Before you begin, recall the definitions of sampling and query access for vectors and matrices (SQ(v), SQ(A)) and oversampling and query access (SQ_v(\phi(v)), SQ_\phi(A)) [CGLLTW22, Definition 3.2]. Below, time complexities are in the word RAM model: basically, assume that reading input numbers, and performing operations on those numbers, cost \( O(1) \).

**Problem 1** (Errare humanum est...). Suppose we have \( SQ_\phi(u), SQ_\phi(v) \) for vectors \( u, v \). Show that we have \( SQ_\phi(A) \) for \( A := uv^\dagger \) and \( \phi = \phi_u \phi_v \) with cost \( \text{sq}_\phi(A) = \text{sq}_{\phi_u}(u) + \text{sq}_{\phi_v}(v) \).

**Problem 2** (...sed perseverare (non?) diabolicum.). Suppose we are given a matrix \( A \in \mathbb{C}^{m \times m} \) with at most \( s \) non-zero entries per row, and suppose all entries are bounded by \( c \). We are given this matrix as a list of non-zero entries \((i, j, A(i, j))\). Show how to perform \( SQ_\phi(A) \) queries for \( \phi = c^2 \frac{sm}{\|A\|_F} \) with \( \text{sq}_\phi(A) = s \).\(^1\) This means that we can run “dequantized” algorithms on sparse matrices with condition number \( \kappa \); why doesn’t this imply that QSVT admits no exponential speedup for sparse matrices?

**Problem 3** (The alias method [Vos91]). Let \( p = (p_1, \ldots, p_m) \) be a set of probabilities, so \( p_i \geq 0 \) and \( \sum p_i = 1 \). Suppose also that all of the \( p_i \)'s are described in binary with \( O(1) \) bits.

1. Suppose we are given a uniformly random number \( x \in [0,1] \) as a stream of random bits. Show how to sample \( i \in [m] \) such that \( \Pr[\text{sample } i] = p_i \) in \( O(m) \) operations.

2. Suppose we are given \( p = (p_1, \ldots, p_m) \) in the following form: we get a list of \( m \) probability distributions \( d_1, \ldots, d_m \) such that \( \frac{1}{m}(d_1 + \cdots + d_m) = p \) and every \( d_i \) is supported on at most two outcomes. Show that we can sample \( i \in [m] \) according to \( p \) in \( O(1) \) time.

3. Prove that we can convert any distribution \( p \) into the form described above. Prove that we can do this in \( O(m) \) time.\(^2\)

**References**


\(^1\)Hint: We immediately have query access to \( A \). What’s a good upper bound that’s easy to sample from?

\(^2\)This implies that, if we get time to pre-process, we can get a data structure such that we can respond to \( SQ(v) \) queries in \( O(1) \) time (in the word RAM access model).