1 Frequency Moments and the AMS algorithm

\textbf{\(\ell_p\)-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 \(p = 2\).

\textbf{Theorem 1 [Alon, Matthias, and Szegedy, 1996]:} One can estimate the \(\ell_2\) norm within factor \(1 + \varepsilon\) [with high constant probability] using a linear sketch of size (dimension) \(s = O(\varepsilon^{-2})\). It implies, in particular, a streaming algorithm.

\textbf{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.

\textbf{Analysis:} We saw in class that \(\mathbb{E}[Z^2] = \sum_i x_i^2 = \|x\|_2^2\), and \(\text{Var}(Z^2) \leq 3(\mathbb{E}[Z^2])^2\).

\textbf{Exer:} Refine the analysis from class to show that \(\text{Var}(Z^2) \leq 2(\mathbb{E}[Z^2])^2\).

\textbf{Algorithm AMS+:}
1. Run \(t = O(1/\varepsilon^2)\) independent copies of Algorithm AMS, denoting their \(Z\) values by \(Y_1, \ldots, Y_t\), and output their mean \(\bar{Y} = \sum_j Y_j^2 / t\).

Observe that the sketch \((Y_1, \ldots, Y_t)\) is still linear.

---

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

**Analysis:** We saw in class that

$$\Pr[|\hat{Y} - EY| \geq \varepsilon E\hat{Y}] \leq \frac{\text{Var}(\hat{Y})}{\varepsilon^2 (E\hat{Y})^2} \leq \frac{3}{t\varepsilon^2}.$$  

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

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

**How to store the $n$ values $r_1, \ldots, r_n$:**

Observe that the analysis of algorithm AMS work as long as $r_1, \ldots, r_n$ are 4-wise independent. (The $t$ repetitions are independent.)

**Exer:** Show how the construction we saw for pairwise independent hash functions $h: [n] \to [M]$ can be extended to construct $k$-wise independent hashes (random variables) using $O(k \log n)$ truly random bits (storage).

**Hint:** Use higher-degree polynomials, and rely on the determinant of a Vandermonde matrix.

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

## 2 $\ell_2$ Point Query via CountSketch

The idea is to hash coordinates to buckets (similar to algorithm CountMin), but furthermore use tug-of-war inside each bucket (as in algorithm AMS). The analysis will show it is a good estimate for each $x_2^2$ (instead of $x_i$).

**Theorem 2 [Charikar, Chen and Farach-Colton, 2003]:** One can estimate $\ell_2$ point queries using a (linear) sketch of $O(\alpha^{-2})$ memory words within error $\alpha$ [with constant high probability].

It achieves better accuracy than CountMin ($\ell_2$ instead of $\ell_1$), but requires more storage ($1/\alpha^2$ instead of $1/\alpha$).

**Algorithm CountSketch:**

1. **Init:** Set $w = 4/\alpha^2$ and choose a pairwise independent hash function $h: [n] \to [w]$
2. Choose pairwise independent signs $r_1, \ldots, r_n \in \{-1, +1\}$
3. **Update:** Maintain vector $S = [S_1, \ldots, S_w]$ where $S_j = \sum_{i: h(i) = j} r_i x_i$.
4. **Output:** To estimate $x_i$ return $\tilde{x}_i = r_i \cdot S_{h(i)}$.

Storage requirement: $O(w)$ words, i.e., $O(\alpha^{-2} \log(nm))$ bits. The hash functions can be stored using $O(\log n)$ bits.

**Correctness:** We saw in class that $\Pr[|\tilde{x}_i - x_i|^2 \geq \alpha^2 \|x\|_2^2] \leq 1/4$, i.e., with high (constant)
probability, $x_i \in x_i \pm \alpha \|x\|_2$.

**Exer:** Explain how to amplify the success probability to $1 - 1/n^2$ using the median of $O(\log n)$ independent copies.

## 3 Application 1: Heavy Hitters

**Problem Definition:** For parameter $\phi \in (0, 1)$ and $p \in [1, \infty)$, define

$$HH_p^\phi(x) = \{i \in [n] : |x_i| \geq \phi \|x\|_p\}.$$  

Observe that its cardinality is bounded by $\left|HH_p^\phi(x)\right| \leq 1/\phi^p$.

We will focus on $p = 1$ and $\phi$ is “not too small”.

**Approximate Heavy Hitters:**

Parameters: $\phi, \varepsilon \in (0, 1)$.

Goal: return a set $S \subseteq [n]$ such that

$$HH_p^\phi \subseteq S \subseteq HH_p^\phi(1 - \varepsilon).$$

**Reduction from HH to point query (for $p = 1$):**

Assume we have an algorithm for $\ell_1$ point queries with parameter $\alpha = \varepsilon \phi/2$. Amplify the success probability to $1 - \frac{1}{3^2}$ (if needed).

1. compute, using that algorithm, an estimate $\tilde{x}_i$ for every $i \in [n]$ (this step takes time $O(n \log n)$ or even more)

2. report the set $S = \{i : \tilde{x}_i \geq (\phi - \varepsilon \phi/2)\|x\|_1\}$ (it is easy to know $\|x\|_1$ when $x \geq 0$, but more difficult in general)

Storage requirement: We can employ algorithm CountMin+ for $\ell_1$ point queries, which requires $O(\alpha^{-1} \log n)$ words, and has error probability $1/n^2$, which is small enough. Then our approximate HHI algorithm will take $O(\phi^{-1} \varepsilon^{-1} \log^2 n)$ bits.

Correctness: With probability $\geq 2/3$, all the $n$ estimates are correct within additive $\varepsilon/2$. In this case, $S$ contains all the $\phi$-HH, and is contained in the $(\phi(1 - \varepsilon))$-HH.

**Exer:** Extend the above approach to $p = 2$ (using CountSketch). How much storage it requires? Use the AMS sketch to estimate the $\ell_2$-norm.

## 4 Application 2: Compressed Sensing (or Sparse Recovery)

**Problem Definition:** The input is a “signal” $x \in \mathbb{R}^n$, but instead of reading it directly we have only via linear measurements, i.e., we can observe/access $y_i = \langle A_i, x \rangle$ for $A_1, \ldots, A_p \in \mathbb{R}^n$ of our
choice. Informally, the goal is to design few $A_i$’s and then to use them recover $x$. We shall focus on non-adaptive $A_i$, i.e., the entire sequence has to be determined in advance.

Let $A_{p \times n}$ be a matrix whose rows are the $A_i$’s, then we know that $Ax = y$. A trivial solution is to choose $A$ that is invertible, which requires $p = n$. In general, this is optimal, because for smaller $p$ there might be infinitely many solutions $x$ to $Ax = y$.

Initial goal: Suppose that $x$ is $k$-sparse (has at most $k$ nonzeros, i.e., $\|x\|_0 = k$). What $p = p(n, k)$ is needed to recover $x$?

True goal: Suppose $x$ is approximately $k$-sparse. For what $p$ can we recover an approximation to $x$?

Remark: In most applications, it’s preferable that $A$ has bounded precision (i.e., the entries of $A$ are integers of bounded magnitude), as otherwise $y$ must be “acquired” with very high precision. Sometimes it’s even important that $A$’s entries are nonnegative.

CountMin Approach: Recall that CountMin is a (randomized) linear sketch of $x \in \mathbb{R}^n$, hence it can be viewed as multiplying $x$ by some matrix $A$ with $p = O(\alpha^{-1} \log n)$ rows.

Sparse 0-1 vector: Suppose first $x \in \{0, 1\}^n$ and is $k$-sparse. Then $\|x\|_1 = k$, and a CountMin+ sketch of accuracy $\alpha = \frac{1}{k}$ succeeds with probability at least $1 - 1/n$ in estimating all $x_i$’s within additive $\pm \alpha \|x\|_1 \leq \pm \frac{1}{3}$, which can distinguish whether $x_i$ is 0 or 1.

Sparse vector: If the nonzeros of $x$ have different magnitudes, the above approach might require $\alpha \ll \frac{1}{k}$.

But a deeper inspection of CountMin shows that every coordinate has a good chance to “not collide” with any nonzero coordinate. This behavior is amplified by the repetitions + median trick’s, and then WHP the estimator is exact, i.e., $\hat{x}_i = x_i$.

Approximately sparse vector: We will now prove an even more general result.

For $z \in \mathbb{R}^n$, denote by $z_{\text{top}}(k)$ the vector $z$ after zeroing all but the $k$ heaviest entries (largest in absolute value), breaking ties arbitrarily. Notice this vector is the “best” $k$-sparse approximation to $z$. Similarly, denote by $z_{\text{tail}}(k) \in \mathbb{R}^n$ the vector $z$ after zeroing the $k$ heaviest entries. Then $z_{\text{tail}}(k) = z - z_{\text{top}}(k)$ is the “error” of approximating $z$ by a $k$-sparse vector.

Theorem 3 [Cormode and MuthuKrishnan, 2006]: CountMin+ with parameter $\alpha = \varepsilon/k$ can be used to recover a vector $x' \in \mathbb{R}^n$ that satisfies

$$\|x - x'\|_1 \leq (1 + 3\varepsilon)\|x_{\text{tail}}(k)\|_1.$$ 

In fact, $x' = \hat{x}_{\text{top}}(k)$ and is thus $k$-sparse. (Recall $\hat{x} \in \mathbb{R}^n$ is the estimate of algorithm CountMin.)

The above condition is usually called an $\ell_1/\ell_1$ guarantee.

Remark 1: Observe that if $x$ is $k$-sparse, then this method recovers it (exactly). In general, it guarantees the output’s “quality” (distance from true $x$) is comparable to the best $k$-sparse vector.

Remark 2: Different constructions achieve/optimize for other guarantees like different norms, deterministic recovery, small explicit description of $A$, or fast recovery time. Often, the optimal number of measurements is $O(k \log(n/k))$ (ignoring dependence on $\varepsilon$).
**Lemma 3a:** CountMin with parameter $\alpha = \varepsilon/k$ computes, with high probability, an estimate $\hat{x}_i \in x_i \pm \alpha \|x_{\text{tail}(k)}\|_1$, i.e., $\|x - \hat{x}\|_\infty \leq \alpha \|x_{\text{tail}(k)}\|_1$.

**Exer:** Prove this lemma.

**Hint:** Show that with high probability, both (a) coordinate $i$ will not collide with the $k$ (other) heaviest coordinates and (b) the contribution from the rest (tail) is comparable to the expectation.

**Lemma 3b:** If $\|x - \hat{x}\|_\infty \leq \alpha \|x_{\text{tail}(k)}\|_1$ then $\|x - \hat{x}_{\text{top}(k)}\|_1 \leq (1 + 3k\alpha) \|x_{\text{tail}(k)}\|_1$.

**Notice that we bound the error using $\ell_1$ norm (stronger).**

**Proof of lemma:** We will use $z_T$ to denote the vector $z$ after zeroing all coordinates outside $T \subset [n]$.

Let $\hat{T} \subset [n]$ be the indices of the $k$ heaviest coordinates in $\hat{x}$, then by definition $x' = \hat{x}_{\text{top}(k)} = \hat{x}_{\hat{T}}$.

Let $T \subset [n]$ be the indices of the $k$ heaviest coordinates in $x$, hence $x_T = x_{\text{top}(k)}$.

Now calculate (all norms are $\ell_1$-norms):

\[
\|x - x'\| = \|x_{\hat{T}} - x'_{\hat{T}}\| + \|x_{-\hat{T}}\|
\leq \|x_{\hat{T}} - x'_{\hat{T}}\| + \|x\| - \|x_{\hat{T}}\|
\leq 2\|x_{\hat{T}} - x'_{\hat{T}}\| + \|x\| - \|x_{\hat{T}}\|
\leq 2\|x_{\hat{T}} - x'_{\hat{T}}\| + \|x\| - \|x_{\hat{T}}\|
\leq (2|\hat{T}| + |\hat{T}|) \|x_{\text{tail}(k)}\|.
\]

**QED.**

**Exer:** Can you extend the above sparse recovery to $\ell_2/\ell_2$ guarantee by using CountSketch (instead of CountMin)? How many measurements would it require?