Sublinear Time and Space Algorithms 2026A – Problem Set 4

Robert Krauthgamer

Due: January 9, 2026

General instructions: Please keep your answers short and easy to read. You can use results, calculations or notation seen in class without repeating them, unless asked explicitly to redo them.

1. We saw in class an ℓ_1 point query of a frequency vector $x \in \mathbb{R}^n$, where on query $i \in [n]$ it outputs $\tilde{x}_i$ such that

   $$\Pr\left[\tilde{x}_i \notin x_i \pm \alpha\|x\|_1\right] \le 1/4.$$

   Explain how to refine the algorithm and/or analysis to achieve smaller error

   $$\Pr\left[\tilde{x}_i \notin x_i \pm \alpha\|x_{-i}\|_1\right] \le 1/4,$$

   where x_{-i} is the vector x but with coordinate i zeroed.

   Assume for simplicity that $x_i \ge 0$ for all $i \in [n]$ (even though it is not necessary).

2. Consider a dynamic geometric stream over $[\Delta]^d$ in the special case $d = 2$. Let $X \subset [\Delta]^2$ be the final set of points (insertions minus deletions), and to make sure it is well-defined, assume that a point can be deleted only if it was inserted earlier.

   Design a streaming algorithm that $(1+\varepsilon)$-approximates the diameter of X.

   Hint: Assume first you are given a guess $D > 0$ that is a 2-approximation for the diameter.

   Remark: We saw in class a very simple 2-approximation algorithm for insertion-only streams; here we allow deletions and want better approximation.

   Remark: As done in class, do not count storage of the algorithm's random coins.

3. Consider a dynamic geometric stream over $[\Delta]^d$ (i.e., a stream of insertions and deletions of points), let $X \subset [\Delta]^d$ be the final set of points, and assume it is well-defined (a point can be deleted only if it was inserted earlier). Assume also that $n := |X| \le \mathrm{poly}(\Delta^d)$. The cost of clustering X to a center point $c \in [\Delta]^d$ is defined as

   $$\mathrm{cost}_c(X) = \sum_{x \in X} \|x - c\|_2.$$

   Design a streaming algorithm that uses storage of $(d \log \Delta)^{O(1)}$ bits, and reports an $O(d \log \Delta)$-approximation to the cost of an optimal center, given by

   $$\mathrm{cost}^*(X) := \min_{c \in [\Delta]^d} \mathrm{cost}_c(X).$$

Hint: Build a quadtree and at each tree level use an ℓ_1 point query, or its refinement from Question 1 above. Start with an easier task: given a query point $c \in [\Delta]^d$ at the end of the stream, report an $O(d \log \Delta)$-approximation of $\text{cost}_c(X)$. Next, to estimate the cost of an optimal center, find at each tree level if some quadtree cell contains at least $n/2$ points.

4. In the sliding-window model (not seen in class), the input is an insertion-only stream σ of items from domain $[n]$, and there is also a parameter $w \leq \text{poly}(n)$ known in advance. At each time t, the *active window* is the list of w most recent items, defined as $W_t = (\sigma_{t-w+1}, \ldots, \sigma_t)$. The challenge is to avoid storing the entire active window, e.g., deleting each item that leaves the window requires storing the entire window.

   Design an algorithm that uses storage of $\text{polylog}(n)$ bits, and can report, at any desired time t, a uniformly random sample from the distinct elements of the window W_t.

   Remark: A sample is requested only once, at time t not known to the algorithm.

   Hint: Assign to each item a random "priority", using a random hash function $h : [n] \to [0, 1]$. Start with the special case where every window W_t contains w distinct items, and analyze the expected storage requirement of your algorithm.