Recall the inhomogeneous SIS problem: given \( A \in \mathbb{Z}_q^{n \times m} \) and \( u \in \mathbb{Z}_q^m \), find \( x \in \mathbb{Z}_q^n \) such that \( Ax = y \) and \( \|x\| \leq \beta \).

It turns out that this can actually be used as a trapdoor function. Namely, there exist efficient algorithms:

\[
\text{TrapGen}(n, m, q, \beta) \rightarrow (A, id_A): \quad \text{On input the lattice parameters } n, m, q, \text{ the trapdoor-generation algorithm outputs a matrix } A \in \mathbb{Z}_q^{n \times m} \text{ and a trapdoor } id_A
\]

\[
f_A(x) \rightarrow y: \quad \text{On input } x \in \mathbb{Z}_q^n, \text{ computes } y = Ax \in \mathbb{Z}_q^m.
\]

\[
f_A^{-1}(id_A, y) \rightarrow x: \quad \text{On input the trapdoor } id_A \text{ and an element } y \in \mathbb{Z}_q^m, \text{ the inversion algorithm outputs a vector } x \text{ such that } \|x\| \leq \beta.
\]

Moreover, for a suitable choice of \( n, m, q, \beta \), these algorithms satisfy the following properties:

1. For all \( y \in \mathbb{Z}_q^m \), \( f_A^{-1}(id_A, y) \) outputs \( x \in \mathbb{Z}_q^n \) such that \( \|x\| \leq \beta \) and \( Ax = y \).
2. 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 sketch 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):

\[
G = \begin{pmatrix}
1 & 2 & 4 & \cdots & 2^{\log_q 28} \\
1 & 2 & 4 & \cdots & 2^{\log_q 124} \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
1 & 2 & 4 & \cdots & 2^{\log_q m}
\end{pmatrix}
\]

Each row of \( G \) consists of the powers of two (up to \( 2^{\log_q 28} \)). Thus, \( G \in \mathbb{Z}_q^{m \times m} \). Oftentimes, we will just write \( G \in \mathbb{Z}_q^{m \times m} \) when \( m > n \log_q 1 \). Note that we can always pad \( G \) with all-zero columns to obtain the desired dimension.

Observation: SIS is easy with respect to \( G \):

\[
G \cdot \begin{pmatrix} \frac{q}{3} \\ \frac{q}{2} \\ \frac{q}{3} \\ 0 \end{pmatrix} = 0 \in \mathbb{Z}_q^n \quad \Rightarrow \quad \text{norm of this vector is } 2
\]

Inhomogeneous SIS is also easy with respect to \( G \): take any target vector \( y \in \mathbb{Z}_q^n \).

Let \( y_1, y_2, \ldots, y_n \) be the binary decomposition of \( y \) (the \( i \)-th component of \( y \)). Then,

\[
G \cdot \begin{pmatrix} y_1 \\ \_1^{\log_q 28} \\ \_1^{\log_q 124} \\ \vdots \\ \_1^{\log_q m} \\ \vdots \\ y_n \\ \_1^{\log_q 28} \\ \_1^{\log_q 124} \\ \vdots \\ \_1^{\log_q m} \\ \_1^{\log_q m} \end{pmatrix} = \begin{pmatrix} y_1 \\ \_1^{\log_q 28} \\ \_1^{\log_q 124} \\ \vdots \\ \_1^{\log_q m} \\ \vdots \\ y_n \\ \_1^{\log_q 28} \\ \_1^{\log_q 124} \\ \vdots \\ \_1^{\log_q m} \\ \_1^{\log_q m} \end{pmatrix} = y
\]

Observe that this is a 0/1 vector (binary valued vector), so the \( L_1 \)-norm is exactly 1.

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

Important: \( G^{-1} \) is not a matrix (even though \( G \) is)!
Then, for all \( y \in \mathbb{Z}_q^k \), \( G \cdot G^{-1}(y) = y \) and \( \|G^{-1}(y)\| = 1 \). Thus, both SIS and inhomogeneous SIS are easy with respect to the matrix \( G \).

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 \( A \in \mathbb{Z}_q^{nxk} \) is a short matrix \( R \in \mathbb{Z}_q^{kn} \) such that

\[
AR = G \in \mathbb{Z}_q^{kn}
\]

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 \( R \in \mathbb{Z}_q^{kn} \) such that \( AR = G \). We can then define the inversion algorithm as follows:

- \( f_A^{-1}(t_A; R, y) \in \mathbb{Z}_q^{kn} \): Output \( x = R \cdot G^{-1}(y) \). Important note: When using trapdoor functions in a setting where the adversary can see trapdoor evaluations, we actually need to randomize the computation of \( f_A \). Otherwise, we leak the trapdoor.

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. \( \|x\| = \|R \cdot G^{-1}(y)\| \leq \|R\| \|G^{-1}(y)\| = \|R\| \|y\| \)

Thus, if \( \|R\| \) is small, then \( \|x\| \) is also small (think of \( y \) as a large polynomial in \( n \)).

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 \( \bar{A} \in \mathbb{Z}_q^{nxm} \) and \( \bar{R} \in \{0,1\}^{mn} \).
- Set \( A = [\bar{A} | \bar{AR} + G] \in \mathbb{Z}_q^{nx2m} \).
- Output \( A \in \mathbb{Z}_q^{nx2m} \), \( t_A = R = [\bar{R} | \bar{I}] \in \mathbb{Z}_q^{2kn} \).

First, we have by construction that \( AR = -\bar{AR} + \bar{AR} + G = G \), 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 \( \bar{AR} + G \) is statistically close to uniform given \( \bar{A} \). We appeal to the LHL:

1. From the previous lecture, the function \( f_A(x) = Ax \) is pairwise independent.
2. Thus, by the LHL, if \( m \geq 3n \log q \), then \( A \bar{x} \) is statistically close to uniform in \( \mathbb{Z}_q^n \) when \( \bar{x} \in \{0,1\}^n \).
3. Claim now follows by a hybrid argument (applied to each column of \( R \)).

Thus, given \( \bar{A} \), the matrix \( \bar{AR} \) is still statistically close to uniform. Correspondingly, \( A \) is statistically close to uniform.

Digital signatures from lattice trapdoors: We can use lattice trapdoors to obtain a digital signature scheme in the random oracle model (this is essentially an analog of RSA signatures):

- \( \text{KeyGen}(1^n) \): \( (A, t_A) \leftarrow \text{TrapGen}(n, m, q, p) \) [lattice parameters \( n, m, q, p \) are based on security parameter \( n \)]
  - Output \( vk = A \) and \( sk = t_A \).
- \( \text{Sign}(sk, m) \): Output \( \sigma \leftarrow f_A^{-1}(t_A; H(m)) \). Here, \( H : \{0,1\}^* \rightarrow \mathbb{Z}_q^n \) is modeled as a random oracle.
- \( \text{Verify}(vk, m, \sigma) \): Check that \( \|\sigma\| \leq p \) and that \( \sigma = H(m) \).

Hardness reduces to hardness of inhomogeneous SIS (similar proof to RSA-FDH). Sketch:

1. Replace \( A \) with a uniformly random matrix (as required by inhomogeneous SIS) — follows by property of TrapGen.
2. Given inhomogeneous SIS challenge \( (A, y) \), set public key to \( A \) and \( H(m^*) = y \) where \( m^* \) is the message the adversary forge on (guess this at beginning).
3. To simulate signing queries on a message m (without knowledge of trapdoor), first sample \( x \sim D_s \) and set \( H(n) = Ax \)

- Here \( D_s \) corresponds to the distribution of vectors output by the preimage-sampling algorithm \( f_\mathcal{A}^s \) [this is typically a discrete Gaussian distribution with standard deviation \( \sigma \), where \( \sigma \) is chosen so that \( Ax \) is statistically close to uniform over \( L(A) \)]

- Thus, by programming the random oracle, we can sign arbitrary messages without knowledge of the trapdoor for \( A \)

Summary so far: The SIS problem can be used to realize many asymmetric primitives such as OWFs, CRHF, and signatures

- Useful trick: "Concealing" a trapdoor (e.g., short matrix/basis) within a random-looking one - cannon theme in lattice-based cryptography.

For public-key primitives, we will rely on a very similar assumption: learning with errors (LWE), which can also be viewed as a "dual" of SIS. We introduce the assumption below:

Learning with Errors (LWE): The LWE problem is defined with respect to lattice parameters \((n,m,r,\sigma)\), where \( r \) is an error distribution over \( \mathbb{Z}_q^m \) (alternatively, this is a discrete Gaussian distribution over \( \mathbb{Z}_q^m \)). The \( \text{LWEn}_m \) assumption states that for a random choice \( A \in \mathbb{Z}_q^{m \times n}, s \in \mathbb{Z}_q^n, e \sim r^m \), the following two distributions are computationally indistinguishable:

\[
(A, s^T A + e) \approx (A, r)
\]

where \( r \sim \mathbb{Z}_q^m \).

In words, the LWE assumption says that noisy linear combinations of a secret vector over \( \mathbb{Z}_q^n \) looks indistinguishable from random.

A few notes/observations on LWE:

- Typically, \( m \) is sufficiently large so that the LWE secret \( s \) is uniquely determined.

- Without the error terms, this problem is easy for \( m > n \): simply use Gaussian elimination to solve for \( s \)

- Observe that if SIS is easy, then LWE is easy. Namely, if the adversary can find a short \( u \in \mathbb{Z}_q^n \) such that \( Au = 0 \), then, the adversary can compute:

\[
(s^T A + e)u = s^T Au + e^T u = e^T u \Rightarrow \|e^T u\| \leq m \cdot \|e\| \cdot \|u\|
\]

- This is small (compared to \( q \))

\( e^T u \) will be uniform over \( \mathbb{Z}_q^m \), one unlikely to be small.

- We can also choose the LWE secret from the error distribution (so \( s \) is short) - can be useful for both efficiency and for functionality (this is at least as hard as LWE with secrets drawn from any distribution, including the uniform one)

- Can also consider search vs. decision versions of the problem (i.e., search LWE says given \( (A, s^T A + e) \), find \( s \)). There are search-to-decision reductions for LWE.

LWE as a lattice problem: The search version of LWE essentially asks one to find \( s \) given \( s^T A + e \). This can be viewed as solving the "bounded-distance decoding" (BDD) problem on the \( q \)-ary lattice

\[
\mathcal{L}(A^T) = \{ s \in \mathbb{Z}_q^n : A^T s \} + \mathbb{Z}_q^n
\]

i.e., given a point that is close to a lattice element \( s \in \mathcal{L}(A^T) \), find the point \( s \)
Connects to worst-case hardness: Regev showed that for any \(m = \text{poly}(n)\) and modulus \(q < 2^m\) and for a discrete Gaussian noise distribution (with values bounded by \(\beta\)), solving \(\text{LWE}_{\text{sym}}\) is as hard as quantitatively solving \(\text{GapSVP}_\gamma\) an arbitrary \(n\)-dimensional lattice with approximation factor \(\gamma = \mathcal{O}(n^{0.5})\).

Long sequence of subsequent works have shown classical reductions to worst-case lattice problems (for suitable instantiations of the parameters).

Symmetric encryption from \(\text{LWE}\) (for binary-valued messages)

\[
\begin{align*}
\text{Setup}(1^n): & \quad \text{Sample } s \leftarrow \mathbb{Z}_q^n. \\
\text{Encrypt}(s, m): & \quad \text{Sample } a \leftarrow \mathbb{Z}_q^n \text{ and } e \leftarrow \mathbb{Z}_q. \quad \text{Output } (a, sa + e \cdot \ell^\top).
\end{align*}
\]

\[
\text{Decrypt}(s, ct): \quad \text{Output } [ct_s - s^\top ct]_z.
\]

**Correctness:** \(ct_s - s^\top ct = sa + e + \mu \cdot [\frac{q}{2}] - s^\top a = \mu \cdot [\frac{q}{2}] + e\)

if \(|e| < \frac{q}{2}\), then decryption recovers the correct bit.

**Security:** By the \(\text{LWE}\) assumption, \((a, sa + e) \approx (a, r)\)

where \(r \approx \mathbb{Z}_q^n\). Thus, \((a, sa + e) \approx (a, r + [\frac{q}{2}]) \approx (a, sa + e + [\frac{q}{2}]).\)

Observe: this encryption scheme is additively homomorphic (over \(\mathbb{Z}_q^n\)):
\[
(a_1, sa_1 + e_1 + \mu_1 \cdot [\frac{q}{2}]) \Rightarrow (a_1 + a_2, s^\top (a_1 + a_2) + (e_1 + e_2) + (\mu_1 + \mu_2) \cdot [\frac{q}{2}]).
\]

This decryption then computes \((\mu_1 + \mu_2) \cdot [\frac{q}{2}] + e_1 + e_2\),

which when rounded yields \(\mu_1 + \mu_2\) (mod \(q\)) provided that \(|e_1 + e_2 + 1| < \frac{q}{4}\).

Using the results from \(\text{LWE}\), we can obtain a public key encryption scheme if we can “refresh” the ciphertexts.

**Idea:** We will rely on the \(\text{LHL}\). We will include encryptions of 0 in the public key and refresh ciphertexts by taking a subset sum of ciphertexts of 0:

\[
\begin{align*}
\text{Setup}(1^n): & \quad A \leftarrow \mathbb{Z}_q^{nm}, \\
& \quad s \leftarrow \mathbb{Z}_q^n, \\
& \quad c \leftarrow \mathbb{Z}_q^n. \\
\end{align*}
\]

\[
\begin{align*}
\text{Encrypt}(pk, m): & \quad \text{Sample } r \leftarrow \mathbb{Z}_q^{m}, \\
& \quad \text{Output } (A, b^\top c + r^\top m \cdot [\frac{q}{2}]).
\end{align*}
\]

\[
\begin{align*}
\text{Decrypt}(sk, ct): & \quad \text{Output } [ct_s^\top s^\top ct]_z.
\end{align*}
\]

**Correctness:** \(ct_s - s^\top ct = b^\top r + \mu \cdot [\frac{q}{2}] - s^\top Ar = s^\top Ar + e^\top r + \mu \cdot [\frac{q}{2}] - s^\top Ar = \mu \cdot [\frac{q}{2}] + e^\top r\)

if \(|e^\top r| < \frac{q}{4}\), then decryption succeeds (since \(e\) is small and \(r\) is binary, \(e^\top r\) is not large: \(|e^\top r| \ll m\|\ell\|m\|\ell| = m\|\ell|^2\).
Security: Follows by LWE and LHL:

Hybp: Real public key
Hyb1: Uniformly random public key (e.g. \( b \in \mathbb{Z}_q^m \))
Hyb2: Uniformly random ciphertext (e.g. \( ct = (u, t) \) where \( u \in \mathbb{Z}_q^m \) and \( t \in \{0, 1\} \))

LHL: \((\tilde{A}, \tilde{A} r) \approx (\tilde{A}, u)\)

where \( \tilde{A} = \left[ \frac{A}{v} \right] \in \mathbb{Z}_b^{m \times n}, \)
\( r \in \{0, 1\}^n, \) and \( u \in \mathbb{Z}_q^m \)

Encrypting multiple bits: May seem wasteful to use a vector to encrypt a single bit. We can consider a simple variant of Regev encryption where we reuse \( A \) to encrypt multiple bits:

Setup(\( n, l^* \)): sample \( A \in \mathbb{Z}_q^{n \times m} \)
\( S \leftarrow \mathbb{Z}_q^{n \times l^*} \)
\( E \leftarrow \chi^{n \times l^*} \)

\( \mathcal{B} \leftarrow S^T A + E \in \mathbb{Z}_q^{2 \times m} \)

pk: \((A, B^T)\)

sk: \(S\)

Encrypt(\( pk, \mu \in \{0, 1\}^l \)): sample \( r \in \{0, 1\}^l \)
output \((A r, B^T r + \mu \cdot 1^l)\)

Decrypt(\( sk, ct \)): output \( L c_t - S^T c_t \).

Correctness: As before: \( c_t - S^T c_t = B^T r + \mu \cdot 1^l - S^T A r = E^T r + \mu \cdot 1^l \)

Security: As before: by LWE, \((A, S^T A + E^T) \approx (A, R)\) where \( A \in \mathbb{Z}_q^{n \times m} \), \( S \in \mathbb{Z}_q^{n \times l^*} \), \( E \in \chi^{n \times l^*} \), \( R \in \mathbb{Z}_b^{m \times m} \)

\(-\) in particular, apply a hybrid argument and argue for each row of \( S \) (and corresponding row of \( S^T A + E^T \))