Notes: - The norm bound β should satisfy $\beta < q$. Otherwise, a trivial solution is to set $x = (q, 0, 0, \ldots, 0)^T$.

- We need to choose m, β to be large enough so that a solution does exist.

↳ When $m = \Omega(n \log q)$ and $\beta > \sqrt{m}$ a solution always exists. In particular, when $m \geq \lceil n \log q \rceil$, there always exists $x \in \{-1, 0, 1\}^m$ such that $Ax = 0$:

- There are $2^m \geq 2^{n \log q} = q^n$ vectors $y \in \{0,1\}^m$
- Since $Ay \in \mathbb{Z}_q^n$, there are at most q^n possible outputs of Ay

} By a counting argument, there exist $y_1 \neq y_2 \in \{0,1\}^m$ such that $Ay_1 = Ay_2$

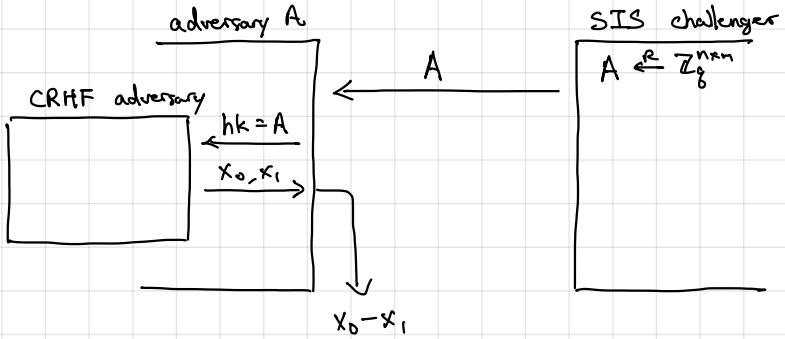
- Thus, if we set $x = y_1 - y_2 \in \{-1, 0, 1\}^m$, then $Ax = A(y_1 - y_2) = Ay_1 - Ay_2 = 0 \in \mathbb{Z}_q^n$ and $\|y_1 - y_2\| \leq \sqrt{m}$

We can directly appeal to SIS to obtain a CRHF: $H : \mathbb{Z}_q^{n \times m} \times \{0,1\}^m \to \mathbb{Z}_q^n$ where we set $m > \lceil n \log q \rceil$.

$H(A, x) := Ax \bmod q$ — view A as hash key and x as input

In this case, domain has size $2^m > 2^{n \log q} = q^n$, which is the size of the output space. Collision resistance follows assuming $\text{SIS}_{n,m,q,\beta}$ for any $\beta \geq 1$

Collision resistance follows via SIS (with parameters n, m, q, β):



Hash key distributed as required by real scheme.

If $x_0 \neq x_1$ and $H(A, x_0) = H(A, x_1)$, then

$$Ax_0 = Ax_1 \bmod q$$

which means

$$A(x_0 - x_1) = 0 \bmod q$$

Since $x_0 \neq x_1$, $x_0 - x_1 \neq 0$ so this is a valid SIS solution where $\|x_0 - x_1\| = 1$.

The SIS hash function supports efficient local updates:

Suppose you have a public hash $h = H(x)$ of a bit-string $x \in \{0,1\}^m$. Later, you want to update $x \mapsto x'$ where x and x' only differ on a few indices (e.g., updating an entry in an address book). For instance, suppose x and x' differ only on the first bit (e.g., $x_1 = 0$ and $x_1' = 1$). Then observe the following

$$h = H(k, x) = A \cdot x = \begin{pmatrix} | & | & & | \\ a_1 & a_2 & \cdots & a_m \\ | & | & & | \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{pmatrix} = \sum_{i \in [m]} x_i a_i = \sum_{i=2}^{m} x_i a_i \quad \text{since } x_1 = 0$$

$$h' = H(k, x') = A \cdot x' = \sum_{i \in [m]} x_i' a_i = x_1' a_1 + \sum_{i=2}^{m} x_i' a_i = a_1 + \sum_{i=2}^{m} x_i' a_i = a_1 + h \quad \text{since } x_i' = x_i \text{ for all } i \geq 2$$

Thus, we can easily update h to h' by just adding to it the first column of A without (re)computing the full hash function.

We will now show how to construct digital signatures from SIS in the random oracle model.

We first introduce the inhomogeneous SIS (ISIS) problem.

Inhomogeneous SIS: The inhomogeneous SIS problem is defined with respect to lattice parameters n, m, q and a norm bound β. The $\text{ISIS}_{n,m,q,\beta}$ problem says that for $A \xleftarrow{R} \mathbb{Z}_q^{n \times m}$, $u \xleftarrow{R} \mathbb{Z}_q^n$, no efficient adversary can find a non-zero vector $x \in \mathbb{Z}^m$ where $Ax = u \in \mathbb{Z}_q^n$ and $\|x\| \leq \beta$

For many choices of parameters, hardness of SIS $\Rightarrow$ hardness of inhomogeneous SIS

The SIS and ISIS problems can be leveraged to construct lattice trapdoors. We define the syntax here:

- $\text{TrapGen}(n, m, q, \beta) \to (A, td_A)$: On input the lattice parameters n, m, q, the trapdoor-generation algorithm outputs a matrix $A \in \mathbb{Z}_q^{n \times m}$ and a trapdoor td_A
- $f_A(x) \to y$: On input $x \in \mathbb{Z}_q^m$, computes $y = Ax \in \mathbb{Z}_q^n$
- $f_A^{-1}(td_A, y) \to x$: On input the trapdoor td_A and an element $y \in \mathbb{Z}_q^n$, the inversion algorithm outputs a value $\|x\| \leq \beta$

Moreover, for a suitable choice of n, m, q, β, these algorithms satisfy the following properties:

- For all $y \in \mathbb{Z}_q^n$, $f_A^{-1}(td_A, y)$ outputs $x \in \mathbb{Z}_q^n$ such that $\|x\| \leq \beta$ and $Ax = y$
- The matrix A output by TrapGen is statistically close to uniform over $\mathbb{Z}_q^{n \times m}$

Lattice trapdoors have received significant amount of study and we will not have time to study it extensively. Here, we will describe the high-level idea behind a very useful and versatile trapdoor known as a "gadget" trapdoor

First, we define the "gadget" matrix (there are actually many possible gadget matrices — here, we use a common one sometimes called the "powers-of-two" matrix):

$$G = \begin{pmatrix} 1 & 2 & 4 & 8 & \cdots & 2^{\lceil \log q \rceil - 1} & & & & & & \\ & & & & & & 1 & 2 & 4 & \cdots & 2^{\lceil \log q \rceil - 1} & \\ & & & & & & & & & \ddots & & \\ & & & & & & & & & & 1 & 2 & 4 & \cdots & 2^{\lceil \log q \rceil - 1} \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & 2 & 4 & \cdots & 2^{\lceil \log q \rceil - 1} \end{pmatrix}}_{g^T} \otimes I_n = g^T \otimes I_n$$

Each row of G consists of the powers of two (up to $2^{\lceil \log q \rceil - 1}$). Thus, $G \in \mathbb{Z}_q^{n \times n\lceil \log q \rceil}$. Oftentimes, we will just write $G \in \mathbb{Z}_q^{n \times m}$ where $m > n\lceil \log q \rceil$. Note that we can always pad G with all-zero columns to obtain the desired dimension.

Observation: given any $y \in \mathbb{Z}_q^n$, it is easy to find an $x \in \{0,1\}^m$ where $Gx = y$.

Let $y_{i,\lceil \log q \rceil - 1}, \ldots, y_{i,0}$ be the binary decomposition of y_i (the i^{th} component of y). Then,

$$G \cdot \begin{pmatrix} y_{1,0} \\ y_{1,2} \\ \vdots \\ y_{1,\lceil \log q \rceil - 1} \\ y_{2,0} \\ \vdots \\ y_{2,\lceil \log q \rceil - 1} \\ \vdots \\ y_{n,0} \\ \vdots \\ y_{n,\lceil \log q \rceil - 1} \end{pmatrix} = \begin{pmatrix} \sum_{j=0}^{\lceil \log q \rceil - 1} 2^j y_{1,j} \\ \vdots \\ \sum_{j=0}^{\lceil \log q \rceil - 1} 2^j y_{n,j} \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} = y$$

↑ Observe that this is a 0/1 vector (binary valued vector)

We will denote this "bit-decomposition" operation by the function $G^{-1}: \mathbb{Z}_q^n \to \{0,1\}^m$

↑ important: G^{-1} is not a matrix (even though G is)!

Then, for all $y \in \mathbb{Z}_q^n$, $G \cdot G^{-1}(y) = y$ and $\|G^{-1}(y)\| = 1$.

↑ ℓ_∞-norm (max absolute value of component of the vector)

Implication: SIS is easy with respect to G:

$$G \cdot \begin{pmatrix} 2 \\ -1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} = 0 \in \mathbb{Z}_q^n \quad \Rightarrow \text{ norm of this vector is } 2$$

Inhomogeneous SIS is also easy with respect to G: take any target vector $y \in \mathbb{Z}_q^n$ and output $G^{-1}(y) \in \{0,1\}^m$.

We now have a matrix with a "public" trapdoor. To construct a secret trapdoor function (useful for cryptographic applications), we will "hide" the gadget matrix in the matrix A, and the trapdoor will be a "short" matrix (i.e., matrix with small entries) that recovers the gadget.

More precisely, a gadget trapdoor for a matrix $A \in \mathbb{Z}_q^{n \times k}$ is a short matrix $R \in \mathbb{Z}_q^{k \times m}$ such that

$$A \cdot R = G \in \mathbb{Z}_q^{n \times m}$$

We say that R is "short" if all values are small. [We will write $\|R\|$ to refer to the largest value in R.]

Suppose we know $R \in \mathbb{Z}_q^{m \times m}$ such that $AR = G$. We can then define the inversion algorithm as follows:

- $f_A^{-1}(td_A = R, y \in \mathbb{Z}_q^n)$: Output $x = R \cdot G^{-1}(y)$.

We check two properties.

1. $Ax = AR \cdot G^{-1}(y) = G \cdot G^{-1}(y) = y$ so x is indeed a valid pre-image
2. $\|x\| = \|R \cdot G^{-1}(y)\| \leq m \cdot \|R\| \|G^{-1}(y)\| = m \cdot \|R\|$
   Thus, if $\|R\|$ is small, then $\|x\|$ is also small (think of β as a large polynomial in n).
   (Recall we are using ℓ_∞-norm now)

Important note: When using trapdoor functions in a setting where the adversary can see trapdoor evaluations, we actually need to randomize the computation of f_A^{-1}. Otherwise, we leak the trapdoor. (We will revisit this later.)

Remaining question: How do we generate A together with a trapdoor (and so that A is statistically close to uniform)?

Many techniques to do so; we will look at one approach using the LHL:

Sample $\bar{A} \xleftarrow{R} \mathbb{Z}_q^{n \times m}$ and $\bar{R} \xleftarrow{R} \{0,1\}^{m \times m}$.

Set $A = [\bar{A} \mid \bar{A}\bar{R} + G] \in \mathbb{Z}_q^{n \times 2m}$

Output $A \in \mathbb{Z}_q^{n \times 2m}$, $td_A = R = \begin{bmatrix} -\bar{R} \\ I \end{bmatrix} \in \mathbb{Z}_q^{2m \times m}$

First, we have by construction that $AR = -\bar{A}\bar{R} + \bar{A}\bar{R} + G = G$, and moreover $\|R\| = 1$. It suffices to argue that A is statistically close to uniform (without the trapdoor R). This boils down to showing that $\bar{A}\bar{R} + G$ is statistically close to uniform given $\bar{A}$.

To do so, we will use the leftover hash lemma. We state the specific version we need in this course:

Leftover hash lemma: Let $n, m, q \in \mathbb{N}$ where $m \geq n \log q$. Then, the following distributions are statistically indistinguishable.

1. Sample $A \xleftarrow{R} \mathbb{Z}_q^{n \times m}$, $r \xleftarrow{R} \{0,1\}^m$, and output (A, Ar).
2. Sample $A \xleftarrow{R} \mathbb{Z}_q^{n \times m}$, $b \xleftarrow{R} \mathbb{Z}_q^n$, and output (A, b).

Essentially: random subset sum of columns of A is statistically close to uniform.

By LHL applied to each column of $\bar{A}\bar{R}$ (and a hybrid argument), we conclude that $\bar{A}\bar{R}$ is statistically close to uniform

↳ Technically to use LHL, we will need to take $m \geq 2n \log q$ so we also pad G so $\bar{A}\bar{R} + G$ has the right dimensions (e.g., we can always pad G as $[G \mid 0]$ with all-zero blocks → can still solve SIS/ISIS with padded version). Could have also taken $R \in \mathbb{Z}_q^{m \times n \log q}$, but padding will be convenient in the future.