,q,\chi_r,\mathcal{U}(R_2)}$ problem.*

Proof. In order to prove Theorem 6, it suffices to prove the following two claims:

Claim 1. $\mathrm{sspRLWE}_{n,q,\chi_r,\mathcal{U}(R_2),\bar{v}} \Rightarrow \mathrm{sspRLWE}_{n,q,\chi_r,B_\eta,\bar{v}}$**:** If there is a PPT algorithm $\mathcal{A}$ solving $\mathrm{sspRLWE}_{n,q,\chi_r,B_\eta,\bar{v}}$, then there is another PPT algorithm $\bar{\mathcal{A}}$ solving $\mathrm{sspRLWE}_{n,q,\chi_r,\mathcal{U}(R_2),\bar{v}}$.

Claim 2. $\mathrm{RLWE}_{n,q,\chi_r,\mathcal{U}(R_2)} \Rightarrow \mathrm{sspRLWE}_{n,q,\chi_r,\mathcal{U}(R_2),\bar{v}}$**:** If there is a PPT algorithm $\bar{\mathcal{A}}$ solving $\mathrm{sspRLWE}_{n,q,\chi_r,\mathcal{U}(R_2),\bar{v}}$, then there is another PPT algorithm $\mathcal{B}$ solving $\mathrm{RLWE}_{n,q,\chi_r,\mathcal{U}(R_2)}$.

For Claim 1, we construct an algorithm $\bar{\mathcal{A}}$ as follows. Formally, given an $\mathrm{sspRLWE}_{n,q,\chi_r,\mathcal{U}(R_2),\bar{v}}$ instance $(a, b = ar + e) \in R_q \times R_q$ as input, $\bar{\mathcal{A}}$ first randomly chooses a polynomial $e' \in R_q$ with coefficients sampling from the following distribution

$$B'_\eta = \left\{ \sum_{i=0}^{\eta-1} (a_i - b_i) : (a_0, \ldots, a_{\eta-2}, b_0, \ldots, b_{\eta-1}) \leftarrow \{0,1\}^{2\eta-1} \right\}$$

in time $O(n\eta)$ and computes $(a, b' = b + e') = as + (e + e') \in R_q$. Since $\eta \leq \frac{q-1}{2}$, it is easy to check that the coefficients of $\hat{e} = e + e' \bmod q = e + e'$ follows the distribution B_η, and that (a, b') is an $\mathrm{sspRLWE}_{n,q,\chi_r,B_\eta,\bar{v}}$ instance. Then, it runs algorithm $\mathcal{A}$ with input (a, b'), which is expected to return $\bar{v}\hat{e} \bmod 2$ in polynomial time. Finally, it returns $\bar{v}\hat{e} + \bar{v}e' = \bar{v}e \bmod 2$. This shows that $\bar{\mathcal{A}}$ can output $\bar{v}e \bmod 2$ in polynomial time. This completes the proof of Claim 1.

We now define an algorithm $\mathcal{B}$ for Claim 2 as follows. Formally, given an $\mathrm{RLWE}_{n,q,\chi_r,\mathcal{U}(R_2)}$ instance $(a, b = as + e)$ as input, it first runs algorithm $\bar{\mathcal{A}}$ with input (a, b) , which is expected to return $\bar{v}e \bmod 2$ in polynomial time. Note that $\bar{v} = 1 + x^{\frac{n}{2}}$ for $k = 2$. Let $u = \bar{v}e$, we have

$$u_j = \begin{cases} e_j - e_{\frac{n}{2}+j} \in \{-1, 0, 1\}, & \text{if } j \in [\frac{n}{2}] \\ e_j + e_{j-\frac{n}{2}} \in \{0, 1, 2\}, & \text{otherwise,} \end{cases}$$

where $u = u_0 + u_1 + \cdots + u_{n-1}x^{n-1}$ and $e = e_0 + e_1 x + \cdots + e_{n-1}x^{n-1} \in R_2$. Thus, we have that $u_j \bmod 2 = 0$ if and only if $u_j = 0$ for all $j \in [\frac{n}{2}]$ and that $u_j \bmod 2 = 1$ if and only if $u_j = 1$ for all $j \geq \frac{n}{2}$. Thus, $\mathcal{B}$ can expect to obtain $n/2$ equations on the n variables consisting of the coefficients of secret s from $(\bar{v}a, \bar{v}b = \bar{v}as + \bar{v}e)$. Let d be the order of q modulo $2n$, we have that $x^n + 1$ modulo q factors into n/d irreducible polynomials of the same degree d, the probability that a random $a \leftarrow R_q$ is invertible is $(1 - \frac{1}{q^d})^{n/d} \geq 1/2$. Thus, with probability greater than $1/2$ we have that those obtained equations are linearly independent. By repeating the above process using fresh $\mathrm{RLWE}_{n,q,\chi_r,\mathcal{U}(R_2)}$ instances a polynomial number of times, $\mathcal{B}$ can collect n linearly independent equations to recover all the n coefficients of s by using Gaussian elimination. In all, $\mathcal{B}$ can solve the $\mathrm{RLWE}_{n,q,\chi_r,\mathcal{U}(R_2)}$ problem in polynomial time. This completes the proof.

5 Concrete Attacks and Parameters

As discussed in [18], the most efficient known attacks against the NTRU and RLWE problems are lattice attacks. In this section, we mainly show how to apply lattice attacks to our (variants of) NTRU and RLWE problems, and take account of other relevant attacks by directly using the LWE estimator script [1] to obtain the concrete security estimates for our recommended parameters.

5.1 Lattice Attacks Against NTRU and (ssp)RLWE

In general, the lattice attacks against NTRU and RLWE problems work by defining the same set

$$\mathcal{L}_c^{\perp}(h) = \{(u, w) \in R_q = \mathbb{Z}[x]/(x^n + 1) : hu + w = c \in R_q\}.$$

The NTRU problem correspond to the special case $c = 0$, and $\mathcal{L}_0^{\perp}(h)$ essentially forms a lattice. To solve the decisional NTRU problem, namely, to distinguish the quotient $h = g/(vf' + 1) \in R_q$, where f', g have small coefficients noticeably less than $\sqrt{q/3}$, from a uniformly-random $h \in R_q$, an algorithm can try to find a good approximation to the shortest vector in $\mathcal{L}_0^{\perp}(h)$ [18]. This is because the vector $(f = vf' + 1, -g)$ will be a short vector significantly less than $\sqrt{nq}$ for $h = g/f$ (recall that $v = 1 - x^{n/k}$ is small in our case), while a vector of ℓ_2-norm less than $\Omega(\sqrt{nq})$ is very unlikely to exist in $\mathcal{L}_0^{\perp}(h)$ for a random $h \in R_q$.

For RLWE problems, we have $c \neq 0$ for $(h, c = hr + e)$, and $\mathcal{L}_c^{\perp}(h)$ is a shift of the lattice $\mathcal{L}_0^{\perp}(h)$. Finding the shortest vector in it is known as the Bounded Distance Decoding (BDD) problem, which in turn can be solved by finding the short vector $(e, r, 1) \in \mathbb{Z}^{2n+1}$ in an embedding lattice with dimension $2n + 1$ and basis

$$\mathbf{B} = \begin{pmatrix} qI_n & \mathsf{Rot}(h) & \mathbf{c} \\ 0 & I_n & 0 \\ 0 & 0 & 1 \end{pmatrix} \in \mathbb{Z}^{(2n+1)\times(2n+1)},$$

where $\mathsf{Rot}(h) \in \mathbb{Z}_q^n \times \mathbb{Z}_q^n$ is the anti-circular matrix corresponding ring multiplication in R_q, and $\mathbf{c} \in \mathbb{Z}_q^n$ is the coefficient vector of $c \in R_q$ in column form. For the same secret and noise distributions, the complexity of attacking the NTRU and RLWE problems are typically identical for modulus $q = O(n)$. Since for RLWE problems we can directly use the LWE estimator to obtain concrete security estimates, it suffices to how to use the LWE estimator to obtain concrete security estimates for our NTRU and sspRLWE problems.

On the $\mathrm{DNTRU}_{n,q,\chi_f,\chi_g,v}$ *Problem with* $v = 1 - x^{n/k}$ *over* $R_q = \mathbb{Z}[x]/(x^n + 1)$. First, as discussed in Sect. 2.4, for the setting that $v = 1 - x^{n/k} \in R_q$ is invertible, our NTRU problem $\mathrm{DNTRU}_{n,q,\chi_f,\chi_g,v}$ is essentially equivalent to the standard NTRU problem (with $v = 3$) up to the choices of the secret key distribution. Second, the ℓ_2-norm of $vf' + 1$ is only roughly about $\sqrt{2}$ times larger than that of f', which is small as long as f' is chosen from a small distribution. Thus, one can either solve the NTRU problem by taking $f = vf' + 1$ as whole just as in the standard lattice attacks against the NTRU problem with secret distributions $(\chi_f' = v\chi_f, \chi_g)$ in lattice $\mathcal{L}_0^{\perp}(h)$ for $h = g/f$, or solve the BDD problem on the shifted lattice $\mathcal{L}_h^{\perp}(-vh)$ by treating it as an RLWE instance $(vh, h = -vhf' + g)$ with secret distribution χ_f and noise distribution χ_g. We use the latter for concrete estimates for our NTRU problems in the LWE estimator because the norm of the short vector $(g, f', 1)$ in the latter case (which is independent from v) is smaller than that of $(f = vf' + 1, -g)$ in the former case.

Table 4. Practical Parameters Sets for Our KEM Schemes

Parameters	(n, q)	Key Dist (χ_f, χ_g)	Enc Dist (χ_r, χ_e)	Size (PK, CT)	Dec Failure	BKZ Sizes (SK, CT)	LWE Estimator (SK, CT)
NEV-512	(512, 769)	(B_1, B_1)	$(B_1, \mathcal{T}_{1/6})$	(615,615)	2^{-138}	(426, 413)	(145, 141)
NEV′-512	(512, 769)	(B_1, B_1)	(B_1, B_1)	(615,615)	2^{-200}	(426, 426)	(145, 145)
NEV-1024	(1024, 769)	(B_1, B_1)	$(B_1, \mathcal{T}_{1/6})$	(1229,1229)	2^{-152}	(953, 929)	(292, 281)
NEV′-1024	(1024, 769)	(B_1, B_1)	(B_1, B_1)	(1229,1229)	2^{-200}	(953, 953)	(292, 292)

On the $\text{sspRLWE}_{n,q,\chi_r,B_\eta,v}$ *Problem over* $R_q = \mathbb{Z}[x]/(x^n + 1)$. In Sect. 4.3, we have shown that the $\text{sspRLWE}_{n,q,\chi_r,B_\eta,v}$ problem is polynomially equivalent to the standard RLWE problem (with different parameters). Although those reductions are too loose to estimate concrete estimates on practical parameters, we believe it is very reasonable to assume that the concrete hardness of the $\text{sspRLWE}_{n,q,\chi_r,B_\eta,v}$ problem with $v = 1+x^{n/k}+\cdots+x^{(k-1)n/k}$ is the same as that of $\text{RLWE}_{n,q,\chi_r,B_\eta}$. Note that similar assumption for RLWE2 is also made in [18]. Thus, we estimate the concrete hardness of the $\text{sspRLWE}_{n,q,\chi_r,B_\eta,v}$ problem by treating it as a standard RLWE problem $\text{RLWE}_{n,q,\chi_r,B_\eta}$ in the LWE estimator.

5.2 Recommended Parameters

In Table 4, we present two parameter sets NEV-512 and NEV-1024 for NEV-PKE and NEV-KEM, along with two parameter sets NEV′-512 and NEV′-1024 for NEV-PKE′ and NEV-KEM′, aiming at NIST levels 1 and 5 security, respectively. The fifth column gives the corresponding sizes of public key (PK) and ciphertext (CT). The sixth column presents the decryption failure probability, which is computed by using a python script adapted from the python script for Kyber [9]. Note that we make the same choice as Kyber [9] to set our decryption failure probabilities $< 2^{-128}$ with some margin so that it is infeasible to obtain a single decryption failure using at most 2^{64} decryption queries (see the directional failure boosting attacks [14]). The seventh column gives the BKZ blocksizes needed to break the security of the secret key (SK) and ciphertext (CT) for each parameter set in the core-SVP model [3]. The last column presents concrete security estimates obtained by running the LWE estimator [1]. As known schemes using the power of 2 cyclotomic ring for both security and performance considerations such as Newhope [3], we cannot find a proper parameter set for NIST level 3 security. Fortunately, as shown in Tables 1 and 2, the performance of our schemes using the parameter sets at NIST level 5 security is already comparable to that of known schemes using parameter sets aiming at NIST level 3 security. For example, in the application of ephemeral key exchanges, our NEV-KEM using the parameter set NEV-1024 has the same size as that of NTRU4096821 and is 4.10–11.05X faster. Compared to Kyber768, our NEV-KEM using NEV-1024 has size about 8.19% larger but is 1.2X faster. Thus, we do not think this security gaps for our parameter sets will be a real problem for practical use: one can simply use NEV-1024 (or NEV′-1024) for applications requiring NIST level 3 security.

6 Implementations

We made two implementations of our schemes: one uses the reference C language, and the other is (partially) optimized by using AVX2 instructions. In the following, we provide some implementation details that heavily affect the performance of our schemes.

6.1 Partial NTT Multiplication

One costly arithmetic operation in our schemes is to do polynomial multiplication in R_q. Since the use of small modulus $q = 769$, we cannot apply full NTT multiplications in $R_q = \mathbb{Z}[x]/(x^n+1)$ for both $n = 512$ and 1024. But because $q-1 \bmod 256 = 1$, we can still speedup polynomial multiplications by first splitting the polynomials in R_q to a set of sub-polynomials in $R'_q = \mathbb{Z}_q[y]/(y^{128}+1)$ and then realize a single polynomial multiplication in R_q by using a number of polynomial multiplications in $R'_q = \mathbb{Z}_q[y]/(y^{128}+1)$, which in turn can be done efficiently using full NTT multiplications. Taking $n = 512$ as an example, by letting $y = x^4$ we can split any two polynomials $a, b \in R_q = \mathbb{Z}[x]/(x^{512}+1)$ as follows:

$$\begin{aligned} a(x) &= a_0(y) + xa_1(y) + x^2a_2(y) + x^3a_3(y) \\ b(x) &= b_0(y) + xb_1(y) + x^2b_2(y) + x^3b_3(y), \end{aligned}$$

where all the a_i's and b_i's are polynomials in $R'_q = \mathbb{Z}_q[y]/(y^{128}+1)$. Since multiplications between a_i's and b_j's can be done using full NTT multiplications in R'_q, we can realize the multiplication between $a(x)$ and $b(x)$ by roughly using 16 NTT multiplications in $R'_q = \mathbb{Z}_q[y]/(y^{128}+1)$ as follows:

$$\begin{aligned} a(x) \cdot b(x) = &(a_0b_0 + y(a_1b_3 + a_2b_2 + a_3b_1)) \\ &+x(a_0b_1 + a_1b_0 + y(a_2b_3 + a_3b_2)) \\ &+x^2(a_0b_2 + a_1b_1 + a_2b_0 + ya_3b_3) \\ &+x^3(a_0b_3 + a_1b_2 + a_2b_1 + a_3b_0). \end{aligned}$$

We can further save 6 NTT multiplications in R'_q by using the Karatsuba method as observed in [46]. For example, to compute the term $a_1b_3 + a_3b_1$ in the first row, we only need a single NTT multiplication by computing $a_1b_3 + a_3b_1 = (a_1 + a_3)(b_1 + b_3) - a_1b_1 - a_3b_3$ given as inputs a_1b_1 and a_3b_3, which will be computed in the third row.

To facilitate the above polynomial multiplications, we directly represent each polynomial in $R_q = \mathbb{Z}[x]/(x^n+1)$ by simply concatenating the coefficient vectors of its split sub-polynomials, which are almost for free when all the coefficients are identically chosen from the same distribution. Moreover, we will keep the split sub-polynomials for the public key, secret key and ciphertext in their NTT forms to save some forward and inverse NTT operations in $R'_q = \mathbb{Z}_q[y]/(y^{128}+1)$.

6.2 Partial NTT Inversion

The other costly arithmetic operation is to do polynomial inversion in R_q to generate the public key. Note that if we can do full NTT multiplications in R_q,

this operation can be simply done by using n inversions in $\mathbb{Z}_q$ using the NTT representation. Fortunately, we can still speedup this operation by making full use of partial NTT multiplications given above as shown in [20]. Specifically, given a polynomial $f \in R_q = \mathbb{Z}_q[x]/(x^n + 1)$, by letting $z = x^2$ we can first use Karatsuba with an even/odd split to obtain two sub-polynomials in $\hat{R}_q = \mathbb{Z}_q[z]/(z^{n/2} + 1)$:

$$f(x) = f_0(z) + xf_1(z).$$

Then, the inversion of f in R_q can be done using one polynomial multiplication in R_q and one polynomial inversion in $\hat{R}_q$ because

$$\frac{1}{f(x)} = \frac{f_0(z) - xf_1(z)}{(f_0(z) + xf_1(z))(f_0(z) - xf_1(z))} = \frac{f_0(z) - xf_1(z)}{f_0^2(z) - zf_1^2(z)}.$$

By repeating this process, we can finally reduce the inversion of f to a few polynomial multiplications in R_q and a single polynomial inversion in $R'_q = \mathbb{Z}_q[y]/(y^{128} + 1)$, which in turn can be done using 128 inversions in $\mathbb{Z}_q$. Since $q = 769$ is very small, we can simply precompute the inversion table for all the elements in $\mathbb{Z}_q$. This is main reason why the key generation algorithm is much faster than NTRU (and some of its variants not using NTT).

6.3 Symmetric Primitives

In our default implementations, we use SHA3 and SHAKE256 as the hash function and the pseudorandom generator (PRG), respectively, which are the same as that of NTRU and Kyber in the NIST PQC submissions. Since the arithmetic operation of our KEMs is so fast that the use of SHA3 and SHAKE256 become the main bottleneck of our schemes: we actually observe a 1.82–2.27X speedup in experiment by replacing SHA3 and SHAKE256 with BLAKE2 and AES256CTR in the AVX2 implementation. For a fair comparison, we will use the same hash and PRG functions as that of BAT and NTTRU in the comparison with them (see Tables 5 and 6): BLAKE2 is used as both the hash and PRG functions in the open source code of BAT [20]; SHA3 and AES256CTR are used as the hash and PRG functions respectively in the open source code of NTTRU [33].

6.4 Multi-target Countermeasure

In the description of our IND-CCA transform in Sect. 3.3, we follow the strategy of Kyber to hash the public key into a prekey $\bar{K}$ and the random coins ρ, aiming at improving the security against multi-target attacks. We also hash the prekey together with the ciphertext into the final session key to make sure that our KEMs are contributory. The above two countermeasures are applied in our default implementations and in efficiency comparison with NTRU and Kyber (see Table 2). Since the performance of symmetric primitives is a major bottleneck of our schemes, those countermeasures will significantly reduce the performance: we observe a 2.25–2.54X speedup in experiment by removing the

two countermeasures in the AVX2 implementation using SHA3 and SHAKE256 as the hash function and the pseudorandom generator (PRG), respectively. Since both BAT [20] and NTTRU [33] do not apply those countermeasures, we turned off the countermeasures in the comparison with them (see Tables 5 and 6).

6.5 Compressed Representation of R_q Elements

We apply the strategy of [20] to store an element in R_q in the compressed form. In particular, we encode coefficients by groups of 5 in 48 bits: each coefficient is split into a low 3 bits and a high 7 bits (value 0 to 96, inclusive); 5 "high bits" are encoded using 33 bits in base 97. For $n = 512$ (resp., 1024), this will lead to a reduction of 25 (resp., 51) bytes in storing a polynomial in R_q. The encoding can be done very efficiently using about 300 (resp., 600) CPU cycles, but the decoding is really costly, and will take about 1200 (resp., 2400) CPU cycles, which is about 3.1X (resp., $1.6X$) slower than a polynomial multiplication in the same dimension. Thus, for applications that the few reduction in size is not very crucial, we highly recommend to remove this encoding/decoding optimization, and to obtain significantly speedup in efficiency especially when fast symmetric primitives are used (see Table 6).

7 Benchmarks and Comparisons

We run the codes of our schemes and several related works on the same 64-bit CentOS Linux 7.6 system (equipped with an Intel Core-i7 4790 3.6 GHz CPU and 4 GB memory), and present the average number of CPU cycles (over 100000 times) for running the corresponding algorithms in Tables 2, 5 and 6. All the codes are complied using the same optimization flags "-O3 -march=native -mtune=native -fomit-frame-pointer".

In Table 2, we give an efficiency comparison between our NEV-KEMs, NTRU and Kyber. The timings for our KEMs are obtained using our default implementations. In particular, we use SHA3 and SHAKE256 as the hash and PRG functions, which are the same as that in the code of Kyber and NTRU, submitted to the NIST PQC standardizations. We also use the multi-target countermeasures to hash the public key to generate the prekey and the random coins, and hash the ciphertext to generate the final session key. From Table 2, one can see that our scheme NEV-KEM (which is based on NEV-PKE from the standard NTRU and RLWE assumption) is 5.03–29.94X faster than NTRU (with key generation being 13.56–88.28X faster, encapsulation being 1.42–2.63X faster, and decapsulation being 2.39–2.99X faster) and 1.42–1.74X faster than Kyber, in the round-trip time of ephemeral key exchange at the same security levels. The efficiency improvement over Kyber is mainly because we do not have to expand a random coins to a uniform matrix over R_q, which needs many calls to the underlying symmetric primitives for rejection sampling. It is also worth to note that our NEV-KEM using the parameter set NEV-1024 at NIST level 5 security has the same public key and ciphertext size as that of NTRU-HPS4096821

at NIST level 3 security, but is 4.10–11.05X faster (with key generation being 11.96–31.94X faster, encapsulation being 1.36–1.51X faster, and decapsulation being 1.08–1.55X faster). The main reason that our KEMs is much faster than NTRU is that we allow (partial) NTT multiplications and inversions in R_q.

Table 5. Comparison between our NEV-KEMs and BAT in efficiency (CPU Cycles)

Schemes	KeyGen (Ref)	Encap (Ref)	Decap (Ref)	KeyGen (AVX2)	Encap (AVX2)	Decap (AVX2)	Speedup (Ref/AVX2)
BAT-512	35 249k	55 930	297 472	33 305k	10 191	68 795	140.5/973.6X
NEV-512	79 465	69 244	104 760	8 202	12 661	13 424	–
NEV′-512	79 220	62 261	101 367	8 224	9 017	10 272	–
BAT-1024	182 931k	111 694	818 690	156 811k	20 387	144 357	334.7/2648.3X
NEV-1024	182 198	144 157	223 059	16 001	18 844	24 429	–
NEV′-1024	182 619	134 622	225 169	15 938	16 109	21 274	–

In Table 5, we give a comparison between our NEV-KEMs and BAT. The timings for our KEMs are obtained using BLAKE2 as the hash and PRG functions without multi-target countermeasures, which are the same as that in the public available code of BAT. The size of BAT is about 19.19% (resp., 9.03%) than our Π_{KEM} at NIST level 1 (resp., 5) security (see Table 3), but our NEV-KEM is about 140-973X (resp., 334-2648X) faster than BAT, with key generation being 443-4060X (resp., 1004-9800X) faster, and decapsulation being 2.84–5.12X(resp., 3.67–5.90X) faster. Our encapsulation is slightly slower than that of BAT (especially in the reference implementation) mainly because BAT uses the strong RLWR assumption with binary secret and only needs to generate a few random bits in encapsulation. The efficiency improvement over BAT is mainly because we do not use the heavy trapdoor inversion algorithm, which requires very complex key generation and decryption operations.

Table 6. Comparison between our NEV schemes and NTTRU in efficiency (CPU Cycles)

Schemes	KeyGen (PKE)	Encap (PKE)	Decap (PKE)	KeyGen (KEM)	Encap (KEM)	Decap (KEM)
NEV-512	4 439	3 636	1 378	5 107 (4 881)	6 419 (5 289)	9 612 (6 675)
NEV′-512	4 453	3 112	1 399	5 201 (4 800)	6 167 (4 683)	9 382 (6 565)
NTTRU-768	8 199	2 976	2 675	9 640	6 586	8 603
NEV-1024	9 467	7 595	3 484	10 715 (9 891)	11 534 (9 841)	19 340 (12 955)
NEV′-1024	9 211	6 733	3 275	10 887 (10 195)	10 281 (8 864)	17 803 (11 531)

In Table 6, we give a comparison with NTTRU using the AVX2 instructions. The columns 2–4 present the timings for the corresponding OW/IND-CPA PKEs, while columns 4–7 give the timings for the final IND-CCA KEMs. The timings for our schemes are obtained using SHA3 and AES256CTR as the hash and PRG functions without multi-target countermeasures, which are the same as that in the public available code of NTTRU. The figures in the brackets give the timings of our NEV-KEMs without using the compressed representation of R_q elements. We note that NTTRU only supports the parameter of $n = 768, q = 7681$ in the cyclotomic ring $\mathbb{Z}_q[x]/(x^n - x^{n/2} + 1)$, aiming at NIST level 3 security. A recent paper [18] presents more parameter sets (see NTRU-A in Table 3) with reported comparable efficiency over the same ring as NTTRU, but their implementation is not publicly available. From Tables 3 and 6, we can expect that our schemes would have comparable computational efficiency with NTTRU and NTRU-A, but is at least 28% more compact, at the same security levels.

Acknowledgements. We thank the anonymous reviewers of ASIACRYPT 2023 for their helpful comments and suggestions on earlier version of our paper. This paper is supported by the National Natural Science Foundation of China (Grant Nos. 62022018, 61932019), and by the National Key Research and Development Program of China (Grant No. 2022YFB2702000).

References

1. Albrecht, M.R., Player, R., Scott, S.: On the concrete hardness of learning with errors. J. Math. Cryptol. **9**, 169–203 (2015)
2. Alkim, E., et al.: Newhope - submission to the NIST post-quantum project (2020)
3. Alkim, E., Ducas, L., Pöppelmann, T., Schwabe, P.: Post-quantum key exchange-a new hope. In: USENIX Security Symposium 2016 (2016)
4. Alkim, E., Ducas, L., Pöppelmann, T., Schwabe, P.: NewHope without reconciliation. Cryptology ePrint Archive, Report 2016/1157 (2016)
5. Bai, S., Langlois, A., Lepoint, T., Stehlé, D., Steinfeld, R.: Improved security proofs in lattice-based cryptography: using the Rényi divergence rather than the statistical distance. In: Iwata, T., Cheon, J.H. (eds.) ASIACRYPT 2015. LNCS, vol. 9452, pp. 3–24. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48797-6_1
6. Bailey, D.V., Coffin, D., Elbirt, A., Silverman, J.H., Woodbury, A.D.: NTRU in constrained devices. In: Koç, Ç.K., Naccache, D., Paar, C. (eds.) CHES 2001. LNCS, vol. 2162, pp. 262–272. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44709-1_22
7. Banaszczyk, W.: New bounds in some transference theorems in the geometry of numbers. Math. Ann. **296**, 625–635 (1993)
8. Bindel, N., Hamburg, M., Hövelmanns, K., Hülsing, A., Persichetti, E.: Tighter proofs of CCA security in the quantum random oracle model. In: Hofheinz, D., Rosen, A. (eds.) TCC 2019. LNCS, vol. 11892, pp. 61–90. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-36033-7_3
9. Bos, J., et al.: Crystals - Kyber: a CCA-secure module-lattice-based KEM. In: 2018 IEEE European Symposium on Security and Privacy (EuroS P), pp. 353–367 (2018)

10. Brakerski, Z., Döttling, N.: Lossiness and entropic hardness for ring-LWE. In: Pass, R., Pietrzak, K. (eds.) TCC 2020. LNCS, vol. 12550, pp. 1–27. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-64375-1_1
11. Chen, C., et al.: NTRU - submission to the NIST post-quantum project (2019)
12. Cramer, R., Shoup, V.: A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, pp. 13–25. Springer, Heidelberg (1998). https://doi.org/10.1007/BFb0055717
13. Dachman-Soled, D., Ducas, L., Gong, H., Rossi, M.: LWE with side information: attacks and concrete security estimation. In: Micciancio, D., Ristenpart, T. (eds.) CRYPTO 2020. LNCS, vol. 12171, pp. 329–358. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-56880-1_12
14. D'Anvers, J.-P., Rossi, M., Virdia, F.: *(One) failure is not an option*: bootstrapping the search for failures in lattice-based encryption schemes. In: Canteaut, A., Ishai, Y. (eds.) EUROCRYPT 2020. LNCS, vol. 12107, pp. 3–33. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-45727-3_1
15. Don, J., Fehr, S., Majenz, C., Schaffner, C.: Online-extractability in the quantum random-oracle model. In: Dunkelman, O., Dziembowski, S. (eds.) EUROCRYPT 2022. LNCS, vol. 13277, pp. 677–706. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-07082-2_24
16. Ducas, L., Lyubashevsky, V., Prest, T.: Efficient identity-based encryption over NTRU lattices. In: Sarkar, P., Iwata, T. (eds.) ASIACRYPT 2014. LNCS, vol. 8874, pp. 22–41. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-45608-8_2
17. Ducas, L., van Woerden, W.: NTRU fatigue: how stretched is overstretched? In: Tibouchi, M., Wang, H. (eds.) ASIACRYPT 2021. LNCS, vol. 13093, pp. 3–32. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-92068-5_1
18. Duman, J., Hövelmanns, K., Kiltz, E., Lyubashevsky, V., Seiler, G., Unruh, D.: A thorough treatment of highly-efficient NTRU instantiations. Cryptology ePrint Archive, Paper 2021/1352 (2021)
19. Fouque, P.A., et al.: Falcon: fast-Fourier lattice-based compact signatures over NTRU (2016)
20. Fouque, P.A., Kirchner, P., Pornin, T., Yu, Y.: BAT: small and fast kem over NTRU lattices. IACR Trans. Cryptograph. Hardw. Embed. Syst. **2022**(2), 240–265 (2022)
21. Fujisaki, E., Okamoto, T.: Secure integration of asymmetric and symmetric encryption schemes. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 537–554. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-48405-1_34
22. Gama, N., Nguyen, P.Q.: New chosen-ciphertext attacks on NTRU. In: Okamoto, T., Wang, X. (eds.) PKC 2007. LNCS, vol. 4450, pp. 89–106. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-71677-8_7
23. Hermans, J., Vercauteren, F., Preneel, B.: Speed records for NTRU. In: Pieprzyk, J. (ed.) CT-RSA 2010. LNCS, vol. 5985, pp. 73–88. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-11925-5_6
24. Hoffstein, J., Pipher, J., Silverman, J.H.: NTRU: a ring-based public key cryptosystem. In: Buhler, J.P. (ed.) ANTS 1998. LNCS, vol. 1423, pp. 267–288. Springer, Heidelberg (1998). https://doi.org/10.1007/BFb0054868
25. Hoffstein, J., Silverman, J.H.: Optimizations for NTRU. In: Buhler, J.P. (ed.) Proceedings of the Conference on Public Key Cryptography and Computational Number Theory, pp. 77–88. Springer, Cham (2000)

26. Hofheinz, D., Hövelmanns, K., Kiltz, E.: A modular analysis of the Fujisaki-Okamoto transformation. In: Kalai, Y., Reyzin, L. (eds.) TCC 2017. LNCS, vol. 10677, pp. 341–371. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-70500-2_12
27. Howgrave-Graham, N., et al.: The impact of decryption failures on the security of NTRU encryption. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 226–246. Springer, Heidelberg (2003). https://doi.org/10.1007/978-3-540-45146-4_14
28. Jiang, H., Zhang, Z., Chen, L., Wang, H., Ma, Z.: IND-CCA-secure key encapsulation mechanism in the quantum random oracle model, revisited. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018. LNCS, vol. 10993, pp. 96–125. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-96878-0_4
29. Kirchner, P., Fouque, P.-A.: Revisiting lattice attacks on overstretched NTRU parameters. In: Coron, J.-S., Nielsen, J.B. (eds.) EUROCRYPT 2017. LNCS, vol. 10210, pp. 3–26. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-56620-7_1
30. Lindner, R., Peikert, C.: Better key sizes (and attacks) for LWE-based encryption. In: Kiayias, A. (ed.) CT-RSA 2011. LNCS, vol. 6558, pp. 319–339. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-19074-2_21
31. López-Alt, A., Tromer, E., Vaikuntanathan, V.: On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption. In: STOC '12, pp. 1219–1234 (2012)
32. Lyubashevsky, V., Peikert, C., Regev, O.: On ideal lattices and learning with errors over rings. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 1–23. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13190-5_1
33. Lyubashevsky, V., Seiler, G.: NTTRU: truly fast NTRU using NTT. Cryptology ePrint Archive, Paper 2019/040 (2019)
34. Micciancio, D., Schultz, M.: Error correction and ciphertext quantization in lattice cryptography. In: Handschuh, H., Lysyanskaya, A. (eds.) CRYPTO 2023. LNCS, vol. 14085, pp. 648–681. Springer Nature Switzerland, Cham (2023). https://doi.org/10.1007/978-3-031-38554-4_21
35. Nguyen, P.Q., Pointcheval, D.: Analysis and improvements of NTRU encryption paddings. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 210–225. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45708-9_14
36. NIST: Post-Quantum Cryptography Standardization. http://csrc.nist.gov/groups/ST/post-quantum-crypto/submission-requirements/index.html
37. NIST: Status report on the second round of the NIST post-quantum cryptography standardization process (2020). https://doi.org/10.6028/NIST.IR.8309
38. NIST: Status report on the third round of the NIST post-quantum cryptography standardization process (2022). https://doi.org/10.6028/NIST.IR.8413-upd1
39. Peikert, C.: An efficient and parallel gaussian sampler for lattices. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 80–97. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-14623-7_5
40. Pellet-Mary, A., Stehlé, D.: On the hardness of the NTRU problem. In: Tibouchi, M., Wang, H. (eds.) ASIACRYPT 2021. LNCS, vol. 13090, pp. 3–35. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-92062-3_1
41. Pöppelmann, T., Güneysu, T.: Towards practical lattice-based public-key encryption on reconfigurable hardware. In: Lange, T., Lauter, K., Lisoněk, P. (eds.) SAC 2013. LNCS, vol. 8282, pp. 68–85. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-43414-7_4

42. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: STOC '05, pp. 84–93. ACM (2005)
43. Stehlé, D., Steinfeld, R.: Making NTRU as secure as worst-case problems over ideal lattices. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 27–47. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-20465-4_4
44. Stehlé, D., Steinfeld, R., Tanaka, K., Xagawa, K.: Efficient public key encryption based on ideal lattices. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 617–635. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-10366-7_36
45. Zhang, J., Yu, Yu., Fan, S., Zhang, Z., Yang, K.: Tweaking the asymmetry of asymmetric-key cryptography on lattices: KEMs and signatures of smaller sizes. In: Kiayias, A., Kohlweiss, M., Wallden, P., Zikas, V. (eds.) PKC 2020. LNCS, vol. 12111, pp. 37–65. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-45388-6_2
46. Zhu, Y., Liu, Z., Pan, Y.: When NTT meets Karatsuba: preprocess-then-NTT technique revisited. In: Gao, D., Li, Q., Guan, X., Liao, X. (eds.) ICICS 2021. LNCS, vol. 12919, pp. 249–264. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-88052-1_15