1 Frequency Moments and the AMS algorithm

\(\ell_p\)-norm problem: Let \(x \in \mathbb{R}^n\) be the frequency vector of the input stream, and fix a parameter \(p > 0\).

Goal: estimate its \(\ell_p\)-norm \(\|x\|_p = (\sum_i |x_i|^p)^{1/p}\). We focus on \(p = 2\).

Theorem 1 [Alon, Matthias, and Szegedy, 1996]: One can estimate the \(\ell_2\) norm of a frequency vector \(x \in \mathbb{R}^n\) within factor \(1 + \varepsilon\) [with high constant probability] using storage requirement of \(s = O(\varepsilon^{-2})\) words. In fact, the algorithm uses a linear sketch of dimension \(s\).

Algorithm AMS (also known as Tug-of-War):

1. Init: choose \(r_1, \ldots, r_n\) independently at random from \((-1, +1)\)
2. Update: maintain \(Z = \sum_i r_i x_i\)
3. Output: to estimate \(\|x\|_2^2\) report \(Z^2\)

The sketch \(Z\) is linear, hence can be updated easily.

Storage requirement: \(O(\log(nm))\) bits, not including randomness; we will discuss implementation issues a bit later.

Analysis: We saw in class that \(\mathbb{E}[Z^2] = \sum_i x_i^2 = \|x\|_2^2\), and \(\text{Var}(Z^2) \leq 2(\mathbb{E}[Z^2])^2\).

Algorithm AMS+:

1. Run \(t = O(1/\varepsilon^2)\) independent copies of Algorithm AMS, denoting their \(Z\) values by \(Z_1, \ldots, Z_t\), and output the mean of these copies \(\bar{Y} = \frac{1}{t} \sum_j Z_j^2\).

Observe that the sketch \((Z_1, \ldots, Z_t)\) is still linear.

Storage requirement: \(O(t) = O(1/\varepsilon^2)\) words (for constant success probability), not including randomness.

*These notes summarize the material covered in class, usually skipping proofs, details, examples and so forth, and possibly adding some remarks, or pointers. The exercises are for self-practice and need not be handed in. In the interest of brevity, most references and credits were omitted.*
Analysis: We saw in class that
\[ \Pr[|\tilde{Y} - \mathbb{E}\tilde{Y}| \geq \varepsilon \mathbb{E}\tilde{Y}] \leq \frac{\text{Var}(\tilde{Y})}{\varepsilon^2(\mathbb{E}\tilde{Y})^2} = \frac{\text{Var}(Z^2)/t}{\varepsilon^2(\mathbb{E}Z^2)^2} \leq \frac{2}{t\varepsilon^2}. \]

Choosing appropriate \( t = O(1/\varepsilon^2) \) makes the probability of error an arbitrarily small constant.

Notice it actually gives a \((1 \pm \varepsilon)\)-approximation to \( \|x\|_2^2 \), which is immediately yields a \((1 \pm \varepsilon)\)-approximation to \( \|x\|_2 \).

**Exer:** What would happen in the accuracy analysis if the \( r_i \)'s were chosen as standard gaussians \( N(0, 1) \)?

2 \( \ell_1 \) Point Query via CountMin

**Problem Definition:** Let \( x \in \mathbb{R}^n \) be the frequency vector of the input stream, and let \( \|x\|_p = (\sum |x_i|^p)^{1/p} \) be its \( \ell_p \)-norm. Let \( \alpha \in (0, 1) \) and \( p \geq 1 \) be parameters known in advance.

The goal is to estimate every coordinate with additive error, namely, given query \( i \in [n] \), report \( \tilde{x}_i \) such that WHP
\[ \tilde{x}_i \in x_i \pm \alpha \|x\|_p. \]

Observe: \( \|x\|_1 \geq \|x\|_2 \geq \ldots \geq \|x\|_\infty \), hence higher norms (larger \( p \)) give better accuracy. We will see an algorithm for \( \ell_1 \), which is the easiest.

**Exer:** Show that the \( \ell_1 \) and \( \ell_2 \) norms differ by at most a factor of \( \sqrt{n} \), and that this is tight. Do the same for \( \ell_2 \) and \( \ell_\infty \).

It is not difficult to see that \( \ell_\infty \) point query is hard. For instance, with \( \alpha < 1/2 \) we could recover an arbitrary binary vector \( x \in \{0, 1\}^n \), which (at least intuitively) requires \( \Omega(n) \) bits to store.

**Theorem 4** [Cormode-Muthukrishnan, 2005]: There is a streaming algorithm for \( \ell_1 \) point queries that uses a (linear) sketch of \( O(\alpha^{-1} \log n) \) memory words to achieve accuracy \( \alpha \) with success probability \( 1 - 1/n^2 \).

We will initially assume all \( x_i \geq 0 \).

**Algorithm CountMin:**

(Assume all \( x_i \geq 0 \).)

1. Init: set \( w = 4/\alpha \) and choose a random hash function \( h : [n] \to [w] \).
2. Update: maintain vector \( S = [S_1, \ldots, S_w] \) where \( S_j = \sum_{i: h(i) = j} x_i \).
3. Output: to estimate \( x_i \) report \( \tilde{x}_i = S_{h(i)} \)

The update step can indeed be implemented in a streaming fashion because the sketch is some linear map \( L : \mathbb{R}^n \to \mathbb{R}^w \), (observe that \( S_j = \sum_i 1_{\{h(i) = j\}} x_i \)), and thus \( L(x + e_i) = L(x) + L(e_i) \).

We call \( S \) a sketch to emphasize it is a succinct version of the input, and \( L \) a sketching matrix.

**Analysis (correctness):** We saw in class that \( \tilde{x}_i \geq x_i \) and \( \Pr[\tilde{x}_i \geq x_i + \alpha \|x\|_1] \leq 1/4. \)
Algorithm CountMin⁺:

1. Run \( t = \log n \) independent copies of algorithm CountMin, keeping in memory the vectors \( S^1, \ldots, S^t \) (and functions \( h^1, \ldots, h^t \))

2. Output: the minimum of all estimates \( \hat{x}_i = \min_{l \in [t]} S^l_{h^l(i)} \)

Analysis (correctness): As before, \( \hat{x}_i \geq x_i \) and

\[
\Pr[\hat{x}_i > x_i + \alpha \|x\|_1] \leq (1/4)^t = 1/n^2.
\]

By a union bound, with probability at least \( 1 - 1/n \), for all \( i \in [n] \) we will have \( x_i \leq \hat{x}_i \leq x_i + \alpha \|x\|_1 \).

Space requirement: \( O(\alpha^{-1} \log n) \) words (for success probability \( 1 - 1/n^2 \)), without counting memory used to represent/store the hash functions.

Exer: Let \( x \in \mathbb{R}^n \) be the frequency vector of a stream of \( m \) items (insertions only). Show how to use the CountMin⁺ sketch seen in class (for \( \ell_1 \) point queries) to estimate the median of \( x \), which means to report an index \( j \in [n] \) that with high probability satisfies \( \sum_{i=1}^{j} x_i \in (\frac{1}{2} \pm \epsilon)m \).

General \( x \) (allowing negative entries):

Observe that Algorithm CountMin actually extends to general \( x \) that might be negative, and achieves the guarantee

\[
\Pr[\hat{x}_i \notin x_i \pm \alpha \|x\|_1] \leq 1/4.
\]

Exer: complete the proof.

Next class we will see how to amplify the success probability, using median (instead of minimum) of \( O(\log n) \) independent repetitions.