Notes on NTRUsign

Pritam Chandra

1 Describing the lattice $M_{h,q}$ in NTRU

For $h \in R = \frac{\mathbb{Z}[x]}{x^N - 1}$ and $q \in \mathbb{Z}$, the lattice

$$M_{h,q} := \{(s,t) \in R^2 \,|\, t \equiv s\,h \bmod q\} \tag{1}$$

Therefore, if $(s,t) \in M_{h,q}$, then $t = s\,h + q\,k$ for some $k \in R$. Or,

$$(s,t) = s(1,h) + k(0,q).$$

Now, $(1,h)$ cannot be a multiple of $(0,q)$, hence the two are linearly independent. Hence, $M_{h,q}$ is a R-lattice of rank 2. Precisely,

$$M_{h,q} = \text{span}\{\mathcal{A}\}, \quad \text{where the basis } \mathcal{A} = \{(1,h),(0,q)\} \tag{2}$$

2 Obtaining another basis of $M_{h,q}$

We start with an arbitrary point (f,g) in $M_{h,q}$. In practice f, g are chosen first with certain restrictions that make f, g likely to be invertible in R. Then h is set as $h = f^{-1}g \bmod q$.

We calculate the resultant of f and g with $x^N - 1$ and denote them as,

$$R_f = \text{res}(f, x^N - 1), \quad R_g = \text{res}(g, x^N - 1)$$

The property of invertibility ensures that $R_f, R_g \neq 0$. Moreover, using the property of resultants we can find polynomials ρ_f, k_f and ρ_g, k_g such that

$$\rho_f f + k_f(x^N - 1) = R_f \tag{3}$$
$$\rho_g g + k_g(x^N - 1) = R_g \tag{4}$$

It is also likely that R_f and R_g are coprime (otherwise we restart with a different pair). Thus we can find integers α and β such that

$$\alpha\,R_f + \beta R_g = 1 \tag{5}$$

Using (3), (4) and (5), we have

$$q\alpha\,[\rho_f f + k_f(x^N - 1)] + q\beta\,[\rho_g g + k_g(x^N - 1)] = q$$

We reduce the above equation in R as

$$q\alpha\rho_f f + q\beta\rho_g g = q$$

Substituting $q\alpha\rho_f = F$ and $q\beta\rho_g = -G$, we have

$$fF - gG = q \neq 0 \tag{6}$$

This ensures that (F, G) is linearly independent to (f, g) in R, as the determinant of the matrix whose rows are these two vectors is non-zero.

Moreover, as $q|q(\alpha\rho_f h + \beta\rho_g)$, we have $-q\beta\rho_g \equiv q\alpha\rho_f \cdot h \bmod q$, or,

$$G \equiv Fh \bmod q, \quad \text{Or,} \quad (F, G) \in M_{h,q}$$

And, $\det\begin{bmatrix} f & g \\ F & G \end{bmatrix} = q = \det\begin{bmatrix} 1 & h \\ 0 & q \end{bmatrix}$. Hence $\{(f, g), (F, G)\}$ is indeed a basis of $M_{h,q}$. Or,

$$M_{h,q} = \text{span}\{\mathcal{B}\}, \quad \text{where the basis } \mathcal{B} = \{(f, g), (F, G)\}$$

In practice, there exists algorithms to calculate $\rho_f, \rho_g, \alpha, \beta$ which are used to compute F and G.

3 Making a basis "good"

In our context a basis is "good" if it is fairly orthogonal and if its comprising elements have small coefficients. Fortunately, given any basis $\mathcal{B}$, the following process can transform $\mathcal{B}$ to simultaneously improve in both the desired properties.

As discussed in **2**, f, g are chosen first, and are chosen to be binary polynomials. Thus they already has small coefficients. So, to make the (F, G) obtained in (6) to be fairly orthogonal to (f, g) along with them having small coeffcients we substract the largest possible multiple of (f, g) from (F, G). This is similar to Gram-Schmidt orthogonalisation, except with the constraint of a result with small coefficients.

As reducing the first coordinate automatically reduces the second, we focus on finding the largest $k \in R$ such that $\|F - kf\|$ is small. To ensure $k \in R$, we choose

$$k = \lfloor Ff^{-1} \rceil \tag{7}$$

Thus the new basis obtained is $\{(f, g), (F, f) - k(f, g)\}$. (Note, usually $Ff^{-1} \in R$ as f is invertible in R.)

4 Recovering t from s for $(s,t) \in M_{h,q}$

Given $(s,t) \in M_{h,q}$, by definition of the lattice there exists $k \in R$ such that $t = sh + qk$.

If (s,t) is a solution to ApprCVP to u, then in practise it is sufficient to obtain its first coordinate s. The second coordinate t can be recovered by finding a polynomial k that minimises the distance of (s,t) from u. This is discussed further in ✠[3].

However, in mod q universe, the second coordinate t of w is unambiguous, and is simply given by $t = sh \bmod q$. For brevity we write $t = sh$ in the discussion coming in **6**.

5 Solving ApprCVP in $M_{h,q}$

Let us first consider points in R^2 of the form $(0,m)$. Such a point can be represented with the coordinates (x,y) in the basis $\mathcal{B}$, where,

$$(x,y) = (0,m)\begin{bmatrix} f & g \\ F & G \end{bmatrix}^{-1} = (0,m)\,\frac{1}{q}\begin{bmatrix} G & -g \\ -F & f \end{bmatrix} = \left(-\frac{mF}{q}, \frac{mf}{q}\right)$$

Therefore, assuming the chosen basis is fairly orthogonal, using Babai's algorithm a solution of ApprCVP to $(0,m)$ in the lattice $M_{h,q}$ is,

$$\left(\left\lfloor -\frac{mF}{q} \right\rceil, \left\lfloor \frac{mf}{q} \right\rceil\right)\begin{bmatrix} f & g \\ F & G \end{bmatrix} \tag{8}$$

Given any arbitrary point $u = (s,t) \in R^2$, we will call the point

$$p = (0, t - sh) \tag{9}$$

as the projection of the point u in $M_{h,q}$.

Using (8) we can solve ApprCVP for any projected point, as they have the required form. Let $u_1 = (s_1, t_1) \in M_{h,q}$ be a solution of ApprCVP to p.

Note $u - p = (s, sh) \in M_{h,q}$. So, $u_1 + (u - p)$ is also a lattice point. Moreover, $\|(u_1 + u - p) - u\| = \|u_1 - p\|$ is small as u_1 is close to p.

Therefore, $u_1 + (u - p) = (s_1 + s, t_1 + sh)$ is a solution of ApprCVP to u.

✠[1] In particular, in context of the discussion in **4**, if s_1 is the first coordinate of a solution of ApprCVP to the projection of (s,t), then $s_1 + s$ is the first coordinate of a solution of ApprCVP to (s,t).

6 ApprCVP in multiple lattices using "Perturbation"

We do all calculations here in mod q *universe.*

Idea.

Our aim is to find a point (s, sh_o) in the lattice $M_{h_0,q}$ close to a given initial point $(0, m)$. But we do not do this directly, and instead do it in steps. We first find our first point close to $(0, m)$ in $M_{h_B,q}$. Next, we find a second point close to the first point in the lattice $M_{h_{B-1},q}$. We continue the process until we find the $B+1$-st point in the lattice $M_{h_0,q}$. We call this process perturbation.

Algorithm.

Let u_B be a solution to ApprCVP to a given point $(0, m)$ in $M_{h_B,q}$.

For each $0 \leq j \leq B-1$, we want to find a lattice point u_j in the lattice $M_{h_j,q}$ close to the point $u_{j+1} \in M_{h_{j+1},q}$. We do this by replacing the roles of h, u, p, u_1 in **5** by h_j, u_{j+1}, p_j, u_j. We include the discussion in **1** by calculating only the first coordinates of the vectors.

More precisely, starting with $j = B-1$,

1. Let s be the first coordinate of u_{j+1}. The point u_{j+1} is typically a non-lattice point for the lattice $M_{h_j,q}$. However, $u_{j+1} \in M_{h_{j+1},q}$ implies $u_{j+1} = (s, sh_{j+1})$.

   From here using (9), we find p_j, the projection of u_{j+1} in $M_{h_j,q}$.

   $$p_j = (0, s[h_{j+1} - h_j]) \tag{10}$$

2. Applying (8) we obtain the first coordinate s_j of a point in $M_{h_j,q}$ close to p_j.

   Then from ✠[1], $s + s_j$ is the first coordinate of a point u_j in $M_{h_j,q}$ close to u_{j+1}. If $j > 0$, we go to step one with $j = j-1$.

Result.

The above process yields a point u_0 in $M_{h_o,q}$ whose first coordinate is $s = s_0 + s_1 + \ldots + s_B$. Being in the lattice $M_{h_0,q}$, the point u_0 is expressed as (s, sh_0). The distance between u_0 and the starting point $(0, m)$ in mod q universe is given by

$$\|(0, m) - (s, sh_0) \bmod q\|. \tag{11}$$

We expect this distance to be smaller than a decided threshold.

7 The NTRUsign Algorithm

Parameters. $N, q, d_f, d_g, B \in \mathbb{Z}_+, \quad \mathcal{N} \in \mathbb{R}_+$

Key Generation. For each $0 \leq j \leq B$, do the following.

1. Choose binary polynomials f_i and g_i with d_f and d_g ones respectively. This conditions make it likely for f and g to be invetible. Set $h_i = g_i f_i^{-1} \bmod q$. It is observed that given such an f_i the coefficients of f^{-1} seem uniformly distributed mod q, hence h_i appears like an arbitrary polynomial. Thus it is suitable used as a public key.

2. Use **2** to complete each basis $\mathcal{B}_i$ by calculating (F_i, G_i), and employ necessary reductions discussed in **3**. The "goodness" of the bases $\mathcal{B}_i$ is tested by whether a successful signature is produced.

3. Store $\{f_i, F_i, h_i\}_{i \in \{0, \ldots, B\}}$ as the private key, and publish h_0 as the public key.

Signing. Let $r = 0$, r is used to generate a new Hash for the document in case the signature fails. Hash the document $\mathcal{D}$ to the polynomial $m_0 = H(\mathcal{D}||r)$.

Initialize $s = 0, m = m_0, j = B$. While $j \geqslant 0$,

1. $s_j = \left(\left\lfloor -\frac{m F_j}{q} \right\rceil, \left\lfloor \frac{m f_j}{q} \right\rceil \right) \begin{bmatrix} f_j \\ F_j \end{bmatrix}$, compute the first coordinate of the closest point in the j-th lattice, refer (8).

2. $m = s_j(h_j - h_{j-1}) \bmod q$, update the second coordinate of the projection in the j-th lattice, refer (10)[1]

3. Set $s = s + s_j$, retrieve the difference in first coordinate from the original (unprojected) point, refer ✠[1]. Go to step 1 with $j = j - 1$.

Thus we expect that the vector in $M_{h_0, q}$ with the first coordinate s is close to the initial point $(0, m_0)$. That is, we check that the distance $\|(0, m_0) - (s, sh_0) \bmod q\| < \mathcal{N}$, the norm bound. If this holds, we publish the pair (r, s) as the signature on the document. Otherwise the bases were not "good" enough, if the obtained vector is not close enough. Then we increment r and start over with a new hash m_1.

1. this is slightly different than the algorithm described in **6**. In **6** s is used in calculating p_j, while **7** uses s_j. For the sake of clarity in **6**, the point close to the projection was lifted up to ensure its closeness to the original point. However, **7** continues directly after obtaining a close point to the projection, without lifting up. I don't think this is a right approach because the projection of a close point can be far from the original point, thereby incresing the final norm.

Verification. Given a signed document $(\mathcal{D}, r, s)$ we first hash it to the point $m = H(\mathcal{D}||r)$ and subsequenty to the point $(0, m) \in M_{h_0,q}$. We then check if s is indeed the first coordinate of a point close enough to hashed point $(0, m)$, i.e. whether

$$\|(0, m) - (s, sh_0) \bmod q\| < \mathcal{N}.$$

8 Why NTRUsign works!

The basis $\mathcal{A}$ is a bad basis of the lattice $M_{h,q}$, whereas the basis $\mathcal{B}$ is a good one. One of the factor that determines the quality of these bases is that the vector h is large (with large coefficients), whereas f and g are small.

The success of the signature scheme is based on the following underlying idea. Given a lattice desrcibed by the (large) public key h and the public parameter q, it is sufficiently hard to solve ApprCVP in it without knowing a good basis. Further, it is sufficiently to hard to recover short vectors (f, g) in the good basis $\mathcal{B}$, from the public key and parameters. This would require sovling SVP in the lattice $M_{h,q}$ which is a well-recognized hard problem. This is further helped by the property of the coefficients of h to be seemingly random.

Therefore, only the signer, with access to the good basis $\mathcal{B}$, can produce a point (s, sh) close to a given point in R^2. However, even with a bad basis it is easy to verify whether a lattice point is indeed close to a given point in R^2. As (11) only involves s and h.

The perturbation technique is to prevent an attack called "transcript leakage" which is not discussed in these notes.

✠[2] Given a point $(s, t) \in R^2$, the norm function mod q is defined as follows.

$$\|(s \bmod q, t \bmod q)\| = \min_{k_1, k_2 \in R} \|(s - k_1 q, t - k_2 q)\|_2$$

where $\|\cdot\|_2$ is the euclidean norm.

✠[3] Continuing from **4**, using the following information

$$\begin{gathered}(s, t) \text{ is represented in the basis } \mathcal{B} \text{ by the coordinates } (x, y) \\ g = fh + qr, \quad \text{for some } r \in R \\ fG - gF = q, \quad f \text{ invertible}\end{gathered}$$

$t = sh + qk$ can be recovered by substituting $k = ryFf^{-1} + yf^{-1} + xr$.

This can be verified using

$$(s, t) = (x, y) \begin{bmatrix} f & g \\ F & G \end{bmatrix}$$

and,

$$\begin{aligned} sh+qk &= (xf+yF)h+q(ryFf^{-1}+yf^{-1}+xr) \\ &= xg-qxr+yF\,(g-qr)f^{-1}+q\,(ryFf^{-1}+yf^{-1}+xr) \\ &= xg-qxr+yFgf^{-1}-qryFf^{-1}+qryFf^{-1}+qyf^{-1}+qxr \\ &= xg+y(Fg+q\,)f^{-1} \\ &= xg+yG=t \end{aligned}$$

9 An illustration of perturbation