# Assignment 5

## Q1 (10 points)

Let $c_1, c_2, e \in \mathbb{F}_2^n$. Show that

$$|c_1 - (c_2 + e)|_{Ham} \geq |c_1 - c_2|_{Ham} - |e|_{Ham}$$

## Q2 (15 points)

Suppose we take the McEliece cryptosystem, with a family of $(n, k, d)$-codes (let $t = \frac{d-1}{2}$ and assume it is an integer). To save space, we modify the scheme so that encryption is as follows, parameterized by an integer $\ell \leq t$:

Enc$(\mathbf{PK} = A, m)$: Compute $c = Am$. Compute $c'$ by cutting off the last $\ell$ bits of $c'$. Select a random error $e \in \mathbb{F}_2^{n-\ell}$ of weight $t - \ell$, and output $c'' = c' + e$.

1. (5 points) Give a decryption algorithm that will succeed with probability $1$.
2. (5 points) How much space does this save, as a percentage?
3. (5 points) Estimate the difficulty of attacking this new scheme. Is this more or less secure than the original scheme?
4. (0 points) Do you think this tradeoff is worth it?

## Q4 (15 points)

1. (10 points) Show that the following two worst-case problems are equivalent:

$(n, k)$-Codeword decoding problem: Given a $n \times k$ matrix $G$, and an $n$-dimensional binary vector $c$, find a $k$-dimensional binary vector $m$ such that $Gm - c$ has weight at most $t$ (if it exists).

$(n, k)$-Syndrome decoding problem: Given a $(n - k) \times n$ matrix $H$, and an $n - k$-dimensional binary vector $y$, find an $n$-dimensional binary vector $e$ of weight at most $t$ such that $He = y$, if it exists.

2. (5 points) Show that the following two average-case problems are equivalent:

$(\mathfrak{C}, n, k)$-McEliece decoding problem: Let $A$ be an $n \times k$ matrix $A$ where $A = PGS$, with $G$ sampled uniformly randomly from a family of codes $\mathfrak{C}$, $P$ a random $n \times n$ permutation, and $S$ a random $k \times k$ invertible matrix. Given $A$, and an $n$-dimensional binary vector $c$, find a $k$-dimensional binary vector $m$ such that $Am - c$ has weight at most $t$ (if it exists).

$(\mathfrak{C}, n, k)$-McEliece Syndrome Decoding Problem: Let $H_0$ be an $(n-k) \times k$ matrix such that $[I|H_0] = SHP$, where $H$ is sampled uniformly randomly as a parity check from a family of codes $\mathfrak{C}$, $P$ is a random $n \times n$ permutation (such that the first $n-k$ columns are full rank), and $S$ is such that $SHP$ row reduces the first $n-k$ columns. Given $H_0$ and an $n-k$-dimensional binary vector $y$, find an $n$-dimensional binary vector $e$ of weight at most $t$ such that $[I|H_0]e = y$, if it exists.

## Q5 (10 points)

Someone implemented classic Mceliece, but they forgot to apply the random invertible matrix and the random permutation. Given their parity check matrix $H$, recover the Goppa code they used. Does this break the scheme?

## Q6 (10 points)

McEliece did not use a full FO transform, but more of an ad-hoc CCA security transformation. We can try the same thing with Kyber. Decryption failures are a bit harder to detect in Kyber, since we cannot directly measure the magnitude of the error. Instead, we will send a commitment to the encrypted message, as follows:

KeyGen(): Generate random $A \in R_q^{\ell \times k}$, $s \leftarrow \chi_s$ and $e \leftarrow \chi_e$ as in regular Kyber. Set $\mathbf{PK} = (A, b = As + e)$, and $\mathbf{SK} = s$.

Encaps($\mathbf{PK}$): Select random $r \leftarrow \chi_s$, $e', e'' \leftarrow \chi_e$, and random $m \in \{0,1\}^n$ (where $n$ is the degree of the polynomial ring) and let $m(X)$ be a polynomial with the elements of $K$ as coefficients. Let

$$c = (c_1, c_2), c_1 = r^T A + e'^T, c_2 = r^T b + e'' + m\lfloor \frac{q}{2} \rceil$$

Let $K = H(m, c, 0)$, and return $(c, H(m))$ as the ciphertext and $K$ as the session key.

Decaps($\mathbf{SK}, (c, h)$): Decrypt $c$ as in normal Kyber to get $m$. Check that $H(m)$ equals $h$. If yes: output $K = H(m, c, 0)$; otherwise, output $K = H(\mathbf{SK}, c, 1)$.

Give an efficient IND-CCA attack on this scheme. Contrast to McEliece.

## Q7 (10 points)

Provide an MPC method for squaring a shared secret $[x]$ that uses half as much online communication as the naive method (i.e., using Beaver triples to compute $[x \cdot x]$). "Online" meaning communication done during a computation, rather than pre-computation.