Learning Goals

- Frequency Estimation
- The Count-Min Sketch
- Cash register and turnstile models
- Count-Sketch

Frequency Estimation

- Recall the streaming model: the stream $i_1, \ldots, i_n \in [d] := \{1, \cdots, d\}$.

Frequency Estimation

- Recall the streaming model: the stream $i_1, \ldots, i_n \in [d] := \{1, \cdots, d\}$.
- The frequency vector $x \in \mathbb{Z}^d$: $x_j = |\{t : i_t = j\}|$.

Frequency Estimation

- Recall the streaming model: the stream $i_1, \ldots, i_n \in [d] \coloneqq \{1, \cdots, d\}$.
- The frequency vector $x \in \mathbb{Z}^d$: $x_j = |\{t : i_t = j\}|$.
- The AMS sketch estimates $||x||_2$.

Frequency Estimation

- Recall the streaming model: the stream $i_1, \ldots, i_n \in [d] \coloneqq \{1, \cdots, d\}$.
- The frequency vector $x \in \mathbb{Z}^d$: $x_j = |\{t : i_t = j\}|$.
- The AMS sketch estimates $||x||_2$.
- What if we would like an estimate of each x_j? This is called Frequency Estimation.

Frequency Estimation

- Recall the streaming model: the stream $i_1, \ldots, i_n \in [d] := \{1, \cdots, d\}$.
- The frequency vector $x \in \mathbb{Z}^d$: $x_j = |\{t : i_t = j\}|$.
- The AMS sketch estimates $||x||_2$.
- What if we would like an estimate of each x_j? This is called Frequency Estimation.
- Recall Bloom filter: we wanted to know quickly whether an element is present, allowing mistakes.

Frequency Estimation

- Recall the streaming model: the stream $i_1, \ldots, i_n \in [d] := \{1, \cdots, d\}$.
- The frequency vector $x \in \mathbb{Z}^d$: $x_j = |\{t : i_t = j\}|$.
- The AMS sketch estimates $||x||_2$.
- What if we would like an estimate of each x_j? This is called Frequency Estimation.
- Recall Bloom filter: we wanted to know quickly whether an element is present, allowing mistakes.
  - There we maintained many hash tables, and return Yes only if there is a record in all tables

Frequency Estimation

- Recall the streaming model: the stream $i_1, \ldots, i_n \in [d] := \{1, \cdots, d\}$.
- The frequency vector $x \in \mathbb{Z}^d$: $x_j = |\{t : i_t = j\}|$.
- The AMS sketch estimates $||x||_2$.
- What if we would like an estimate of each x_j? This is called Frequency Estimation.
- Recall Bloom filter: we wanted to know quickly whether an element is present, allowing mistakes.
    - There we maintained many hash tables, and return Yes only if there is a record in all tables
    - Can we emulate the idea here?

Attempt with one hash table

- Let's try a hash function h from $[d]$ to $[k]$ for some k that we decide later.

Attempt with one hash table

- Let's try a hash function h from $[d]$ to $[k]$ for some k that we decide later.
- Maintain counters $C[1], \cdots, C[k]$, initialized to 0.

Attempt with one hash table

- Let's try a hash function h from $[d]$ to $[k]$ for some k that we decide later.
- Maintain counters $C[1], \cdots, C[k]$, initialized to 0.
- When i_t arrives, increase $C[h(i_t)]$ by 1.

Attempt with one hash table

- Let's try a hash function h from $[d]$ to $[k]$ for some k that we decide later.
- Maintain counters $C[1], \cdots, C[k]$, initialized to 0.
- When i_t arrives, increase $C[h(i_t)]$ by 1.
- In the end, to estimate x_j, we return $C[h(j)]$.

Attempt with one hash table

- Let's try a hash function h from $[d]$ to $[k]$ for some k that we decide later.
- Maintain counters $C[1], \cdots, C[k]$, initialized to 0.
- When i_t arrives, increase $C[h(i_t)]$ by 1.
- In the end, to estimate x_j, we return $C[h(j)]$.
- Clearly, $C[h(j)]$ is an overestimate of x_j due to clashes.

Attempt with one hash table

- Let's try a hash function h from $[d]$ to $[k]$ for some k that we decide later.
- Maintain counters $C[1], \cdots, C[k]$, initialized to 0.
- When i_t arrives, increase $C[h(i_t)]$ by 1.
- In the end, to estimate x_j, we return $C[h(j)]$.
- Clearly, $C[h(j)]$ is an overestimate of x_j due to clashes.
- How many clashes are there?

Attempt with one hash table

- Let's try a hash function h from $[d]$ to $[k]$ for some k that we decide later.
- Maintain counters $C[1], \cdots, C[k]$, initialized to 0.
- When i_t arrives, increase $C[h(i_t)]$ by 1.
- In the end, to estimate x_j, we return $C[h(j)]$.
- Clearly, $C[h(j)]$ is an overestimate of x_j due to clashes.
- How many clashes are there?
  - If h is sampled from a universal hash family, in expectation $C[h(j)] \leq x_j + \frac{n}{k}$.

Attempt with one hash table

- Let's try a hash function h from $[d]$ to $[k]$ for some k that we decide later.
- Maintain counters $C[1], \cdots, C[k]$, initialized to 0.
- When i_t arrives, increase $C[h(i_t)]$ by 1.
- In the end, to estimate x_j, we return $C[h(j)]$.
- Clearly, $C[h(j)]$ is an overestimate of x_j due to clashes.
- How many clashes are there?
  - If h is sampled from a universal hash family, in expectation $C[h(j)] \leq x_j + \frac{n}{k}$.
  - If we set $k = \frac{1}{\epsilon}$, this would give an estimate with ϵn *additive* error in expectation.

Attempt with one hash table

- Let's try a hash function h from $[d]$ to $[k]$ for some k that we decide later.
- Maintain counters $C[1], \cdots, C[k]$, initialized to 0.
- When i_t arrives, increase $C[h(i_t)]$ by 1.
- In the end, to estimate x_j, we return $C[h(j)]$.
- Clearly, $C[h(j)]$ is an overestimate of x_j due to clashes.
- How many clashes are there?
  - If h is sampled from a universal hash family, in expectation $C[h(j)] \leq x_j + \frac{n}{k}$.
  - If we set $k = \frac{1}{\epsilon}$, this would give an estimate with ϵn *additive* error in expectation.
  - Let's try pushing the guarantee to "with high probability" by repetitions!

Count-Min Sketch

The COUNT-MIN algorithm by Cormode and Muthukrishnan (2005):

- Sample ℓ hash functions $h_1, \cdots, h_\ell : [d] \to [k]$ independently from a universal hash family; let k be $\frac{2}{\epsilon}$.

Count-Min Sketch

The COUNT-MIN algorithm by Cormode and Muthukrishnan (2005):

- Sample ℓ hash functions $h_1, \cdots, h_\ell : [d] \to [k]$ independently from a universal hash family; let k be $\frac{2}{\epsilon}$.
- Maintain counters $C_1[1], \cdots, C_1[k], C_2[1], \cdots, C_2[k], \cdots, C_\ell[1], \cdots, C_\ell[k]$, initialized to 0.

Count-Min Sketch

The COUNT-MIN algorithm by Cormode and Muthukrishnan (2005):

- Sample ℓ hash functions $h_1, \cdots, h_\ell : [d] \to [k]$ independently from a universal hash family; let k be $\frac{2}{\epsilon}$.
- Maintain counters $C_1[1], \cdots, C_1[k], C_2[1], \cdots, C_2[k], \cdots, C_\ell[1], \cdots, C_\ell[k]$, initialized to 0.
- When i_t arrives, for $j = 1, \cdots, \ell$, increase $C_j[h_j(i_t)]$ by one.

Count-Min Sketch

The Count-Min algorithm by Cormode and Muthukrishnan (2005):

- Sample ℓ hash functions $h_1, \cdots, h_\ell : [d] \to [k]$ independently from a universal hash family; let k be $\frac{2}{\epsilon}$.
- Maintain counters $C_1[1], \cdots, C_1[k], C_2[1], \cdots, C_2[k], \cdots, C_\ell[1], \cdots, C_\ell[k]$, initialized to 0.
- When i_t arrives, for $j = 1, \cdots, \ell$, increase $C_j[h_j(i_t)]$ by one.
- At the end, to estimate x_j, return $\min\{C_1[h_1(j)], \cdots, C_\ell[h_\ell(j)]\}$.

Analysis of Count-Min

- For each (ball) $j \in [d]$ and (bin) $i \in [\ell]$, $x_j \leq C_i[h_i(j)]$, so the output is never smaller than x_j.

Analysis of Count-Min

- For each (ball) $j \in [d]$ and (bin) $i \in [\ell]$, $x_j \leq C_i[h_i(j)]$, so the output is never smaller than x_j.
- For each $j \in [d]$ and $i \in [\ell]$, let $Y_{i,j}$ be the number of elements that clashes with j under h_i, then $C_i[h_i(j)] \leq x_j + Y_{i,j}$.

Analysis of Count-Min

- For each (ball) $j \in [d]$ and (bin) $i \in [\ell]$, $x_j \leq C_i[h_i(j)]$, so the output is never smaller than x_j.
- For each $j \in [d]$ and $i \in [\ell]$, let $Y_{i,j}$ be the number of elements that clashes with j under h_i, then $C_i[h_i(j)] \leq x_j + Y_{i,j}$.
  - By universality, $\mathbf{E}[Y_{ij}] \leq \frac{n}{k}$.

Analysis of Count-Min

- For each (ball) $j \in [d]$ and (bin) $i \in [\ell]$, $x_j \leq C_i[h_i(j)]$, so the output is never smaller than x_j.
- For each $j \in [d]$ and $i \in [\ell]$, let $Y_{i,j}$ be the number of elements that clashes with j under h_i, then $C_i[h_i(j)] \leq x_j + Y_{i,j}$.
    - By universality, $\mathbf{E}[Y_{ij}] \leq \frac{n}{k}$.
    - By Markov inequality, $\mathbf{Pr}[Y_{ij} \geq \frac{2n}{k} = \epsilon n] \leq \frac{1}{2}$.

Analysis of Count-Min

- For each (ball) $j \in [d]$ and (bin) $i \in [\ell]$, $x_j \leq C_i[h_i(j)]$, so the output is never smaller than x_j.
- For each $j \in [d]$ and $i \in [\ell]$, let $Y_{i,j}$ be the number of elements that clashes with j under h_i, then $C_i[h_i(j)] \leq x_j + Y_{i,j}$.
  - By universality, $\mathbf{E}[Y_{ij}] \leq \frac{n}{k}$.
  - By Markov inequality, $\mathbf{Pr}[Y_{ij} \geq \frac{2n}{k} = \epsilon n] \leq \frac{1}{2}$.
- By independence, $\mathbf{Pr}[\cap_i \{Y_{ij} \geq \epsilon n\}] \leq 2^{-\ell}$.

Analysis of Count-Min

- For each (ball) $j \in [d]$ and (bin) $i \in [\ell]$, $x_j \leq C_i[h_i(j)]$, so the output is never smaller than x_j.
- For each $j \in [d]$ and $i \in [\ell]$, let $Y_{i,j}$ be the number of elements that clashes with j under h_i, then $C_i[h_i(j)] \leq x_j + Y_{i,j}$.
  - By universality, $\mathbf{E}[Y_{ij}] \leq \frac{n}{k}$.
  - By Markov inequality, $\mathbf{Pr}[Y_{ij} \geq \frac{2n}{k} = \epsilon n] \leq \frac{1}{2}$.
- By independence, $\mathbf{Pr}[\cap_i \{Y_{ij} \geq \epsilon n\}] \leq 2^{-\ell}$.
- Therefore, for some $\ell = O(\log d)$, with high probability our estimate is correct within additive ϵn error for all coordinates of x.

Analysis of Count-Min

- For each (ball) $j \in [d]$ and (bin) $i \in [\ell]$, $x_j \leq C_i[h_i(j)]$, so the output is never smaller than x_j.
- For each $j \in [d]$ and $i \in [\ell]$, let $Y_{i,j}$ be the number of elements that clashes with j under h_i, then $C_i[h_i(j)] \leq x_j + Y_{i,j}$.
  - By universality, $\mathbf{E}[Y_{ij}] \leq \frac{n}{k}$.
  - By Markov inequality, $\mathbf{Pr}[Y_{ij} \geq \frac{2n}{k} = \epsilon n] \leq \frac{1}{2}$.
- By independence, $\mathbf{Pr}[\cap_i \{Y_{ij} \geq \epsilon n\}] \leq 2^{-\ell}$.
- Therefore, for some $\ell = O(\log d)$, with high probability our estimate is correct within additive ϵn error for all coordinates of x.
- Space usage:
  - Maintaining the counters: there are $k\ell = \frac{2}{\epsilon} \log d$ counters, each taking $O(\log n)$ space.

Analysis of Count-Min

- For each (ball) $j \in [d]$ and (bin) $i \in [\ell]$, $x_j \leq C_i[h_i(j)]$, so the output is never smaller than x_j.
- For each $j \in [d]$ and $i \in [\ell]$, let $Y_{i,j}$ be the number of elements that clashes with j under h_i, then $C_i[h_i(j)] \leq x_j + Y_{i,j}$.
  - By universality, $\mathbf{E}[Y_{ij}] \leq \frac{n}{k}$.
  - By Markov inequality, $\mathbf{Pr}[Y_{ij} \geq \frac{2n}{k} = \epsilon n] \leq \frac{1}{2}$.
- By independence, $\mathbf{Pr}[\cap_i \{Y_{ij} \geq \epsilon n\}] \leq 2^{-\ell}$.
- Therefore, for some $\ell = O(\log d)$, with high probability our estimate is correct within additive ϵn error for all coordinates of x.
- Space usage:
  - Maintaining the counters: there are $k\ell = \frac{2}{\epsilon} \log d$ counters, each taking $O(\log n)$ space.
  - Maintaining the hash functions: there are $\ell = O(\log d)$ of them, each taking $O(\log d)$ space.

More General Streaming Models

- Our streaming model so far: the stream $i_1, \ldots, i_n \in [d] := \{1, \cdots, d\}$.

More General Streaming Models

- Our streaming model so far: the stream $i_1, \ldots, i_n \in [d] := \{1, \cdots, d\}$.
- The frequency vector $x \in \mathbb{Z}^d$: $x_j = |\{t : i_t = j\}|$.

More General Streaming Models

- Our streaming model so far: the stream $i_1, \ldots, i_n \in [d] := \{1, \cdots, d\}$.
- The frequency vector $x \in \mathbb{Z}^d$: $x_j = |\{t : i_t = j\}|$.
- Slight generalization: each data point at time t is a pair (i_t, Δ_t):
  - i_t is the index
  - Δ_t is the *increment* of the count at index i_t

More General Streaming Models

- Our streaming model so far: the stream $i_1, \ldots, i_n \in [d] := \{1, \cdots, d\}$.
- The frequency vector $x \in \mathbb{Z}^d$: $x_j = |\{t : i_t = j\}|$.
- Slight generalization: each data point at time t is a pair (i_t, Δ_t):
    - i_t is the index
    - Δ_t is the *increment* of the count at index i_t
- Our streaming problems so far are special cases when all $\Delta_t = 1$.

More General Streaming Models

- Our streaming model so far: the stream $i_1, \ldots, i_n \in [d] := \{1, \cdots, d\}$.
- The frequency vector $x \in \mathbb{Z}^d$: $x_j = |\{t : i_t = j\}|$.
- Slight generalization: each data point at time t is a pair (i_t, Δ_t):
  - i_t is the index
  - Δ_t is the *increment* of the count at index i_t
- Our streaming problems so far are special cases when all $\Delta_t = 1$.
- If Δ_t are positive real numbers, this is called the *cash register* model.

More General Streaming Models

- Our streaming model so far: the stream $i_1, \ldots, i_n \in [d] := \{1, \cdots, d\}$.
- The frequency vector $x \in \mathbb{Z}^d$: $x_j = |\{t : i_t = j\}|$.
- Slight generalization: each data point at time t is a pair (i_t, Δ_t):
  - i_t is the index
  - Δ_t is the *increment* of the count at index i_t
- Our streaming problems so far are special cases when all $\Delta_t = 1$.
- If Δ_t are positive real numbers, this is called the *cash register* model.
- If Δ_t are allowed to be negative, but every frequency counter x_j is guaranteed to be non-negative at all time, this is called the *strict turnstile* model.

More General Streaming Models

- Our streaming model so far: the stream $i_1, \ldots, i_n \in [d] \coloneqq \{1, \cdots, d\}$.
- The frequency vector $x \in \mathbb{Z}^d$: $x_j = |\{t : i_t = j\}|$.
- Slight generalization: each data point at time t is a pair (i_t, Δ_t):
  - i_t is the index
  - Δ_t is the *increment* of the count at index i_t
- Our streaming problems so far are special cases when all $\Delta_t = 1$.
- If Δ_t are positive real numbers, this is called the *cash register* model.
- If Δ_t are allowed to be negative, but every frequency counter x_j is guaranteed to be non-negative at all time, this is called the *strict turnstile* model.
- If Δ_t can be negative, and x_j's can be negative as well, this is called the *turnstile* model.

Frequency Estimation in Turnstile Model

Does COUNT-MIN still work in these more general settings?

- In cash register model: Δ_t are positive real numbers.

Frequency Estimation in Turnstile Model

Does Count-Min still work in these more general settings?

- In cash register model: Δ_t are positive real numbers.
  - Count-Min still works, just increase the counters by Δ_t; the error term is relaxed to $\epsilon||x||_1$.

Frequency Estimation in Turnstile Model

Does Count-Min still work in these more general settings?

- In cash register model: Δ_t are positive real numbers.
    - Count-Min still works, just increase the counters by Δ_t; the error term is relaxed to $\epsilon||x||_1$.
        - Recall $||x||_1 = \sum_i |x_i|$.

Frequency Estimation in Turnstile Model

Does COUNT-MIN still work in these more general settings?

- In cash register model: Δ_t are positive real numbers.
  - COUNT-MIN still works, just increase the counters by Δ_t; the error term is relaxed to $\epsilon||x||_1$.
    - Recall $||x||_1 = \sum_i |x_i|$.
- In the strict turnstile model, Δ_t are allowed to be negative, but every frequency counter x_j is guaranteed to be non-negative at all time.

Frequency Estimation in Turnstile Model

Does Count-Min still work in these more general settings?

- In cash register model: Δ_t are positive real numbers.
  - Count-Min still works, just increase the counters by Δ_t; the error term is relaxed to $\epsilon||x||_1$.
    - Recall $||x||_1 = \sum_i |x_i|$.
- In the strict turnstile model, Δ_t are allowed to be negative, but every frequency counter x_j is guaranteed to be non-negative at all time.
  - Count-Min still works.

Frequency Estimation in Turnstile Model

Does Count-Min still work in these more general settings?

- In cash register model: Δ_t are positive real numbers.
  - Count-Min still works, just increase the counters by Δ_t; the error term is relaxed to $\epsilon||x||_1$.
    - Recall $||x||_1 = \sum_i |x_i|$.
- In the strict turnstile model, Δ_t are allowed to be negative, but every frequency counter x_j is guaranteed to be non-negative at all time.
  - Count-Min still works.
- In the turnstile model, Δ_t can be negative, and x_j's can be negative as well.

Frequency Estimation in Turnstile Model

Does Count-Min still work in these more general settings?

- In cash register model: Δ_t are positive real numbers.
  - Count-Min still works, just increase the counters by Δ_t; the error term is relaxed to $\epsilon||x||_1$.
    - Recall $||x||_1 = \sum_i |x_i|$.
- In the strict turnstile model, Δ_t are allowed to be negative, but every frequency counter x_j is guaranteed to be non-negative at all time.
  - Count-Min still works.
- In the turnstile model, Δ_t can be negative, and x_j's can be negative as well.
  - The analysis of Count-Min is problematic in this setting. Markov inequality needs nonnegativity!

Frequency Estimation for the Turnstile Model

- What goes wrong with the analysis of Count-Min is that the error term caused by clashes can be negative.

Frequency Estimation for the Turnstile Model

- What goes wrong with the analysis of Count-Min is that the error term caused by clashes can be negative.
- $C_i[h_i(j)] = x_j$+error.

Frequency Estimation for the Turnstile Model

- What goes wrong with the analysis of COUNT-MIN is that the error term caused by clashes can be negative.
- $C_i[h_i(j)] = x_j$+error.
  - When "error" is all nonnegative, we can take minimum among $C_i[h_i(j)]$. But when error can be negative, taking the minimum may seriously underestimate x_j.

Frequency Estimation for the Turnstile Model

- What goes wrong with the analysis of COUNT-MIN is that the error term caused by clashes can be negative.
- $C_i[h_i(j)] = x_j$+error.
  - When "error" is all nonnegative, we can take minimum among $C_i[h_i(j)]$. But when error can be negative, taking the minimum may seriously underestimate x_j.
  - Similarly, taking the maximum may overestimate x_j.

Frequency Estimation for the Turnstile Model

- What goes wrong with the analysis of Count-Min is that the error term caused by clashes can be negative.
- $C_i[h_i(j)] = x_j +$error.
  - When "error" is all nonnegative, we can take minimum among $C_i[h_i(j)]$. But when error can be negative, taking the minimum may seriously underestimate x_j.
  - Similarly, taking the maximum may overestimate x_j.
  - It is natural to try the *median*.

Frequency Estimation for the Turnstile Model

- What goes wrong with the analysis of COUNT-MIN is that the error term caused by clashes can be negative.
- $C_i[h_i(j)] = x_j +$error.
  - When "error" is all nonnegative, we can take minimum among $C_i[h_i(j)]$. But when error can be negative, taking the minimum may seriously underestimate x_j.
  - Similarly, taking the maximum may overestimate x_j.
  - It is natural to try the *median*.

Claim

Let $Z_1, \cdots, Z_n$ be i.i.d. random variables. Let M be a median. There is a constant $c > 0$ such that:

- If $\mathbf{Pr}[Z_i \geq t] \leq p < \frac{1}{4}$, then $\mathbf{Pr}[M \geq t] \leq e^{-cn}$.
- If $\mathbf{Pr}[Z_i \leq t] \leq p < \frac{1}{4}$, then $\mathbf{Pr}[M \leq t] \leq e^{-cn}$.

- Algorithm: same setup and initialization as before
  - At input (i_t, Δ_t), for $i = 1, \cdots, \ell$, increase counter $C_i[h_i(i_t)]$ by Δ_t
  - In the end, as an estimate of x_j, output a median of $\{C_1[h_1(j)], \cdots, C_\ell[h_\ell(j)]\}$.

- Algorithm: same setup and initialization as before
  - At input (i_t, Δ_t), for $i = 1, \cdots, \ell$, increase counter $C_i[h_i(i_t)]$ by Δ_t
  - In the end, as an estimate of x_j, output a median of $\{C_1[h_1(j)], \cdots, C_\ell[h_\ell(j)]\}$.
- Consider any fixed index $j \in [d]$.

- Algorithm: same setup and initialization as before
  - At input (i_t, Δ_t), for $i = 1, \cdots, \ell$, increase counter $C_i[h_i(i_t)]$ by Δ_t
  - In the end, as an estimate of x_j, output a median of $\{C_1[h_1(j)], \cdots, C_\ell[h_\ell(j)]\}$.
- Consider any fixed index $j \in [d]$.
  - Which indices may cause positive error? $P := \{j' : x_{j'} > 0\}$.

- Algorithm: same setup and initialization as before
  - At input (i_t, Δ_t), for $i = 1, \cdots, \ell$, increase counter $C_i[h_i(i_t)]$ by Δ_t
  - In the end, as an estimate of x_j, output a median of $\{C_1[h_1(j)], \cdots, C_\ell[h_\ell(j)]\}$.
- Consider any fixed index $j \in [d]$.
  - Which indices may cause positive error? $P := \{j' : x_{j'} > 0\}$.
  - Similarly, indices that may cause negative error are $N := \{j' : x_{j'} < 0\}$.

- Algorithm: same setup and initialization as before
  - At input (i_t, Δ_t), for $i = 1, \cdots, \ell$, increase counter $C_i[h_i(i_t)]$ by Δ_t
  - In the end, as an estimate of x_j, output a median of $\{C_1[h_1(j)], \cdots, C_\ell[h_\ell(j)]\}$.
- Consider any fixed index $j \in [d]$.
  - Which indices may cause positive error? $P := \{j' : x_{j'} > 0\}$.
  - Similarly, indices that may cause negative error are $N := \{j' : x_{j'} < 0\}$.
- For any counter C_i, the expected error caused by indices in P is $\leq \frac{||x||_1}{k}$.

- Algorithm: same setup and initialization as before
  - At input (i_t, Δ_t), for $i = 1, \cdots, \ell$, increase counter $C_i[h_i(i_t)]$ by Δ_t
  - In the end, as an estimate of x_j, output a median of $\{C_1[h_1(j)], \cdots, C_\ell[h_\ell(j)]\}$.
- Consider any fixed index $j \in [d]$.
  - Which indices may cause positive error? $P := \{j' : x_{j'} > 0\}$.
  - Similarly, indices that may cause negative error are $N := \{j' : x_{j'} < 0\}$.
- For any counter C_i, the expected error caused by indices in P is $\leq \frac{||x||_1}{k}$.
- By Markov inequality, $\mathbf{Pr}[\sum_{j' \in P \setminus \{j\}} x_{j'} \mathbb{1}_{h_i(j)=h_i(j')} \geq \frac{4||x||_1}{k}] \leq \frac{1}{4}$.

- Algorithm: same setup and initialization as before
  - At input (i_t, Δ_t), for $i = 1, \cdots, \ell$, increase counter $C_i[h_i(i_t)]$ by Δ_t
  - In the end, as an estimate of x_j, output a median of $\{C_1[h_1(j)], \cdots, C_\ell[h_\ell(j)]\}$.
- Consider any fixed index $j \in [d]$.
  - Which indices may cause positive error? $P := \{j' : x_{j'} > 0\}$.
  - Similarly, indices that may cause negative error are $N := \{j' : x_{j'} < 0\}$.
- For any counter C_i, the expected error caused by indices in P is $\leq \frac{||x||_1}{k}$.
- By Markov inequality, $\mathbf{Pr}[\sum_{j' \in P \setminus \{j\}} x_{j'} \mathbb{1}_{h_i(j)=h_i(j')} \geq \frac{4||x||_1}{k}] \leq \frac{1}{4}$.
- Similarly, $\mathbf{Pr}[\sum_{j' \in N \setminus \{j\}} |x_{j'}| \mathbb{1}_{h_i(j)=h_i(j')} \geq \frac{4||x||_1}{k}] \leq \frac{1}{4}$.

- Algorithm: same setup and initialization as before
  - At input (i_t, Δ_t), for $i = 1, \cdots, \ell$, increase counter $C_i[h_i(i_t)]$ by Δ_t
  - In the end, as an estimate of x_j, output a median of $\{C_1[h_1(j)], \cdots, C_\ell[h_\ell(j)]\}$.
- Consider any fixed index $j \in [d]$.
  - Which indices may cause positive error? $P \coloneqq \{j' : x_{j'} > 0\}$.
  - Similarly, indices that may cause negative error are $N \coloneqq \{j' : x_{j'} < 0\}$.
- For any counter C_i, the expected error caused by indices in P is $\leq \frac{||x||_1}{k}$.
- By Markov inequality, $\mathbf{Pr}[\sum_{j' \in P \setminus \{j\}} x_{j'} \mathbb{1}_{h_i(j)=h_i(j')} \geq \frac{4||x||_1}{k}] \leq \frac{1}{4}$.
- Similarly, $\mathbf{Pr}[\sum_{j' \in N \setminus \{j\}} |x_{j'}| \mathbb{1}_{h_i(j)=h_i(j')} \geq \frac{4||x||_1}{k}] \leq \frac{1}{4}$.
- Setting $k = \frac{4}{\epsilon}$, $\ell = O(\log d)$, with high probability our output for every coordinate is correct within $\epsilon||x||_1$ additive error.

Count-Sketch

- We can do a bit better to control the error term to within $\epsilon||x||_2$
  - Recall that $||x||_p = \left(\sum_i x_i^p\right)^{1/p}$ decreases with p for $p \in (0, \infty)$.

Count-Sketch

- We can do a bit better to control the error term to within $\epsilon||x||_2$
  - Recall that $||x||_p = \left(\sum_i x_i^p\right)^{1/p}$ decreases with p for $p \in (0, \infty)$.
  - Common bound: $||x||_2 \geq \frac{1}{\sqrt{n}}||x||_1$ by Cauchy-Schwartz

Count-Sketch

- We can do a bit better to control the error term to within $\epsilon||x||_2$
  - Recall that $||x||_p = \left(\sum_i x_i^p\right)^{1/p}$ decreases with p for $p \in (0, \infty)$.
  - Common bound: $||x||_2 \geq \frac{1}{\sqrt{n}}||x||_1$ by Cauchy-Schwartz
- Count-Sketch due to Charikar, Chen, Farach-Colton (2004)
  - Same setup as before, except that now
    - each h_i is drawn from a 2-wise universal hash family;
    - maintain hash functions $g_1, \ldots, g_\ell : [d] \rightarrow \{+1, -1\}$, each drawn independently from a 2-wise universal hash family.

COUNT-SKETCH

- We can do a bit better to control the error term to within $\epsilon||x||_2$
  - Recall that $||x||_p = \left(\sum_i x_i^p\right)^{1/p}$ decreases with p for $p \in (0, \infty)$.
  - Common bound: $||x||_2 \geq \frac{1}{\sqrt{n}}||x||_1$ by Cauchy-Schwartz
- COUNT-SKETCH due to Charikar, Chen, Farach-Colton (2004)
  - Same setup as before, except that now
    - each h_i is drawn from a 2-wise universal hash family;
    - maintain hash functions $g_1, \ldots, g_\ell : [d] \to \{+1, -1\}$, each drawn independently from a 2-wise universal hash family.
  - At input (i_t, Δ_t), for $i = 1, \cdots, \ell$, increase counter $C_i[h_i(i_t)]$ by $g_{i_t}\Delta_t$.
  - In the end, for index $j \in [d]$, output a median M among $g_1(j)C_1[h_1(j)], \cdots, g_\ell(j)C_\ell[h_\ell(j)]$.

Analysis of Count-Sketch

- For any $i \in [\ell]$ and $j \in [d]$. By pairwise independence of $g_i(\cdot)$'s,

$$\mathbf{E}\left[C_i[h_i(j)]g_i(j)\right] = x_j + \mathbf{E}\left[\sum_{j' \neq j} g_i(j)g_i(j')x_{j'}\mathbb{1}_{h_i(j')=h_i(j)}\right] = x_j.$$

Analysis of Count-Sketch

- For any $i \in [\ell]$ and $j \in [d]$. By pairwise independence of $g_i(\cdot)$'s,

$$\mathbf{E}\left[C_i[h_i(j)]g_i(j)\right] = x_j + \mathbf{E}\left[\sum_{j' \neq j} g_i(j)g_i(j')x_{j'}\mathbb{1}_{h_i(j')=h_i(j)}\right] = x_j.$$

- We bound the deviation by Chebyshev inequality:

Analysis of COUNT-SKETCH

- For any $i \in [\ell]$ and $j \in [d]$. By pairwise independence of $g_i(\cdot)$'s,

$$\mathbf{E}\left[C_i[h_i(j)]g_i(j)\right] = x_j + \mathbf{E}\left[\sum_{j' \neq j} g_i(j)g_i(j')x_{j'}\mathbb{1}_{h_i(j')=h_i(j)}\right] = x_j.$$

- We bound the deviation by Chebyshev inequality:

$$\mathrm{Var}\left[C_i[h_i(j)]g_i(j)\right] = \mathrm{Var}\left[\sum_{j' \neq j} g_i(j)g_i(j')x_{j'}\mathbb{1}_{h_i(j')=h_i(j)}\right] \leq \frac{||x||_2^2}{k}.$$

Analysis of Count-Sketch

- For any $i \in [\ell]$ and $j \in [d]$. By pairwise independence of $g_i(\cdot)$'s,

$$\mathbf{E}\left[C_i[h_i(j)]g_i(j)\right] = x_j + \mathbf{E}\left[\sum_{j' \neq j} g_i(j)g_i(j')x_{j'}\mathbb{1}_{h_i(j')=h_i(j)}\right] = x_j.$$

- We bound the deviation by Chebyshev inequality:

$$\mathrm{Var}\left[C_i[h_i(j)]g_i(j)\right] = \mathrm{Var}\left[\sum_{j' \neq j} g_i(j)g_i(j')x_{j'}\mathbb{1}_{h_i(j')=h_i(j)}\right] \leq \frac{||x||_2^2}{k}.$$

$$\mathbf{Pr}\left[|g_i(j)C_i[h_i(j)] - x_j| \geq \epsilon||x||_2\right] \leq \frac{||x||_2^2}{k\epsilon^2||x||_2^2} = \frac{1}{k\epsilon^2}$$

Analysis of Count-Sketch

- For any $i \in [\ell]$ and $j \in [d]$. By pairwise independence of $g_i(\cdot)$'s,

$$\mathbf{E}\left[C_i[h_i(j)]g_i(j)\right] = x_j + \mathbf{E}\left[\sum_{j' \neq j} g_i(j)g_i(j')x_{j'}\mathbb{1}_{h_i(j')=h_i(j)}\right] = x_j.$$

- We bound the deviation by Chebyshev inequality:

$$\mathrm{Var}\left[C_i[h_i(j)]g_i(j)\right] = \mathrm{Var}\left[\sum_{j' \neq j} g_i(j)g_i(j')x_{j'}\mathbb{1}_{h_i(j')=h_i(j)}\right] \leq \frac{||x||_2^2}{k}.$$

$$\mathbf{Pr}\left[|g_i(j)C_i[h_i(j)] - x_j| \geq \epsilon||x||_2\right] \leq \frac{||x||_2^2}{k\epsilon^2||x||_2^2} = \frac{1}{k\epsilon^2}$$

We can take $k = O(\frac{1}{\epsilon^2})$.