Problems 1, 3, 4 require you to have read the lecture notes or watched the lecture, but you should be able to attempt them on your own after that. They are important to attempt, in order to build a good understanding about the algorithms seen in class.

Problem 2 does not require having seen the lecture, and can be skipped. It is a useful fact, good to know, but that you can take for granted.

Problem 5 is good to go over if you have time, but also can be skipped. In that can, go over the solution afterwards.

Problem 6 is on the more difficult side, but worth doing to understand why the MG algorithms is a sketching algorithm.

Problem 7 is interesting if you have time, but not necessary.

Warm-up

Problem 1. Discussion: what are the parallels between Bloom filters and CountMinSketch?

Solution 1. One can think of CountMinSketch as a "counting" version of a Bloom Filter.

- Bloom filter: we set bits to 1 in each of the T hash tables, and report the AND;
- CountMinSketch: we get counts in each of the T hash tables, and report the MIN;

AND(bits) = MIN(bits), so you can see the Bloom filter as reporting the MIN too (but capping the counts at 1).

Problem 2. Prove the following fact about "monotonicity of ℓ_p norms": if $x \in \mathbb{R}^d$, then $\|x\|_\infty \leq \|x\|_2 \leq \|x\|_1$. Show, in addition, that $\|x\|_2 \geq \|x\|_1/\sqrt{d}$. When are these inequalities tight?
(($\star$) More generally: if $1 \leq p \leq q \leq \infty$, then $\|x\|_q \leq \|x\|_p$.)

Solution 2. For the first half of the inequality:

$$\|x\|_\infty^2 = (\max_{i\in[d]} x_i)^2 \leqslant \sum_{i=1}^{d} x_i^2 = \|x\|_2^2.$$

For the second half, i.e., to show $\|x\|_2 \leqslant \|x\|_1$, we have

$$\sum_{i=1}^{d} x_i^2 = \sum_{i=j:i\in[d],j\in[d]} |x_i|\cdot|x_j| \leqslant \sum_{i=1}^{d}\sum_{j=1}^{d} |x_i|\cdot|x_j| = \|x\|_1^2.$$

Lastly, to show $\|x\|_2 \geqslant \|x\|_1/\sqrt{d}$, we have by the Cauchy-Schwarz inequality (between x and $y = (1,1,\ldots,1)$),

$$\|x\|_1 = \sum_{i=1}^{d} |x_i|\cdot 1 = \sum_{i=1}^{d} |x_i|\cdot|y_i| \leqslant \|x\|_2\cdot\|y\|_2 = \|x\|_2\cdot\sqrt{\sum_{i=1}^{d} 1} = \|x\|_2\cdot\sqrt{d}.$$

For more general cases, see, e.g., the answer: https://math.stackexchange.com/a/483825.

Problem 3. Discuss the advantages and disadvantages of Misra-Gries versus CountMinSketch when used in the cash register model: speed, memory, approximation. Can you think of a situation where having an overestimate (CountMinSketch) is better than an underestimate (Misra-Gries)?

Solution 3. CountMinSketch might use more memory but the runtime is actually much faster in some cases. In addition, it provides a *linear* sketch.

For Misra-Gries , the algorithm needs to first look up in the set (using hash table this will be $O(1)$ each time in expectation) and then sometimes it needs to decrement everything in the array, and that takes $O(k) = O\left(\frac{1}{\varepsilon}\right)$ time.

For CountMinSketch , each time you only need to look up $O(\log \frac{n}{\delta})$ and update $O(\log \frac{n}{\delta})$ elements of the array.

For the last part, consider the task of finding heavy hitters, $H_\varepsilon := \{j \in [n] : f_j \geqslant \varepsilon \cdot m\}$: that is, outputting a set $\hat{H}$ containing all ε-heavy hitters, and not too many other elements. Having an overestimate means that setting the threshold to $\varepsilon \cdot m \leqslant f_j \leqslant \hat{f}_j$ suffices to have $H_\varepsilon \subseteq \hat{H}$; in addition (this is only an informal, rule-of-thumb idea, not necessarily always true) one can hope/expect that there will be fewer false positives ($|\hat{H}|$ not too large), as the threshold to be included is εm, compared to a smaller threshold such as $\frac{\varepsilon}{2}m$ when solving the Heavy Hitters problem with Misra-Gries.

Problem solving

Problem 4. For the same space budget s (ignoring the constants in the $O(\cdot)$'s), are the theoretical guarantees provided by CountMinSketch better, worse, or incomparable to those of CountSketch?

Solution 4. Note the difference between the two guarantees, ignoring the constant factors and the dependence on the probability of failure δ.

- CS (CountSketch):

$$s_1 = O\left(\frac{1}{\varepsilon^2}\log(mn)\right) \Rightarrow |\widehat{f}_j - f_j| \leqslant \varepsilon \cdot \|f\|_2.$$

- CMS (CountMinSketch):

$$s_2 = O\left(\frac{1}{\varepsilon}\log(mn)\right) \Rightarrow |\hat{f}_j - f_j| \leqslant \varepsilon \cdot \|f\|_1.$$

Note the relation: $\frac{\|f_1\|}{\sqrt{n}} \leqslant \|f_2\| \leqslant \|f_1\|$.

Say we fix the same $s = s_1 = s_2$. This then gives $s_1 = s_2 = \frac{1}{\varepsilon^2}\log(mn)$ and thus

$$\text{CS} : |\widehat{f}_j - f_j| \leqslant \varepsilon \cdot \|f\|_2.$$

$$\text{CMS} : |\hat{f}_j - f_j| \leqslant \varepsilon^2 \cdot \|f\|_1.$$

Comparing $\text{Err(CS)} = \varepsilon \cdot \|f\|_2$ and $\text{Err(CMS)} = \varepsilon^2 \cdot \|f\|_1$, we have that

$$\varepsilon \cdot \frac{\|f_1\|}{\sqrt{n}} \leqslant \varepsilon \cdot \|f\|_2 \leqslant \varepsilon \cdot \|f_1\|.$$

$$\frac{\text{Err(CMS)}}{\varepsilon\sqrt{n}} \leqslant \text{Err(CS)} \leqslant \frac{\text{Err(CMS)}}{\varepsilon}.$$

Consider two cases here:

1. When $\varepsilon \ll 1/\sqrt{n}$, then $\text{Err(CMS)} \leqslant \varepsilon\sqrt{n} \cdot \text{Err(CS)} \ll \text{Err(CS)}$. So CMS is better when ε is very small.

2. But if ε is **constant** and n **becomes big**, and $\|f\|_2$ is well spread out (say it's uniform, then $f_j = \frac{m}{n}$), we have that

$$\|f\|_2 = \sqrt{\sum_{i=1}^{n} \left(\frac{m}{n}\right)^2} = \frac{m}{\sqrt{n}} \ll m = \|f\|_1.$$

In this case, $\text{Err(CMS)} = \varepsilon \cdot \|f\|_2 = \Theta(\|f\|_2) = \Theta\left(\frac{m}{\sqrt{n}}\right) \ll \Theta(m) = \Theta(\|f\|_1) = \varepsilon^2 \cdot \|f\|_1 = \text{Err(CS)}$.

Takeaway: they each have their own favored regime.

Problem 5. Generalise the analysis of the CountMinSketch algorithm to show it works in the *strict* turnstile model, where updates of the stream are of the form $(j, c) \in [n] \times \{-B, \ldots, B\}$ (can be negative) but one must have $f_j \geq 0$ at every time. Check the guarantees you can provide on the output $\widehat{f}$. Does the analysis extend to the general turnstile model, where f_j can become negative?

Solution 5. (Proof sketch: for more details, see, e.g., `https://www.sketchingbigdata.org/fall20/lec/notes.pdf`, Section 4.1.1.)

Since $f_j \geqslant 0$ at all times, the whole argument can go through as is. Note however that it doesn't work when $f_j < 0$, as Markov inequality needs the condition that the random variable is non-negative: this rules out the general turnstile model.

Problem 6. ($\star$) Show that the Misra-Gries algorithm is a sketching algorithm: namely, suppose we run Misra-Gries (with the same parameter $k = \lceil 1/\varepsilon \rceil$) on two streams σ_1, σ_2, getting output vectors $\widehat{f}^{(1)}, \widehat{f}^{(2)}$. Combine then as follows:

1. Set $\widehat{f} \leftarrow \widehat{f}^{(1)} + \widehat{f}^{(2)}$
2. If $\widehat{f}$ has more than k non-zero entries, let $v > 0$ be the value of the $(k+1)$-th, in non-increasing order.
3. Set $\widehat{f}_j \leftarrow \max(\widehat{f}_j - v, 0)$ for all j

a) Argue that $\widehat{f}$ has at most k non-zero entries.

b) Show that the sketch $\widehat{f}$ provides the original Misra-Gries estimation guarantees, for the combined stream $\sigma_1 \circ \sigma_2$.

c) Is this sketch a linear sketch?

Solution 6. Note the the guarantee of MG's algorithm is a bit stronger than stated, which will be handy in this proof. Denote total number of counts in the $k = \lceil 1/\varepsilon \rceil$ arrays as $\hat{n}_1$ (resp. $\hat{n}_2$) for stream σ_1 (resp. σ_2) after running MG. We in fact have (subtracting one from all $k+1$ when k array is full; and at the end $\hat{n}_1$ remains)

$$f_j^{(1)} - \frac{n_1 - \hat{n}_1}{k+1} \leqslant \hat{f}_j^{(1)} \leqslant f_j^{(1)}.$$

$n_1 = |\sigma_1|, n_2 = |\sigma_2|$. So the error for is bounded by $\frac{n_1-\hat{n}_1}{k+1}$ and $\frac{n_2-\hat{n}_2}{k+1}$ respectively. Combining the two we have that the error is at most

$$\leqslant \frac{n_1 - \hat{n}_1}{k+1} + \frac{n_2 - \hat{n}_2}{k+1}.$$

Factoring in the subtraction step of v. We have that the error would increase to be

$$\leqslant \frac{n_1 - \hat{n}_1}{k+1} + \frac{n_2 - \hat{n}_2}{k+1} + v.$$

What is left is to bound v. Denote $\hat{n}_{12}$ as the number of counts left in k array after the subtraction of v. Since we are subtracting for $k+1$ array all v (and potentially more, but others could be starting at 1; the $(k+2)$-th and above entries), we have that

$$\hat{n}_1 + \hat{n}_2 - \hat{n}_{12} \geqslant (k+1) \cdot v.$$

Thus, $v \leqslant \frac{\hat{n}_1+\hat{n}_2-\hat{n}_{12}}{k+1}$. Combining both, the error is at most

$$\frac{n_1 - \hat{n}_1}{k+1} + \frac{n_2 - \hat{n}_2}{k+1} + \frac{\hat{n}_1 + \hat{n}_2 - \hat{n}_{12}}{k+1} = \frac{n_1 + n_2 - \hat{n}_{12}}{k+1} = \frac{n_{12} - \hat{n}_{12}}{k+1}.$$

(Proof credit: Theorem 2.2 of https://users.cs.duke.edu/ pankaj/publications/papers/merge-summ.pdf.)

Advanced

Problem 7. Modify the CountMinSketch algorithm so that it outputs a *list* of the ℓ_1 Heavy Hitters in the strict turnstile model: that is (similarly to an exercise in Tutorial 8), given parameter $\varepsilon \in (0,1]$, it should output a set $H \subseteq [n]$ such that $H_\varepsilon(\sigma) \subseteq H \subseteq H_{\varepsilon/2}(\sigma)$, where

$$H_\varepsilon(\sigma) = \{j \in [n] : f_j \geq \varepsilon \cdot \|f\|_1\}$$