Randomized Algorithms, Lecture 11

Jacob Holm (`jaho@di.ku.dk`)

May 28th 2019

Good afternoon.

Today's Lecture

- Streaming Algorithms
  - Basic Streaming Model
  - Deterministic
    - Misra-Gries
  - Turnstile model
  - Randomized Sketching
    - Basic Count Sketch
    - The Median Trick
    - Linear Sketch
    - Count-Min Sketch
  - Summary

Basic Streaming Model

A *stream* is a sequence $\sigma = a_0, a_1, \ldots, a_{n-1} \in [u]$, where n and u are *huge*, that can only be accessed on element at a time, in order.

We have very little space (ideally $\mathcal{O}(\log n + \log u)$ bits) and must answer questions about the part of the stream we have seen so far.

Easy examples:

This lecture: Frequency estimation.
Given $x \in [u]$ compute an estimate $\hat{f}_x$ of the *frequency* $f_x = \left|\{i \in [n] \mid x_i = x\}\right|$.

The notes use m and n for our n and u.
For any integer s, the notes also use the notation $[s]$ to mean $\{1, \ldots, s\}$ rather than $\{0, \ldots, s-1\}$.

Basic Streaming Model

A *stream* is a sequence $\sigma = a_0, a_1, \ldots, a_{n-1} \in [u]$, where n and u are *huge*, that can only be accessed on element at a time, in order.

We have very little space (ideally $\mathcal{O}(\log n + \log u)$ bits) and must answer questions about the part of the stream we have seen so far.

Easy examples:

This lecture: Frequency estimation.
Given $x \in [u]$ compute an estimate $\hat{f}_x$ of the *frequency* $f_x = \left|\{i \in [n] \mid x_i = x\}\right|$.

Basic Streaming Model

A *stream* is a sequence $\sigma = a_0, a_1, \ldots, a_{n-1} \in [u]$, where n and u are *huge*, that can only be accessed on element at a time, in order.

We have very little space (ideally $\mathcal{O}(\log n + \log u)$ bits) and must answer questions about the part of the stream we have seen so far.

Easy examples:

This lecture: Frequency estimation.
Given $x \in [u]$ compute an estimate $\hat{f}_x$ of the *frequency* $f_x = \left|\{i \in [n] \mid x_i = x\}\right|$.

Basic Streaming Model

A *stream* is a sequence $\sigma = a_0, a_1, \ldots, a_{n-1} \in [u]$, where n and u are *huge*, that can only be accessed on element at a time, in order.

We have very little space (ideally $\mathcal{O}(\log n + \log u)$ bits) and must answer questions about the part of the stream we have seen so far.

Easy examples: Minimum/maximum element, number of elements, average element, etc.

This lecture: Frequency estimation.
Given $x \in [u]$ compute an estimate $\hat{f}_x$ of the *frequency* $f_x = \left|\{i \in [n] \mid x_i = x\}\right|$.

Basic Streaming Model

A *stream* is a sequence $\sigma = a_0, a_1, \ldots, a_{n-1} \in [u]$, where n and u are *huge*, that can only be accessed on element at a time, in order.

We have very little space (ideally $\mathcal{O}(\log n + \log u)$ bits) and must answer questions about the part of the stream we have seen so far.

Easy examples: Minimum/maximum element, number of elements, average element, etc.

This lecture: Frequency estimation.
Given $x \in [u]$ compute an estimate $\hat{f}_x$ of the *frequency* $f_x = \left|\{i \in [n] \mid x_i = x\}\right|$.

Note that we don't know beforehand which elements $x \in [u]$ we want the estimate for. If we did, we could just count them directly.

Misra-Gries: Overview

Claim: Let $k \in \mathbb{N}$. Using only $\mathcal{O}(k)$ words of $\mathcal{O}(\log n + \log u)$ bits each, we can maintain values $\hat{f}_x \geq 0$ for all $x \in [u]$ such that $f_x - \frac{n}{k} \leq \hat{f}_x \leq f_x$.

We only explicitly store $\hat{f}_x$ when > 0.

Implicitly initialize $\hat{f}_x \leftarrow 0$ for $x \in [u]$.

When processing $x \in \sigma$, first set $\hat{f}_x \leftarrow \hat{f}_x + 1$, then if the set $A = \{y \in [u] \mid \hat{f}_y > 0\}$ has $|A| \geq k$, set $\hat{f}_y \leftarrow \hat{f}_y - 1$ for $y \in A$.

Misra-Gries: Overview

Claim: Let $k \in \mathbb{N}$. Using only $\mathcal{O}(k)$ words of $\mathcal{O}(\log n + \log u)$ bits each, we can maintain values $\hat{f}_x \geq 0$ for all $x \in [u]$ such that $f_x - \frac{n}{k} \leq \hat{f}_x \leq f_x$.

We only explicitly store $\hat{f}_x$ when > 0.

Implicitly initialize $\hat{f}_x \leftarrow 0$ for $x \in [u]$.

When processing $x \in \sigma$, first set $\hat{f}_x \leftarrow \hat{f}_x + 1$, then if the set $A = \{y \in [u] \mid \hat{f}_y > 0\}$ has $|A| \geq k$, set $\hat{f}_y \leftarrow \hat{f}_y - 1$ for $y \in A$.

Misra-Gries: Overview

Claim: Let $k \in \mathbb{N}$. Using only $\mathcal{O}(k)$ words of $\mathcal{O}(\log n + \log u)$ bits each, we can maintain values $\hat{f}_x \geq 0$ for all $x \in [u]$ such that $f_x - \frac{n}{k} \leq \hat{f}_x \leq f_x$.

We only explicitly store $\hat{f}_x$ when > 0.

Implicitly initialize $\hat{f}_x \leftarrow 0$ for $x \in [u]$.

When processing $x \in \sigma$, first set $\hat{f}_x \leftarrow \hat{f}_x + 1$, then if the set $A = \{y \in [u] \mid \hat{f}_y > 0\}$ has $|A| \geq k$, set $\hat{f}_y \leftarrow \hat{f}_y - 1$ for $y \in A$.

Misra-Gries: Overview

Claim: Let $k \in \mathbb{N}$. Using only $\mathcal{O}(k)$ words of $\mathcal{O}(\log n + \log u)$ bits each, we can maintain values $\hat{f}_x \geq 0$ for all $x \in [u]$ such that $f_x - \frac{n}{k} \leq \hat{f}_x \leq f_x$.

We only explicitly store $\hat{f}_x$ when > 0.

Implicitly initialize $\hat{f}_x \leftarrow 0$ for $x \in [u]$.

When processing $x \in \sigma$, first set $\hat{f}_x \leftarrow \hat{f}_x + 1$, then if the set $A = \{y \in [u] \mid \hat{f}_y > 0\}$ has $|A| \geq k$, set $\hat{f}_y \leftarrow \hat{f}_y - 1$ for $y \in A$.

Misra-Gries: Overview

Claim: Let $k \in \mathbb{N}$. Using only $\mathcal{O}(k)$ words of $\mathcal{O}(\log n + \log u)$ bits each, we can maintain values $\hat{f}_x \geq 0$ for all $x \in [u]$ such that $f_x - \frac{n}{k} \leq \hat{f}_x \leq f_x$.

We only explicitly store $\hat{f}_x$ when > 0.

Implicitly initialize $\hat{f}_x \leftarrow 0$ for $x \in [u]$.

When processing $x \in \sigma$, first set $\hat{f}_x \leftarrow \hat{f}_x + 1$, then if the set $A = \{y \in [u] \mid \hat{f}_y > 0\}$ has $|A| \geq k$, set $\hat{f}_y \leftarrow \hat{f}_y - 1$ for $y \in A$.

Misra-Gries: Pseudocode

function MG-INITIALIZE()
    $A \leftarrow \{\}$ $\triangleright$ Implicitly set $\hat{f}_x \leftarrow 0$ for $x \in [u]$

function MG-PROCESS(x)
    $\triangleright$ $\hat{f}_x \leftarrow \hat{f}_x + 1$
    if $x \notin A$ **then**
        $A[x] \leftarrow 1$
    else
        $A[x] \leftarrow A[x] + 1$
    if $|A| \geq k$ **then**
        for $y \in A$ **do**
            $\triangleright$ $\hat{f}_y \leftarrow \hat{f}_y - 1$
            if $A[y] = 1$ **then**
                delete $A[y]$
            else
                $A[y] \leftarrow A[y] - 1$

We use the Python notation here, so $\{\}$ is the empty *dictionary*, aka *associative array*.
We use the notation $x \in A$ to denote that A contains a value for x, and if $x \in A$ we use $A[x]$ to denote that value.

Misra-Gries: Pseudocode

function MG-INITIALIZE()
    $A \leftarrow \{\}$ $\triangleright$ Implicitly set $\hat{f}_x \leftarrow 0$ for $x \in [u]$

function MG-PROCESS(x)
    $\triangleright$ $\hat{f}_x \leftarrow \hat{f}_x + 1$
    if $x \notin A$ **then**
        $A[x] \leftarrow 1$
    else
        $A[x] \leftarrow A[x] + 1$
    if $|A| \geq k$ **then**
        for $y \in A$ **do**
            $\triangleright$ $\hat{f}_y \leftarrow \hat{f}_y - 1$
            if $A[y] = 1$ **then**
                delete $A[y]$
            else
                $A[y] \leftarrow A[y] - 1$

Misra-Gries: Pseudocode

function MG-INITIALIZE()
  $A \leftarrow \{\}$ ▷ Implicitly set $\hat{f}_x \leftarrow 0$ for $x \in [u]$

function MG-PROCESS(x)
  ▷ $\hat{f}_x \leftarrow \hat{f}_x + 1$
  if $x \notin A$ **then**
    $A[x] \leftarrow 1$
  else
    $A[x] \leftarrow A[x] + 1$
  if $|A| \geq k$ **then**
    for $y \in A$ **do**
      ▷ $\hat{f}_y \leftarrow \hat{f}_y - 1$
      if $A[y] = 1$ **then**
        delete $A[y]$
      else
        $A[y] \leftarrow A[y] - 1$

Misra-Gries: Pseudocode

```
function MG-INITIALIZE()
    A ← {}                          ▷ Implicitly set f̂_x ← 0 for x ∈ [u]
function MG-PROCESS(x)
    ▷ f̂_x ← f̂_x + 1
    if x ∉ A then
        A[x] ← 1
    else
        A[x] ← A[x] + 1
    if |A| ≥ k then
        for y ∈ A do
            ▷ f̂_y ← f̂_y − 1
            if A[y] = 1 then
                delete A[y]
            else
                A[y] ← A[y] − 1
```

Misra-Gries: Pseudocode

```
function MG-INITIALIZE()
    A ← {}                          ▷ Implicitly set f̂_x ← 0 for x ∈ [u]
function MG-PROCESS(x)
    ▷ f̂_x ← f̂_x + 1
    if x ∉ A then
        A[x] ← 1
    else
        A[x] ← A[x] + 1
    if |A| ≥ k then
        for y ∈ A do
            ▷ f̂_y ← f̂_y − 1
            if A[y] = 1 then
                delete A[y]
            else
                A[y] ← A[y] − 1
```

Misra-Gries: Pseudocode

function MG-INITIALIZE()
    $A \leftarrow \{\}$ $\triangleright$ Implicitly set $\hat{f}_x \leftarrow 0$ for $x \in [u]$

function MG-PROCESS(x)
    $\triangleright$ $\hat{f}_x \leftarrow \hat{f}_x + 1$
    if $x \notin A$ **then**
        $A[x] \leftarrow 1$
    else
        $A[x] \leftarrow A[x] + 1$
    if $|A| \geq k$ **then**
        for $y \in A$ **do**
            $\triangleright$ $\hat{f}_y \leftarrow \hat{f}_y - 1$
            if $A[y] = 1$ **then**
                delete $A[y]$
            else
                $A[y] \leftarrow A[y] - 1$

Misra-Gries: Analysis

Theorem

After processing n elements, we have
$f_x - \frac{n}{k} \leq \hat{f}_x \leq f_x$ *for all* $x \in [u]$.

Proof.

The algorithm starts with $\hat{f}_x = f_x = 0$ and only increases $\hat{f}_x$ when f_x increases, so clearly $\hat{f}_x \leq f_x$. Each time $\hat{f}_x$ is decreased, x is part of a set $A \subseteq [u]$ of size $\geq k$ where every $y \in A$ has $\hat{f}_y > 0$ and all are decreased at the same time. The total number of rounds of decreases is therefore at most $\frac{n}{k}$. In particular, the total number of times that $\hat{f}_x$ is decreased is at most $\frac{n}{k}$. $\square$

Misra-Gries: Analysis

Theorem

After processing n elements, we have
$f_x - \frac{n}{k} \leq \hat{f}_x \leq f_x$ *for all* $x \in [u]$.

Proof.

The algorithm starts with $\hat{f}_x = f_x = 0$ and only increases $\hat{f}_x$ when f_x increases, so clearly $\hat{f}_x \leq f_x$.
Each time $\hat{f}_x$ is decreased, x is part of a set $A \subseteq [u]$ of size $\geq k$ where every $y \in A$ has $\hat{f}_y > 0$ and all are decreased at the same time. The total number of rounds of decreases is therefore at most $\frac{n}{k}$. In particular, the total number of times that $\hat{f}_x$ is decreased is at most $\frac{n}{k}$. $\square$

Misra-Gries: Analysis

Theorem

After processing n elements, we have
$f_x - \frac{n}{k} \leq \hat{f}_x \leq f_x$ *for all* $x \in [u]$.

Proof.

The algorithm starts with $\hat{f}_x = f_x = 0$ and only increases $\hat{f}_x$ when f_x increases, so clearly $\hat{f}_x \leq f_x$. Each time $\hat{f}_x$ is decreased, x is part of a set $A \subseteq [u]$ of size $\geq k$ where every $y \in A$ has $\hat{f}_y > 0$ and all are decreased at the same time. The total number of rounds of decreases is therefore at most $\frac{n}{k}$. In particular, the total number of times that $\hat{f}_x$ is decreased is at most $\frac{n}{k}$. $\square$

Misra-Gries: Analysis

Theorem

After processing n elements, we have
$f_x - \frac{n}{k} \leq \hat{f}_x \leq f_x$ *for all* $x \in [u]$.

Proof.

The algorithm starts with $\hat{f}_x = f_x = 0$ and only increases $\hat{f}_x$ when f_x increases, so clearly $\hat{f}_x \leq f_x$. Each time $\hat{f}_x$ is decreased, x is part of a set $A \subseteq [u]$ of size $\geq k$ where every $y \in A$ has $\hat{f}_y > 0$ and all are decreased at the same time. The total number of rounds of decreases is therefore at most $\frac{n}{k}$. In particular, the total number of times that $\hat{f}_x$ is decreased is at most $\frac{n}{k}$. □

Misra-Gries: Analysis

Theorem

After processing n elements, we have
$f_x - \frac{n}{k} \leq \hat{f}_x \leq f_x$ *for all* $x \in [u]$.

Proof.

The algorithm starts with $\hat{f}_x = f_x = 0$ and only increases $\hat{f}_x$ when f_x increases, so clearly $\hat{f}_x \leq f_x$. Each time $\hat{f}_x$ is decreased, x is part of a set $A \subseteq [u]$ of size $\geq k$ where every $y \in A$ has $\hat{f}_y > 0$ and all are decreased at the same time. The total number of rounds of decreases is therefore at most $\frac{n}{k}$. In particular, the total number of times that $\hat{f}_x$ is decreased is at most $\frac{n}{k}$. $\square$

Turnstile model

Suppose now that our stream of data consists of a sequence of n *pairs* $\sigma = (x_0, \Delta_0), \ldots, (x_{n-1}, \Delta_{n-1}) \in [n] \times \{-u, \ldots, u\}$.

For each $x \in [n]$, let $I_x := \{i \in [n] \mid x_i = x\}$ and define the frequency of x as $f_x := \sum_{i \in I_x} \Delta_i$.

This called the *turnstile* model. If $f_x \geq 0$ for all $x \in [n]$ at all times, it is called the *strict turnstile* model. If $\Delta_i > 0$ for all $i \in [n]$, it is called the *cash register model* or *insertion model*.

Given x still want to compute an estimate $\hat{f}_x$ of f_x.

However, no deterministic algorithm can do frequency estimation in the turnstile model using $\mathcal{O}(\log n + \log u)$ bits.

Next, we'll see a randomized algorithm for frequency estimation in the turnstile model.

Turnstile model

Suppose now that our stream of data consists of a sequence of *n pairs* $\sigma = (x_0, \Delta_0), \ldots, (x_{n-1}, \Delta_{n-1}) \in [n] \times \{-u, \ldots, u\}$.

For each $x \in [n]$, let $I_x := \{i \in [n] \mid x_i = x\}$ and define the frequency of x as $f_x := \sum_{i \in I_x} \Delta_i$.

This called the *turnstile* model. If $f_x \geq 0$ for all $x \in [n]$ at all times, it is called the *strict turnstile* model. If $\Delta_i > 0$ for all $i \in [n]$, it is called the *cash register model* or *insertion model*.

Given x still want to compute an estimate $\hat{f}_x$ of f_x.

However, no deterministic algorithm can do frequency estimation in the turnstile model using $\mathcal{O}(\log n + \log u)$ bits.

Next, we'll see a randomized algorithm for frequency estimation in the turnstile model.

Turnstile model

Suppose now that our stream of data consists of a sequence of n *pairs* $\sigma = (x_0, \Delta_0), \dots, (x_{n-1}, \Delta_{n-1}) \in [n] \times \{-u, \dots, u\}$.

For each $x \in [n]$, let $I_x := \{i \in [n] \mid x_i = x\}$ and define the frequency of x as $f_x := \sum_{i \in I_x} \Delta_i$.

This called the *turnstile* model. If $f_x \geq 0$ for all $x \in [n]$ at all times, it is called the *strict turnstile* model. If $\Delta_i > 0$ for all $i \in [n]$, it is called the *cash register model* or *insertion model*.

Given x still want to compute an estimate $\hat{f}_x$ of f_x.

However, no deterministic algorithm can do frequency estimation in the turnstile model using $\mathcal{O}(\log n + \log u)$ bits.

Next, we'll see a randomized algorithm for frequency estimation in the turnstile model.

Turnstile model

Suppose now that our stream of data consists of a sequence of n *pairs* $\sigma = (x_0, \Delta_0), \dots, (x_{n-1}, \Delta_{n-1}) \in [n] \times \{-u, \dots, u\}$.

For each $x \in [n]$, let $I_x := \{i \in [n] \mid x_i = x\}$ and define the frequency of x as $f_x := \sum_{i \in I_x} \Delta_i$.

This called the *turnstile* model. If $f_x \geq 0$ for all $x \in [n]$ at all times, it is called the *strict turnstile* model. If $\Delta_i > 0$ for all $i \in [n]$, it is called the *cash register model* or *insertion model*.

Given x still want to compute an estimate $\hat{f}_x$ of f_x.

However, no deterministic algorithm can do frequency estimation in the turnstile model using $\mathcal{O}(\log n + \log u)$ bits.

Next, we'll see a randomized algorithm for frequency estimation in the turnstile model.

Turnstile model

Suppose now that our stream of data consists of a sequence of n *pairs* $\sigma = (x_0, \Delta_0), \ldots, (x_{n-1}, \Delta_{n-1}) \in [n] \times \{-u, \ldots, u\}$.

For each $x \in [n]$, let $I_x := \{i \in [n] \mid x_i = x\}$ and define the frequency of x as $f_x := \sum_{i \in I_x} \Delta_i$.

This called the *turnstile* model. If $f_x \geq 0$ for all $x \in [n]$ at all times, it is called the *strict turnstile* model. If $\Delta_i > 0$ for all $i \in [n]$, it is called the *cash register model* or *insertion model*.

Given x still want to compute an estimate $\hat{f}_x$ of f_x.

However, no deterministic algorithm can do frequency estimation in the turnstile model using $\mathcal{O}(\log n + \log u)$ bits.

Next, we'll see a randomized algorithm for frequency estimation in the turnstile model.

Turnstile model

Suppose now that our stream of data consists of a sequence of *n pairs* $\sigma = (x_0, \Delta_0), \ldots, (x_{n-1}, \Delta_{n-1}) \in [n] \times \{-u, \ldots, u\}$.

For each $x \in [n]$, let $I_x := \{i \in [n] \mid x_i = x\}$ and define the frequency of x as $f_x := \sum_{i \in I_x} \Delta_i$.

This called the *turnstile* model. If $f_x \geq 0$ for all $x \in [n]$ at all times, it is called the *strict turnstile* model. If $\Delta_i > 0$ for all $i \in [n]$, it is called the *cash register model* or *insertion model*.

Given x still want to compute an estimate $\hat{f}_x$ of f_x.

However, no deterministic algorithm can do frequency estimation in the turnstile model using $\mathcal{O}(\log n + \log u)$ bits.

Next, we'll see a randomized algorithm for frequency estimation in the turnstile model.

Turnstile model

Suppose now that our stream of data consists of a sequence of n *pairs* $\sigma = (x_0, \Delta_0), \ldots, (x_{n-1}, \Delta_{n-1}) \in [n] \times \{-u, \ldots, u\}$.

For each $x \in [n]$, let $I_x := \{i \in [n] \mid x_i = x\}$ and define the frequency of x as $f_x := \sum_{i \in I_x} \Delta_i$.

This called the *turnstile* model. If $f_x \geq 0$ for all $x \in [n]$ at all times, it is called the *strict turnstile* model. If $\Delta_i > 0$ for all $i \in [n]$, it is called the *cash register model* or *insertion model*.

Given x still want to compute an estimate $\hat{f}_x$ of f_x.

However, no deterministic algorithm can do frequency estimation in the turnstile model using $\mathcal{O}(\log n + \log u)$ bits.

Next, we'll see a randomized algorithm for frequency estimation in the turnstile model.

Misra-Gries *can* be extended to the cash-register model, by interpreting each (x_i, Δ_i) as a sequence of Δ_i copies of x_i.

Turnstile model

Suppose now that our stream of data consists of a sequence of n *pairs* $\sigma = (x_0, \Delta_0), \ldots, (x_{n-1}, \Delta_{n-1}) \in [n] \times \{-u, \ldots, u\}$.

For each $x \in [n]$, let $I_x := \{i \in [n] \mid x_i = x\}$ and define the frequency of x as $f_x := \sum_{i \in I_x} \Delta_i$.

This called the *turnstile* model. If $f_x \geq 0$ for all $x \in [n]$ at all times, it is called the *strict turnstile* model. If $\Delta_i > 0$ for all $i \in [n]$, it is called the *cash register model* or *insertion model*.

Given x still want to compute an estimate $\hat{f}_x$ of f_x.

However, no deterministic algorithm can do frequency estimation in the turnstile model using $\mathcal{O}(\log n + \log u)$ bits.

Next, we'll see a randomized algorithm for frequency estimation in the turnstile model.

Basic Count Sketch: Pseudocode

function BCS-INITIALIZE(n, ε)
  $k \leftarrow \left\lceil \frac{3}{\varepsilon^2} \right\rceil$
  $C[0, \ldots, k-1] \leftarrow 0$
  Pick 2-independent $h : [n] \rightarrow [k]$
  Pick 2-independent $s : [n] \rightarrow \{-1, +1\}$

function BCS-PROCESS(x, Δ)
  $C[h(x)] \leftarrow C[h(x)] + s(x) \cdot \Delta$

function BCS-QUERY(x)
  return $s(x) \cdot C[h(x)]$ ▷ Returns $\hat{f}_x$

Basic Count Sketch: Pseudocode

```
function BCS-INITIALIZE(n, ε)
    k ← ⌈3/ε²⌉
    C[0, ..., k − 1] ← 0
    Pick 2-independent h : [n] → [k]
    Pick 2-independent s : [n] → {−1, +1}

function BCS-PROCESS(x, Δ)
    C[h(x)] ← C[h(x)] + s(x) · Δ

function BCS-QUERY(x)
    return s(x) · C[h(x)]                ▷ Returns f̂_x
```

Basic Count Sketch: Pseudocode

```
function BCS-INITIALIZE(n, ε)
    k ← ⌈3/ε²⌉
    C[0, ..., k − 1] ← 0
    Pick 2-independent h : [n] → [k]
    Pick 2-independent s : [n] → {−1, +1}

function BCS-PROCESS(x, Δ)
    C[h(x)] ← C[h(x)] + s(x) · Δ

function BCS-QUERY(x)
    return s(x) · C[h(x)]                ▷ Returns f̂_x
```

Basic Count Sketch: Pseudocode

function BCS-INITIALIZE(n, ε)

  $k \leftarrow \left\lceil \frac{3}{\varepsilon^2} \right\rceil$

  $C[0, \ldots, k-1] \leftarrow 0$

  Pick 2-independent $h : [n] \to [k]$

  Pick 2-independent $s : [n] \to \{-1, +1\}$

function BCS-PROCESS(x, Δ)

  $C[h(x)] \leftarrow C[h(x)] + s(x) \cdot \Delta$

function BCS-QUERY(x)

  return $s(x) \cdot C[h(x)]$ $\triangleright$ Returns $\hat{f}_x$

Note that it is also important that s and h are independent of each other. This is implied in the way they are chosen.

In particular, do *not* fall for the temptation to just chose a single hash function $f : [n] \to [2k]$ and defining $h(x) = \lfloor \frac{f(x)}{2} \rfloor$ and $s(x) = 2(f(x) \bmod 2) - 1$.

With this definition, each of h, s would still be 2-independent, but they would not be independent of each other.

Basic Count Sketch: Pseudocode

function BCS-INITIALIZE(n, ε)

  $k \leftarrow \left\lceil \frac{3}{\varepsilon^2} \right\rceil$

  $C[0, \ldots, k-1] \leftarrow 0$

  Pick 2-independent $h : [n] \to [k]$

  Pick 2-independent $s : [n] \to \{-1, +1\}$

function BCS-PROCESS(x, Δ)

  $C[h(x)] \leftarrow C[h(x)] + s(x) \cdot \Delta$

function BCS-QUERY(x)

  return $s(x) \cdot C[h(x)]$ $\triangleright$ Returns $\hat{f}_x$

Basic Count Sketch: Pseudocode

```
function BCS-INITIALIZE(n, ε)
    k ← ⌈3/ε²⌉
    C[0, ..., k − 1] ← 0
    Pick 2-independent h : [n] → [k]
    Pick 2-independent s : [n] → {−1, +1}

function BCS-PROCESS(x, Δ)
    C[h(x)] ← C[h(x)] + s(x) · Δ

function BCS-QUERY(x)
    return s(x) · C[h(x)]                ▷ Returns f̂_x
```

Basic Count Sketch: Analysis

Lemma

For $x \in [n]$, let $\hat{f}_x = s(x) \cdot C[h(x)]$. Then $\mathbb{E}[\hat{f}_x] = f_x$.

Thus, $\hat{f}_x$ is an *unbiased estimator* for f_x.

Proof.

$$\begin{aligned}
\hat{f}_x &= s(x) \cdot C[h(x)] \\
&= s(x) \cdot \sum_{y \in [n]} f_y s(y) B_{xy} \text{ where } B_{xy} = [h(y) = h(x)] \\
&= \sum_{y \in [n]} f_y s(x) s(y) B_{xy} = f_x + \sum_{y \neq x} f_y s(x) s(y) B_{xy} \\
\mathbb{E}[\hat{f}_x] &= f_x + \sum_{y \neq x} f_y \, \mathbb{E}[s(x) s(y) B_{xy}] \\
&= f_x + \sum_{y \neq x} f_y \, \mathbb{E}[s(x)] \, \mathbb{E}[s(y)] \, \mathbb{E}[B_{xy}] \quad = \quad f_x \qquad \square
\end{aligned}$$

Basic Count Sketch: Analysis

Lemma

For $x \in [n]$, let $\hat{f}_x = s(x) \cdot C[h(x)]$. Then $\mathbb{E}[\hat{f}_x] = f_x$.

Thus, $\hat{f}_x$ is an *unbiased estimator* for f_x.

Proof.

$$\begin{aligned}
\hat{f}_x &= s(x) \cdot C[h(x)] \\
&= s(x) \cdot \sum_{y \in [n]} f_y s(y) B_{xy} \text{ where } B_{xy} = [h(y) = h(x)] \\
&= \sum_{y \in [n]} f_y s(x) s(y) B_{xy} = f_x + \sum_{y \neq x} f_y s(x) s(y) B_{xy} \\
\mathbb{E}[\hat{f}_x] &= f_x + \sum_{y \neq x} f_y \, \mathbb{E}[s(x) s(y) B_{xy}] \\
&= f_x + \sum_{y \neq x} f_y \, \mathbb{E}[s(x)] \, \mathbb{E}[s(y)] \, \mathbb{E}[B_{xy}] \quad = \quad f_x \qquad \square
\end{aligned}$$

Basic Count Sketch: Analysis

Lemma

For $x \in [n]$, let $\hat{f}_x = s(x) \cdot C[h(x)]$. Then $\mathbb{E}[\hat{f}_x] = f_x$.

Thus, $\hat{f}_x$ is an *unbiased estimator* for f_x.

Proof.

$$\hat{f}_x = s(x) \cdot C[h(x)]$$

$$= s(x) \cdot \sum_{y \in [n]} f_y s(y) B_{xy} \text{ where } B_{xy} = [h(y) = h(x)]$$

$$= \sum_{y \in [n]} f_y s(x) s(y) B_{xy} = f_x + \sum_{y \neq x} f_y s(x) s(y) B_{xy}$$

$$\mathbb{E}[\hat{f}_x] = f_x + \sum_{y \neq x} f_y \, \mathbb{E}[s(x) s(y) B_{xy}]$$

$$= f_x + \sum_{y \neq x} f_y \, \mathbb{E}[s(x)] \, \mathbb{E}[s(y)] \, \mathbb{E}[B_{xy}] \quad = \quad f_x \qquad \square$$

Basic Count Sketch: Analysis

Lemma

For $x \in [n]$, let $\hat{f}_x = s(x) \cdot C[h(x)]$. Then $\mathbb{E}[\hat{f}_x] = f_x$.

Thus, $\hat{f}_x$ is an *unbiased estimator* for f_x.

Proof.

$$\hat{f}_x = s(x) \cdot C[h(x)]$$
$$= s(x) \cdot \sum_{y \in [n]} f_y s(y) B_{xy} \text{ where } B_{xy} = [h(y) = h(x)]$$
$$= \sum_{y \in [n]} f_y s(x) s(y) B_{xy} = f_x + \sum_{y \neq x} f_y s(x) s(y) B_{xy}$$
$$\mathbb{E}[\hat{f}_x] = f_x + \sum_{y \neq x} f_y \, \mathbb{E}[s(x) s(y) B_{xy}]$$
$$= f_x + \sum_{y \neq x} f_y \, \mathbb{E}[s(x)] \, \mathbb{E}[s(y)] \, \mathbb{E}[B_{xy}] \quad = \quad f_x \qquad \square$$

By definition,

$$C[h(x)] = \sum_{\substack{j \in [n] \\ h(x_j) = h(x)}} s(x_j) \Delta_j$$
$$= \sum_{\substack{y \in [n] \\ h(y) = h(x)}} \sum_{j \in I_y} s(y) \Delta_j$$
$$= \sum_{\substack{y \in [n] \\ h(y) = h(x)}} s(y) f_y$$
$$= \sum_{y \in [n]} f_y s(y) [h(y) = h(x)]$$
$$= \sum_{y \in [n]} f_y s(y) B_{xy}$$

Basic Count Sketch: Analysis

Lemma

For $x \in [n]$, let $\hat{f}_x = s(x) \cdot C[h(x)]$. Then $\mathbb{E}[\hat{f}_x] = f_x$.

Thus, $\hat{f}_x$ is an *unbiased estimator* for f_x.

Proof.

$$\begin{aligned}
\hat{f}_x &= s(x) \cdot C[h(x)] \\
&= s(x) \cdot \sum_{y \in [n]} f_y s(y) B_{xy} \text{ where } B_{xy} = [h(y) = h(x)] \\
&= \sum_{y \in [n]} f_y s(x) s(y) B_{xy} = f_x + \sum_{y \neq x} f_y s(x) s(y) B_{xy}
\end{aligned}$$

$$\begin{aligned}
\mathbb{E}[\hat{f}_x] &= f_x + \sum_{y \neq x} f_y \, \mathbb{E}[s(x) s(y) B_{xy}] \\
&= f_x + \sum_{y \neq x} f_y \, \mathbb{E}[s(x)] \, \mathbb{E}[s(y)] \, \mathbb{E}[B_{xy}] \quad = \quad f_x \qquad \square
\end{aligned}$$

Basic Count Sketch: Analysis

Lemma

For $x \in [n]$, let $\hat{f}_x = s(x) \cdot C[h(x)]$. Then $\mathbb{E}[\hat{f}_x] = f_x$.

Thus, $\hat{f}_x$ is an *unbiased estimator* for f_x.

Proof.

$$\begin{aligned}
\hat{f}_x &= s(x) \cdot C[h(x)] \\
&= s(x) \cdot \sum_{y \in [n]} f_y s(y) B_{xy} \text{ where } B_{xy} = [h(y) = h(x)] \\
&= \sum_{y \in [n]} f_y s(x) s(y) B_{xy} = f_x + \sum_{y \neq x} f_y s(x) s(y) B_{xy}
\end{aligned}$$

$$\begin{aligned}
\mathbb{E}[\hat{f}_x] &= f_x + \sum_{y \neq x} f_y \, \mathbb{E}[s(x)s(y)B_{xy}] \\
&= f_x + \sum_{y \neq x} f_y \, \mathbb{E}[s(x)] \, \mathbb{E}[s(y)] \, \mathbb{E}[B_{xy}] \quad = \quad f_x \qquad \square
\end{aligned}$$

The term for $y = x$ is special, because then $s(x) = s(y) \in \{-1, 1\}$ so $s(x)s(y) = 1$ and

$$f_y \cdot s(x) \cdot s(y) \cdot B_{xy} = f_x \cdot 1 \cdot 1 = f_x$$

Basic Count Sketch: Analysis

Lemma

For $x \in [n]$, let $\hat{f}_x = s(x) \cdot C[h(x)]$. Then $\mathbb{E}[\hat{f}_x] = f_x$.

Thus, $\hat{f}_x$ is an *unbiased estimator* for f_x.

Proof.

$$\begin{aligned}
\hat{f}_x &= s(x) \cdot C[h(x)] \\
&= s(x) \cdot \sum_{y \in [n]} f_y s(y) B_{xy} \text{ where } B_{xy} = [h(y) = h(x)] \\
&= \sum_{y \in [n]} f_y s(x) s(y) B_{xy} = f_x + \sum_{y \neq x} f_y s(x) s(y) B_{xy} \\
\mathbb{E}[\hat{f}_x] &= f_x + \sum_{y \neq x} f_y \, \mathbb{E}[s(x) s(y) B_{xy}] \\
&= f_x + \sum_{y \neq x} f_y \, \mathbb{E}[s(x)] \, \mathbb{E}[s(y)] \, \mathbb{E}[B_{xy}] \quad = \quad f_x \qquad \square
\end{aligned}$$

Take the expected value on both sides, and use linearity of expectation.

Basic Count Sketch: Analysis

Lemma

For $x \in [n]$, let $\hat{f}_x = s(x) \cdot C[h(x)]$. Then $\mathbb{E}[\hat{f}_x] = f_x$.

Thus, $\hat{f}_x$ is an *unbiased estimator* for f_x.

Proof.

$$\begin{aligned}
\hat{f}_x &= s(x) \cdot C[h(x)] \\
&= s(x) \cdot \sum_{y \in [n]} f_y s(y) B_{xy} \text{ where } B_{xy} = [h(y) = h(x)] \\
&= \sum_{y \in [n]} f_y s(x) s(y) B_{xy} = f_x + \sum_{y \neq x} f_y s(x) s(y) B_{xy} \\
\mathbb{E}[\hat{f}_x] &= f_x + \sum_{y \neq x} f_y \, \mathbb{E}[s(x) s(y) B_{xy}] \\
&= f_x + \sum_{y \neq x} f_y \, \mathbb{E}[s(x)] \, \mathbb{E}[s(y)] \, \mathbb{E}[B_{xy}] \quad = \quad f_x \qquad \square
\end{aligned}$$

Since s is 2-independent, and s and h are independent of each other, the expectation of this product is the product of the 3 expectations.

Basic Count Sketch: Analysis

Lemma

For $x \in [n]$, let $\hat{f}_x = s(x) \cdot C[h(x)]$. Then $\mathbb{E}[\hat{f}_x] = f_x$.

Thus, $\hat{f}_x$ is an *unbiased estimator* for f_x.

Proof.

$$\begin{aligned}
\hat{f}_x &= s(x) \cdot C[h(x)] \\
&= s(x) \cdot \sum_{y \in [n]} f_y s(y) B_{xy} \text{ where } B_{xy} = [h(y) = h(x)] \\
&= \sum_{y \in [n]} f_y s(x) s(y) B_{xy} = f_x + \sum_{y \neq x} f_y s(x) s(y) B_{xy} \\
\mathbb{E}[\hat{f}_x] &= f_x + \sum_{y \neq x} f_y \, \mathbb{E}[s(x) s(y) B_{xy}] \\
&= f_x + \sum_{y \neq x} f_y \, \mathbb{E}[s(x)] \, \mathbb{E}[s(y)] \, \mathbb{E}[B_{xy}] \quad = \quad f_x \qquad \square
\end{aligned}$$

Since $s : [n] \to \{-1, 1\}$ is 2-independent, by definition $s(x)$ is uniform in $\{-1, 1\}$. Thus, $\mathbb{E}[s(x)] = 0$.

Basic Count Sketch: Analysis

For an n-dimensional vector $\mathbf{f}$ and $i \in [n]$ define $\mathbf{f}_{-i}$ to be the $(n-1)$-dimensional vector obtained by dropping index i. Then $\|\mathbf{f}_{-x}\|_2^2 = \sum_{y \neq x} f_y^2 = \|\mathbf{f}\|_2^2 - f_x^2$.

Lemma

$\mathrm{Var}[\hat{f}_x] = \mathbb{E}[(\hat{f}_x - f_x)^2] = \frac{\|\mathbf{f}_{-x}\|_2^2}{k}$

Proof.

$$
\begin{aligned}
\mathbb{E}[(\hat{f}_x - f_x)^2] &= \mathbb{E}\Big[\Big(\sum_{y \neq x} f_y s(x) s(y) B_{xy}\Big)^2\Big] \\
&= \sum_{y \neq x} \sum_{z \neq x} \mathbb{E}\Big[f_y s(x) s(y) B_{xy} f_z s(x) s(z) B_{xz}\Big] \\
&= \sum_{y \neq x} f_y^2 \, \mathbb{E}[B_{xy}^2] + 0 \\
&= \sum_{y \neq x} f_y^2 \cdot \frac{1}{k} = \frac{\|\mathbf{f}_{-x}\|_2^2}{k} \qquad \square
\end{aligned}
$$

Basic Count Sketch: Analysis

For an n-dimensional vector $\mathbf{f}$ and $i \in [n]$ define $\mathbf{f}_{-i}$ to be the $(n-1)$-dimensional vector obtained by dropping index i. Then $\|\mathbf{f}_{-x}\|_2^2 = \sum_{y \neq x} f_y^2 = \|\mathbf{f}\|_2^2 - f_x^2$.

Lemma

$\mathrm{Var}[\hat{f}_x] = \mathbb{E}[(\hat{f}_x - f_x)^2] = \frac{\|\mathbf{f}_{-x}\|_2^2}{k}$

Proof.

$$\begin{aligned}
\mathbb{E}[(\hat{f}_x - f_x)^2] &= \mathbb{E}\Big[\Big(\sum_{y \neq x} f_y s(x) s(y) B_{xy}\Big)^2\Big] \\
&= \sum_{y \neq x} \sum_{z \neq x} \mathbb{E}\Big[f_y s(x) s(y) B_{xy} f_z s(x) s(z) B_{xz}\Big] \\
&= \sum_{y \neq x} f_y^2 \, \mathbb{E}[B_{xy}^2] + 0 \\
&= \sum_{y \neq x} f_y^2 \cdot \frac{1}{k} = \frac{\|\mathbf{f}_{-x}\|_2^2}{k}
\end{aligned}$$

$\square$

Basic Count Sketch: Analysis

For an n-dimensional vector $\mathbf{f}$ and $i \in [n]$ define $\mathbf{f}_{-i}$ to be the $(n-1)$-dimensional vector obtained by dropping index i. Then $\|\mathbf{f}_{-x}\|_2^2 = \sum_{y \neq x} f_y^2 = \|\mathbf{f}\|_2^2 - f_x^2$.

Lemma

$\mathrm{Var}[\hat{f}_x] = \mathbb{E}[(\hat{f}_x - f_x)^2] = \frac{\|\mathbf{f}_{-x}\|_2^2}{k}$

Proof.

$$
\begin{aligned}
\mathbb{E}[(\hat{f}_x - f_x)^2] &= \mathbb{E}\Big[\Big(\sum_{y \neq x} f_y s(x) s(y) B_{xy}\Big)^2\Big] \\
&= \sum_{y \neq x} \sum_{z \neq x} \mathbb{E}\Big[f_y s(x) s(y) B_{xy} f_z s(x) s(z) B_{xz}\Big] \\
&= \sum_{y \neq x} f_y^2\, \mathbb{E}[B_{xy}^2] + 0 \\
&= \sum_{y \neq x} f_y^2 \cdot \frac{1}{k} = \frac{\|\mathbf{f}_{-x}\|_2^2}{k} \qquad \square
\end{aligned}
$$

Basic Count Sketch: Analysis

For an n-dimensional vector $\mathbf{f}$ and $i \in [n]$ define $\mathbf{f}_{-i}$ to be the $(n-1)$-dimensional vector obtained by dropping index i. Then $\|\mathbf{f}_{-x}\|_2^2 = \sum_{y \neq x} f_y^2 = \|\mathbf{f}\|_2^2 - f_x^2$.

Lemma

$\mathrm{Var}[\hat{f}_x] = \mathbb{E}[(\hat{f}_x - f_x)^2] = \frac{\|\mathbf{f}_{-x}\|_2^2}{k}$

Proof.

$$\mathbb{E}[(\hat{f}_x - f_x)^2] = \mathbb{E}\Big[\Big(\sum_{y \neq x} f_y s(x) s(y) B_{xy}\Big)^2\Big]$$

$$= \sum_{y \neq x} \sum_{z \neq x} \mathbb{E}\Big[f_y s(x) s(y) B_{xy} f_z s(x) s(z) B_{xz}\Big]$$

$$= \sum_{y \neq x} f_y^2 \, \mathbb{E}[B_{xy}^2] + 0$$

$$= \sum_{y \neq x} f_y^2 \cdot \frac{1}{k} = \frac{\|\mathbf{f}_{-x}\|_2^2}{k} \qquad \square$$

Basic Count Sketch: Analysis

For an n-dimensional vector $\mathbf{f}$ and $i \in [n]$ define $\mathbf{f}_{-i}$ to be the $(n-1)$-dimensional vector obtained by dropping index i. Then $\|\mathbf{f}_{-x}\|_2^2 = \sum_{y \neq x} f_y^2 = \|\mathbf{f}\|_2^2 - f_x^2$.

Lemma

$\mathrm{Var}[\hat{f}_x] = \mathbb{E}[(\hat{f}_x - f_x)^2] = \frac{\|\mathbf{f}_{-x}\|_2^2}{k}$

Proof.

$$\begin{aligned}
\mathbb{E}[(\hat{f}_x - f_x)^2] &= \mathbb{E}\Big[\Big(\sum_{y \neq x} f_y s(x) s(y) B_{xy}\Big)^2\Big] \\
&= \sum_{y \neq x} \sum_{z \neq x} \mathbb{E}\Big[f_y s(x) s(y) B_{xy} f_z s(x) s(z) B_{xz}\Big] \\
&= \sum_{y \neq x} f_y^2 \,\mathbb{E}[B_{xy}^2] + 0 \\
&= \sum_{y \neq x} f_y^2 \cdot \frac{1}{k} = \frac{\|\mathbf{f}_{-x}\|_2^2}{k}
\end{aligned}$$

□

Basic Count Sketch: Analysis

For an n-dimensional vector $\mathbf{f}$ and $i \in [n]$ define $\mathbf{f}_{-i}$ to be the $(n-1)$-dimensional vector obtained by dropping index i. Then $\|\mathbf{f}_{-x}\|_2^2 = \sum_{y \neq x} f_y^2 = \|\mathbf{f}\|_2^2 - f_x^2$.

Lemma

$\mathrm{Var}[\hat{f}_x] = \mathbb{E}[(\hat{f}_x - f_x)^2] = \frac{\|\mathbf{f}_{-x}\|_2^2}{k}$

Proof.

$$\begin{aligned}
\mathbb{E}[(\hat{f}_x - f_x)^2] &= \mathbb{E}\Big[\Big(\sum_{y \neq x} f_y s(x) s(y) B_{xy}\Big)^2\Big] \\
&= \sum_{y \neq x} \sum_{z \neq x} \mathbb{E}\Big[f_y s(x) s(y) B_{xy} f_z s(x) s(z) B_{xz}\Big] \\
&= \sum_{y \neq x} f_y^2 \, \mathbb{E}[B_{xy}^2] + 0 \\
&= \sum_{y \neq x} f_y^2 \cdot \frac{1}{k} = \frac{\|\mathbf{f}_{-x}\|_2^2}{k} \qquad \square
\end{aligned}$$

Any term where $y \neq z$ is 0, by the same argument as before:

$$\begin{aligned}
&\mathbb{E}[f_y s(x) s(y) B_{xy} f_z s(x) s(z) B_{xz}] \\
&\quad = f_y f_z \, \mathbb{E}[(s(x))^2 s(y) s(z) B_{xy} B_{xz}] && \text{(Linearity of } \mathbb{E}) \\
&\quad = f_y f_z \, \mathbb{E}[s(y) s(z) B_{xy} B_{xz}] && ((s(x))^2 = 1) \\
&\quad = f_y f_z \, \mathbb{E}[s(y) s(z)] \, \mathbb{E}[B_{xy} B_{xz}] && (h, s \text{ independent}) \\
&\quad = f_y f_z \, \mathbb{E}[s(y)] \, \mathbb{E}[s(z)] \, \mathbb{E}[B_{xy} B_{xz}] && (s \text{ is 2-independent}) \\
&\quad = 0 && (\mathbb{E}[s(y)] = 0)
\end{aligned}$$

Basic Count Sketch: Analysis

For an n-dimensional vector $\mathbf{f}$ and $i \in [n]$ define $\mathbf{f}_{-i}$ to be the $(n-1)$-dimensional vector obtained by dropping index i. Then $\|\mathbf{f}_{-x}\|_2^2 = \sum_{y \neq x} f_y^2 = \|\mathbf{f}\|_2^2 - f_x^2$.

Lemma

$\mathrm{Var}[\hat{f}_x] = \mathbb{E}[(\hat{f}_x - f_x)^2] = \frac{\|\mathbf{f}_{-x}\|_2^2}{k}$

Proof.

$$
\begin{aligned}
\mathbb{E}[(\hat{f}_x - f_x)^2] &= \mathbb{E}\Big[\Big(\sum_{y \neq x} f_y s(x) s(y) B_{xy}\Big)^2\Big] \\
&= \sum_{y \neq x} \sum_{z \neq x} \mathbb{E}\Big[f_y s(x) s(y) B_{xy} f_z s(x) s(z) B_{xz}\Big] \\
&= \sum_{y \neq x} f_y^2 \, \mathbb{E}[B_{xy}^2] + 0 \\
&= \sum_{y \neq x} f_y^2 \cdot \frac{1}{k} = \frac{\|\mathbf{f}_{-x}\|_2^2}{k}
\end{aligned}
$$

□

$\mathbb{E}[B_{xy}^2] = \mathbb{E}[B_{xy}] = \Pr[B_{xy} = 1] = \Pr[h(x) = h(y)] = \frac{1}{k}$.

Basic Count Sketch: Analysis

For an n-dimensional vector $\mathbf{f}$ and $i \in [n]$ define $\mathbf{f}_{-i}$ to be the $(n-1)$-dimensional vector obtained by dropping index i. Then $\|\mathbf{f}_{-x}\|_2^2 = \sum_{y \neq x} f_y^2 = \|\mathbf{f}\|_2^2 - f_x^2$.

Lemma

$\mathrm{Var}[\hat{f}_x] = \mathbb{E}[(\hat{f}_x - f_x)^2] = \frac{\|\mathbf{f}_{-x}\|_2^2}{k}$

Proof.

$$
\begin{aligned}
\mathbb{E}[(\hat{f}_x - f_x)^2] &= \mathbb{E}\Big[\Big(\sum_{y \neq x} f_y s(x) s(y) B_{xy}\Big)^2\Big] \\
&= \sum_{y \neq x} \sum_{z \neq x} \mathbb{E}\Big[f_y s(x) s(y) B_{xy} f_z s(x) s(z) B_{xz}\Big] \\
&= \sum_{y \neq x} f_y^2 \, \mathbb{E}[B_{xy}^2] + 0 \\
&= \sum_{y \neq x} f_y^2 \cdot \tfrac{1}{k} = \tfrac{\|\mathbf{f}_{-x}\|_2^2}{k} \qquad \square
\end{aligned}
$$

Basic Count Sketch: Analysis

Combining the information from the last two slides

$$\mathbb{E}[\hat{f}_x] = f_x \qquad \mathrm{Var}[\hat{f}_x] = \frac{\|\mathbf{f}_{-x}\|_2^2}{k}$$

We can apply Chebyshev to get

$$\Pr\left[|\hat{f}_x - f_x| \geq \varepsilon \cdot \|\mathbf{f}_{-x}\|_2\right] \leq \frac{\mathrm{Var}[\hat{f}_x]}{(\varepsilon \cdot \|\mathbf{f}_{-x}\|_2)^2} = \frac{1}{k\varepsilon^2} \leq \frac{1}{3}$$

Standard form of Chebyshev says that $\Pr[|X - \mu| \geq t\sigma] \leq \frac{1}{t^2}$.

This uses a different form, which we can derive as follows: For any $B > 0$,

$$\begin{aligned}
\Pr[|X - \mu| \geq B] &= \Pr[|X - \mu| \geq t\sigma] && (\text{Setting } t = \tfrac{B}{\sigma}) \\
&\leq \frac{1}{t^2} && (\text{By Chebyshev}) \\
&= \frac{\sigma^2}{B^2} && (\text{Using } t = \tfrac{B}{\sigma}) \\
&= \frac{\mathrm{Var}[X]}{B^2} && (\text{By definition, } \mathrm{Var}[X] = \sigma^2)
\end{aligned}$$

The median trick

Define a random variable X to be *bad* if $|X - \mathbb{E}[X]| > \Delta$.

Consider random variables $X_1, \ldots, X_t \in \mathbb{R}$ with $\mathbb{E}[X_1] = \cdots = \mathbb{E}[X_t] = \mu$.

Let Y be the *median* of $X_1, \ldots, X_t$. If Y is bad, then at least $\lceil \frac{t}{2} \rceil$ of the X_i are bad.

Suppose the X_i are all independent, and $\Pr[X_i \text{ bad}] \leq \frac{1}{3}$. Let $B_i = [X_i \text{ bad}]$ and $B = \sum_i B_i$. Then $\mathbb{E}[B] \leq \frac{t}{3}$, and

$$\Pr[Y \text{ bad}] \leq \Pr[B \geq \tfrac{t}{2}] = \Pr[B \geq \tfrac{3}{2}\tfrac{t}{3}]$$
$$\leq \left(\frac{e^{\frac{1}{2}}}{(\frac{3}{2})^{\frac{3}{2}}} \right)^{\frac{t}{3}} \leq \left(e^{-\frac{(\frac{1}{2})^2}{3}} \right)^{\frac{t}{3}} = e^{-\frac{t}{36}}$$

Thus, for any $\delta > 0$, to get $\Pr[Y \text{ bad}] \leq \delta$ we want $e^{-\frac{t}{36}} \leq \delta$, or equivalently $t \geq 36 \ln \frac{1}{\delta}$.

The median trick

Define a random variable X to be *bad* if $|X - \mathbb{E}[X]| > \Delta$.

Consider random variables $X_1, \ldots, X_t \in \mathbb{R}$ with $\mathbb{E}[X_1] = \cdots = \mathbb{E}[X_t] = \mu$.

Let Y be the *median* of $X_1, \ldots, X_t$. If Y is bad, then at least $\lceil \frac{t}{2} \rceil$ of the X_i are bad.

Suppose the X_i are all independent, and $\Pr[X_i \text{ bad}] \leq \frac{1}{3}$. Let $B_i = [X_i \text{ bad}]$ and $B = \sum_i B_i$. Then $\mathbb{E}[B] \leq \frac{t}{3}$, and

$$\begin{aligned} \Pr[Y \text{ bad}] &\leq \Pr[B \geq \tfrac{t}{2}] = \Pr[B \geq \tfrac{3}{2}\tfrac{t}{3}] \\ &\leq \left(\frac{e^{\frac{1}{2}}}{(\frac{3}{2})^{\frac{3}{2}}} \right)^{\frac{t}{3}} \leq \left(e^{-\frac{(\frac{1}{2})^2}{3}} \right)^{\frac{t}{3}} = e^{-\frac{t}{36}} \end{aligned}$$

Thus, for any $\delta > 0$, to get $\Pr[Y \text{ bad}] \leq \delta$ we want $e^{-\frac{t}{36}} \leq \delta$, or equivalently $t \geq 36 \ln \frac{1}{\delta}$.

The median trick

Define a random variable X to be *bad* if $|X - \mathbb{E}[X]| > \Delta$.

Consider random variables $X_1, \ldots, X_t \in \mathbb{R}$ with $\mathbb{E}[X_1] = \cdots = \mathbb{E}[X_t] = \mu$.

Let Y be the *median* of $X_1, \ldots, X_t$. If Y is bad, then at least $\lceil \frac{t}{2} \rceil$ of the X_i are bad.

Suppose the X_i are all independent, and $\Pr[X_i \text{ bad}] \leq \frac{1}{3}$. Let $B_i = [X_i \text{ bad}]$ and $B = \sum_i B_i$. Then $\mathbb{E}[B] \leq \frac{t}{3}$, and

$$\begin{aligned}\Pr[Y \text{ bad}] \leq \Pr[B \geq \tfrac{t}{2}] &= \Pr[B \geq \tfrac{3}{2}\tfrac{t}{3}] \\ &\leq \left(\frac{e^{\frac{1}{2}}}{(\frac{3}{2})^{\frac{3}{2}}}\right)^{\frac{t}{3}} \leq \left(e^{-\frac{(\frac{1}{2})^2}{3}}\right)^{\frac{t}{3}} = e^{-\frac{t}{36}}\end{aligned}$$

Thus, for any $\delta > 0$, to get $\Pr[Y \text{ bad}] \leq \delta$ we want $e^{-\frac{t}{36}} \leq \delta$, or equivalently $t \geq 36 \ln \frac{1}{\delta}$.

The median trick

Define a random variable X to be *bad* if $|X - \mathbb{E}[X]| > \Delta$.

Consider random variables $X_1, \ldots, X_t \in \mathbb{R}$ with $\mathbb{E}[X_1] = \cdots = \mathbb{E}[X_t] = \mu$.

Let Y be the *median* of $X_1, \ldots, X_t$. If Y is bad, then at least $\lceil \frac{t}{2} \rceil$ of the X_i are bad.

Suppose the X_i are all independent, and $\Pr[X_i \text{ bad}] \leq \frac{1}{3}$. Let $B_i = [X_i \text{ bad}]$ and $B = \sum_i B_i$. Then $\mathbb{E}[B] \leq \frac{t}{3}$, and

$$\begin{aligned}\Pr[Y \text{ bad}] \leq \Pr[B \geq \tfrac{t}{2}] &= \Pr[B \geq \tfrac{3}{2}\tfrac{t}{3}] \\ &\leq \left(\frac{e^{\frac{1}{2}}}{(\frac{3}{2})^{\frac{3}{2}}} \right)^{\frac{t}{3}} \leq \left(e^{-\frac{(\frac{1}{2})^2}{3}} \right)^{\frac{t}{3}} = e^{-\frac{t}{36}}\end{aligned}$$

Thus, for any $\delta > 0$, to get $\Pr[Y \text{ bad}] \leq \delta$ we want $e^{-\frac{t}{36}} \leq \delta$, or equivalently $t \geq 36 \ln \frac{1}{\delta}$.

The median trick

Define a random variable X to be *bad* if $|X - \mathbb{E}[X]| > \Delta$.

Consider random variables $X_1, \ldots, X_t \in \mathbb{R}$ with $\mathbb{E}[X_1] = \cdots = \mathbb{E}[X_t] = \mu$.

Let Y be the *median* of $X_1, \ldots, X_t$. If Y is bad, then at least $\lceil \frac{t}{2} \rceil$ of the X_i are bad.

Suppose the X_i are all independent, and $\Pr[X_i \text{ bad}] \leq \frac{1}{3}$. Let $B_i = [X_i \text{ bad}]$ and $B = \sum_i B_i$. Then $\mathbb{E}[B] \leq \frac{t}{3}$, and

$$\begin{aligned}\Pr[Y \text{ bad}] &\leq \Pr[B \geq \tfrac{t}{2}] = \Pr[B \geq \tfrac{3}{2}\tfrac{t}{3}] \\ &\leq \left(\frac{e^{\frac{1}{2}}}{(\frac{3}{2})^{\frac{3}{2}}}\right)^{\frac{t}{3}} \leq \left(e^{-\frac{(\frac{1}{2})^2}{3}}\right)^{\frac{t}{3}} = e^{-\frac{t}{36}}\end{aligned}$$

Thus, for any $\delta > 0$, to get $\Pr[Y \text{ bad}] \leq \delta$ we want $e^{-\frac{t}{36}} \leq \delta$, or equivalently $t \geq 36 \ln \frac{1}{\delta}$.

The median trick

Define a random variable X to be *bad* if $|X - \mathbb{E}[X]| > \Delta$.

Consider random variables $X_1, \dots, X_t \in \mathbb{R}$ with $\mathbb{E}[X_1] = \cdots = \mathbb{E}[X_t] = \mu$.

Let Y be the *median* of $X_1, \dots, X_t$. If Y is bad, then at least $\lceil \frac{t}{2} \rceil$ of the X_i are bad.

Suppose the X_i are all independent, and $\Pr[X_i \text{ bad}] \leq \frac{1}{3}$. Let $B_i = [X_i \text{ bad}]$ and $B = \sum_i B_i$. Then $\mathbb{E}[B] \leq \frac{t}{3}$, and

$$\begin{aligned}\Pr[Y \text{ bad}] \leq \Pr[B \geq \tfrac{t}{2}] &= \Pr[B \geq \tfrac{3}{2}\tfrac{t}{3}] \\ &\leq \left(\frac{e^{\frac{1}{2}}}{(\frac{3}{2})^{\frac{3}{2}}}\right)^{\frac{t}{3}} \leq \left(e^{-\frac{(\frac{1}{2})^2}{3}}\right)^{\frac{t}{3}} = e^{-\frac{t}{36}}\end{aligned}$$

Thus, for any $\delta > 0$, to get $\Pr[Y \text{ bad}] \leq \delta$ we want $e^{-\frac{t}{36}} \leq \delta$, or equivalently $t \geq 36 \ln \frac{1}{\delta}$.

The median trick

Define a random variable X to be *bad* if $|X - \mathbb{E}[X]| > \Delta$.

Consider random variables $X_1, \ldots, X_t \in \mathbb{R}$ with $\mathbb{E}[X_1] = \cdots = \mathbb{E}[X_t] = \mu$.

Let Y be the *median* of $X_1, \ldots, X_t$. If Y is bad, then at least $\lceil \frac{t}{2} \rceil$ of the X_i are bad.

Suppose the X_i are all independent, and $\Pr[X_i \text{ bad}] \leq \frac{1}{3}$. Let $B_i = [X_i \text{ bad}]$ and $B = \sum_i B_i$. Then $\mathbb{E}[B] \leq \frac{t}{3}$, and

$$\Pr[Y \text{ bad}] \leq \Pr[B \geq \tfrac{t}{2}] = \Pr[B \geq \tfrac{3}{2}\tfrac{t}{3}]$$
$$\leq \left(\frac{e^{\frac{1}{2}}}{(\frac{3}{2})^{\frac{3}{2}}}\right)^{\frac{t}{3}} \leq \left(e^{-\frac{(\frac{1}{2})^2}{3}}\right)^{\frac{t}{3}} = e^{-\frac{t}{36}}$$

Thus, for any $\delta > 0$, to get $\Pr[Y \text{ bad}] \leq \delta$ we want $e^{-\frac{t}{36}} \leq \delta$, or equivalently $t \geq 36 \ln \frac{1}{\delta}$.

The median trick

Define a random variable X to be *bad* if $|X - \mathbb{E}[X]| > \Delta$.

Consider random variables $X_1, \ldots, X_t \in \mathbb{R}$ with $\mathbb{E}[X_1] = \cdots = \mathbb{E}[X_t] = \mu$.

Let Y be the *median* of $X_1, \ldots, X_t$. If Y is bad, then at least $\lceil \frac{t}{2} \rceil$ of the X_i are bad.

Suppose the X_i are all independent, and $\Pr[X_i \text{ bad}] \leq \frac{1}{3}$. Let $B_i = [X_i \text{ bad}]$ and $B = \sum_i B_i$. Then $\mathbb{E}[B] \leq \frac{t}{3}$, and

$$\Pr[Y \text{ bad}] \leq \Pr[B \geq \tfrac{t}{2}] = \Pr[B \geq \tfrac{3}{2}\tfrac{t}{3}]$$
$$\leq \left(\frac{e^{\frac{1}{2}}}{(\frac{3}{2})^{\frac{3}{2}}}\right)^{\frac{t}{3}} \leq \left(e^{-\frac{(\frac{1}{2})^2}{3}}\right)^{\frac{t}{3}} = e^{-\frac{t}{36}}$$

Thus, for any $\delta > 0$, to get $\Pr[Y \text{ bad}] \leq \delta$ we want $e^{-\frac{t}{36}} \leq \delta$, or equivalently $t \geq 36 \ln \frac{1}{\delta}$.

The median trick

Define a random variable X to be *bad* if $|X - \mathbb{E}[X]| > \Delta$.

Consider random variables $X_1, \ldots, X_t \in \mathbb{R}$ with $\mathbb{E}[X_1] = \cdots = \mathbb{E}[X_t] = \mu$.

Let Y be the *median* of $X_1, \ldots, X_t$. If Y is bad, then at least $\lceil \frac{t}{2} \rceil$ of the X_i are bad.

Suppose the X_i are all independent, and $\Pr[X_i \text{ bad}] \leq \frac{1}{3}$. Let $B_i = [X_i \text{ bad}]$ and $B = \sum_i B_i$. Then $\mathbb{E}[B] \leq \frac{t}{3}$, and

$$\Pr[Y \text{ bad}] \leq \Pr[B \geq \tfrac{t}{2}] = \Pr[B \geq \tfrac{3}{2}\tfrac{t}{3}]$$

$$\leq \left(\frac{e^{\frac{1}{2}}}{(\frac{3}{2})^{\frac{3}{2}}}\right)^{\frac{t}{3}} \leq \left(e^{-\frac{(\frac{1}{2})^2}{3}}\right)^{\frac{t}{3}} = e^{-\frac{t}{36}}$$

Thus, for any $\delta > 0$, to get $\Pr[Y \text{ bad}] \leq \delta$ we want $e^{-\frac{t}{36}} \leq \delta$, or equivalently $t \geq 36 \ln \frac{1}{\delta}$.

The median trick

Define a random variable X to be *bad* if $|X - \mathbb{E}[X]| > \Delta$.

Consider random variables $X_1, \dots, X_t \in \mathbb{R}$ with $\mathbb{E}[X_1] = \cdots = \mathbb{E}[X_t] = \mu$.

Let Y be the *median* of $X_1, \dots, X_t$. If Y is bad, then at least $\lceil \frac{t}{2} \rceil$ of the X_i are bad.

Suppose the X_i are all independent, and $\Pr[X_i \text{ bad}] \leq \frac{1}{3}$. Let $B_i = [X_i \text{ bad}]$ and $B = \sum_i B_i$. Then $\mathbb{E}[B] \leq \frac{t}{3}$, and

$$\begin{aligned}\Pr[Y \text{ bad}] &\leq \Pr[B \geq \tfrac{t}{2}] = \Pr[B \geq \tfrac{3}{2}\tfrac{t}{3}] \\ &\leq \left(\frac{e^{\frac{1}{2}}}{(\frac{3}{2})^{\frac{3}{2}}}\right)^{\frac{t}{3}} \leq \left(e^{-\frac{(\frac{1}{2})^2}{3}}\right)^{\frac{t}{3}} = e^{-\frac{t}{36}}\end{aligned}$$

Thus, for any $\delta > 0$, to get $\Pr[Y \text{ bad}] \leq \delta$ we want $e^{-\frac{t}{36}} \leq \delta$, or equivalently $t \geq 36 \ln \frac{1}{\delta}$.

The median trick

Define a random variable X to be *bad* if $|X - \mathbb{E}[X]| > \Delta$.

Consider random variables $X_1, \ldots, X_t \in \mathbb{R}$ with $\mathbb{E}[X_1] = \cdots = \mathbb{E}[X_t] = \mu$.

Let Y be the *median* of $X_1, \ldots, X_t$. If Y is bad, then at least $\lceil \frac{t}{2} \rceil$ of the X_i are bad.

Suppose the X_i are all independent, and $\Pr[X_i \text{ bad}] \leq \frac{1}{3}$. Let $B_i = [X_i \text{ bad}]$ and $B = \sum_i B_i$. Then $\mathbb{E}[B] \leq \frac{t}{3}$, and

$$\Pr[Y \text{ bad}] \leq \Pr[B \geq \tfrac{t}{2}] = \Pr[B \geq \tfrac{3}{2}\tfrac{t}{3}]$$

$$\leq \left(\frac{e^{\frac{1}{2}}}{(\frac{3}{2})^{\frac{3}{2}}}\right)^{\frac{t}{3}} \leq \left(e^{-\frac{(\frac{1}{2})^2}{3}}\right)^{\frac{t}{3}} = e^{-\frac{t}{36}}$$

Thus, for any $\delta > 0$, to get $\Pr[Y \text{ bad}] \leq \delta$ we want $e^{-\frac{t}{36}} \leq \delta$, or equivalently $t \geq 36 \ln \frac{1}{\delta}$.

This is because for $0 \leq x \leq 1$ we have

$$\frac{e^x}{(1+x)^{1+x}} \leq e^{-\frac{x^2}{3}}$$

We could actually have gotten a slightly tighter result by just evaluating directly. In particular, we can easily reduce the constant from 36 to $\frac{6}{\ln \frac{27}{8e}} \approx 27.727\ldots$

However, it is nice to know simple approximations like the one above, because most of the time you don't need the tight result.

The median trick

Define a random variable X to be *bad* if $|X - \mathbb{E}[X]| > \Delta$.

Consider random variables $X_1, \ldots, X_t \in \mathbb{R}$ with $\mathbb{E}[X_1] = \cdots = \mathbb{E}[X_t] = \mu$.

Let Y be the *median* of $X_1, \ldots, X_t$. If Y is bad, then at least $\lceil \frac{t}{2} \rceil$ of the X_i are bad.

Suppose the X_i are all independent, and $\Pr[X_i \text{ bad}] \leq \frac{1}{3}$. Let $B_i = [X_i \text{ bad}]$ and $B = \sum_i B_i$. Then $\mathbb{E}[B] \leq \frac{t}{3}$, and

$$\begin{aligned}\Pr[Y \text{ bad}] \leq \Pr[B \geq \tfrac{t}{2}] &= \Pr[B \geq \tfrac{3}{2}\tfrac{t}{3}] \\ &\leq \left(\frac{e^{\frac{1}{2}}}{(\frac{3}{2})^{\frac{3}{2}}}\right)^{\frac{t}{3}} \leq \left(e^{-\frac{(\frac{1}{2})^2}{3}}\right)^{\frac{t}{3}} = e^{-\frac{t}{36}}\end{aligned}$$

Thus, for any $\delta > 0$, to get $\Pr[Y \text{ bad}] \leq \delta$ we want $e^{-\frac{t}{36}} \leq \delta$, or equivalently $t \geq 36 \ln \frac{1}{\delta}$.

The median trick

Define a random variable X to be *bad* if $|X - \mathbb{E}[X]| > \Delta$.

Consider random variables $X_1, \ldots, X_t \in \mathbb{R}$ with $\mathbb{E}[X_1] = \cdots = \mathbb{E}[X_t] = \mu$.

Let Y be the *median* of $X_1, \ldots, X_t$. If Y is bad, then at least $\lceil \frac{t}{2} \rceil$ of the X_i are bad.

Suppose the X_i are all independent, and $\Pr[X_i \text{ bad}] \leq \frac{1}{3}$. Let $B_i = [X_i \text{ bad}]$ and $B = \sum_i B_i$. Then $\mathbb{E}[B] \leq \frac{t}{3}$, and

$$\begin{aligned}\Pr[Y \text{ bad}] \leq \Pr[B \geq \tfrac{t}{2}] &= \Pr[B \geq \tfrac{3}{2}\tfrac{t}{3}] \\ &\leq \left(\frac{e^{\frac{1}{2}}}{(\frac{3}{2})^{\frac{3}{2}}}\right)^{\frac{t}{3}} \leq \left(e^{-\frac{(\frac{1}{2})^2}{3}}\right)^{\frac{t}{3}} = e^{-\frac{t}{36}}\end{aligned}$$

Thus, for any $\delta > 0$, to get $\Pr[Y \text{ bad}] \leq \delta$ we want $e^{-\frac{t}{36}} \leq \delta$, or equivalently $t \geq 36 \ln \frac{1}{\delta}$.

The median trick

Define a random variable X to be *bad* if $|X - \mathbb{E}[X]| > \Delta$.

Consider random variables $X_1, \dots, X_t \in \mathbb{R}$ with $\mathbb{E}[X_1] = \cdots = \mathbb{E}[X_t] = \mu$.

Let Y be the *median* of $X_1, \dots, X_t$. If Y is bad, then at least $\lceil \frac{t}{2} \rceil$ of the X_i are bad.

Suppose the X_i are all independent, and $\Pr[X_i \text{ bad}] \le \frac{1}{3}$. Let $B_i = [X_i \text{ bad}]$ and $B = \sum_i B_i$. Then $\mathbb{E}[B] \le \frac{t}{3}$, and

$$\begin{aligned}\Pr[Y \text{ bad}] &\le \Pr[B \ge \tfrac{t}{2}] = \Pr[B \ge \tfrac{3}{2}\tfrac{t}{3}] \\ &\le \left(\frac{e^{\frac{1}{2}}}{(\frac{3}{2})^{\frac{3}{2}}}\right)^{\frac{t}{3}} \le \left(e^{-\frac{(\frac{1}{2})^2}{3}}\right)^{\frac{t}{3}} = e^{-\frac{t}{36}}\end{aligned}$$

Thus, for any $\delta > 0$, to get $\Pr[Y \text{ bad}] \le \delta$ we want $e^{-\frac{t}{36}} \le \delta$, or equivalently $t \ge 36 \ln \frac{1}{\delta}$.

Full Count Sketch: Pseudocode

function CS-INITIALIZE(n, ε, δ)
  $k \leftarrow \lceil \frac{3}{\varepsilon^2} \rceil, t \leftarrow \lceil 36 \ln \frac{1}{\delta} \rceil$
  for $i \in [t]$ **do**
    $C_i[0, \ldots, k-1] \leftarrow 0$
    pick 2-independent $h_i : [n] \to [k]$
    pick 2-independent $s_i : [n] \to \{-1, +1\}$

function CS-PROCESS(x, Δ)
  for $i \in [t]$ **do** $C_i[h_i(x)] \leftarrow C_i[h_i(x)] + s_i(x) \cdot \Delta$

function CS-QUERY(x)
  return $\text{median}_{i \in [t]}\, s_i(x) \cdot C_i[h_i(x)]$ ▷ Returns $\hat{f}_x$

Full Count Sketch: Pseudocode

function CS-INITIALIZE(n, ε, δ)
    $k \leftarrow \lceil \frac{3}{\varepsilon^2} \rceil, t \leftarrow \lceil 36 \ln \frac{1}{\delta} \rceil$
    for $i \in [t]$ **do**
        $C_i[0, \ldots, k-1] \leftarrow 0$
        pick 2-independent $h_i : [n] \rightarrow [k]$
        pick 2-independent $s_i : [n] \rightarrow \{-1, +1\}$

function CS-PROCESS(x, Δ)
    for $i \in [t]$ **do** $C_i[h_i(x)] \leftarrow C_i[h_i(x)] + s_i(x) \cdot \Delta$

function CS-QUERY(x)
    return $\text{median}_{i \in [t]}\, s_i(x) \cdot C_i[h_i(x)]$ $\triangleright$ Returns $\hat{f}_x$

Full Count Sketch: Pseudocode

```
function CS-INITIALIZE(n, ε, δ)
    k ← ⌈3/ε²⌉, t ← ⌈36 ln 1/δ⌉
    for i ∈ [t] do
        C_i[0, ..., k − 1] ← 0
        pick 2-independent h_i : [n] → [k]
        pick 2-independent s_i : [n] → {−1, +1}

function CS-PROCESS(x, Δ)
    for i ∈ [t] do C_i[h_i(x)] ← C_i[h_i(x)] + s_i(x) · Δ

function CS-QUERY(x)
    return median_{i∈[t]} s_i(x) · C_i[h_i(x)]        ▷ Returns f̂_x
```

Full Count Sketch: Pseudocode

```
function CS-INITIALIZE(n, ε, δ)
    k ← ⌈3/ε²⌉, t ← ⌈36 ln 1/δ⌉
    for i ∈ [t] do
        C_i[0, ..., k − 1] ← 0
        pick 2-independent h_i : [n] → [k]
        pick 2-independent s_i : [n] → {−1, +1}

function CS-PROCESS(x, Δ)
    for i ∈ [t] do C_i[h_i(x)] ← C_i[h_i(x)] + s_i(x) · Δ

function CS-QUERY(x)
    return median_{i∈[t]} s_i(x) · C_i[h_i(x)]        ▷ Returns f̂_x
```

Full Count Sketch: Summary

We have shown

Theorem

Given any $\varepsilon, \delta > 0$*, and any* $x \in [n]$*, let* $\hat{f}_x = \textsc{cs-query}(x)$*. Then*

$$\mathbb{E}[\hat{f}_x] = f_x \qquad \Pr\left[|\hat{f}_x - f_x| \geq \varepsilon \|\mathbf{f}_{-x}\|_2\right] \leq \delta$$

and the data structure uses $\mathcal{O}(\frac{1}{\varepsilon^2} \log \frac{1}{\delta}(\log n + \log u))$ *bits of space.*

Heavy Hitters

We can estimate f_x for any x, but how do we find the big ones as they come, i.e. the i such that $f_{x_i} > \varepsilon \|\mathbf{f}_{-x_i}\|_2$?

Heavy Hitters

We can estimate f_x for any x, but how do we find the big ones as they come, i.e. the i such that $f_{x_i} > \varepsilon \|\mathbf{f}_{-x_i}\|_2$?

Ask Mikkel Thorup about his "Heavy hitters via cluster-preserving clustering" paper from 2016 ;)

Linear Sketch

If we let $C(\sigma)$ denote the array C of counters after processing stream σ, then given any two streams σ_1, σ_2 we have $C(\sigma_1\sigma_2) = C(\sigma_1) + C(\sigma_2)$.

This makes the count sketch algorithm (either version) an example of a *linear sketch*.

We'll show one more linear sketch for frequency estimation in the cach register model, that is even simpler than the full count sketching.

Linear Sketch

If we let $C(\sigma)$ denote the array C of counters after processing stream σ, then given any two streams σ_1, σ_2 we have $C(\sigma_1\sigma_2) = C(\sigma_1) + C(\sigma_2)$.

This makes the count sketch algorithm (either version) an example of a *linear sketch*.

We'll show one more linear sketch for frequency estimation in the cach register model, that is even simpler than the full count sketching.

Linear Sketch

If we let $C(\sigma)$ denote the array C of counters after processing stream σ, then given any two streams σ_1, σ_2 we have $C(\sigma_1\sigma_2) = C(\sigma_1) + C(\sigma_2)$.

This makes the count sketch algorithm (either version) an example of a *linear sketch*.

We'll show one more linear sketch for frequency estimation in the cach register model, that is even simpler than the full count sketching.

Count-Min Sketch: Pseudocode

We start by computing k and t (slightly different).

function CMS-INITIALIZE(n, ε, δ)
    $k \leftarrow \left\lceil \frac{2}{\varepsilon} \right\rceil$, $t \leftarrow \lceil \log_2 \frac{1}{\delta} \rceil$
    for $i \in [t]$ **do**
        $C_i[0, \ldots, k-1] \leftarrow 0$
        pick 2-independent hash function $h_i : [n] \to [k]$

function CMS-PROCESS(x, Δ)
    for $i \in [t]$ **do** $C_i[h_i(x)] \leftarrow C_i[h_i(x)] + \Delta$

function CMS-QUERY(x)
    return $\min_{i \in [t]} C_i[h_i(x)]$ ▷ Returns $\hat{f}_x$

Count-Min Sketch: Pseudocode

Then we initialize the table and pick hash functions, but this time we don't need s.

function CMS-INITIALIZE(n, ε, δ)
  $k \leftarrow \left\lceil \frac{2}{\varepsilon} \right\rceil$, $t \leftarrow \left\lceil \log_2 \frac{1}{\delta} \right\rceil$
  for $i \in [t]$ **do**
    $C_i[0, \ldots, k-1] \leftarrow 0$
    pick 2-independent hash function $h_i : [n] \rightarrow [k]$

function CMS-PROCESS(x, Δ)
  for $i \in [t]$ **do** $C_i[h_i(x)] \leftarrow C_i[h_i(x)] + \Delta$

function CMS-QUERY(x)
  return $\min_{i \in [t]} C_i[h_i(x)]$ $\triangleright$ Returns $\hat{f}_x$

Count-Min Sketch: Pseudocode

We simply look up x using each of the t hash functions and add Δ to the running sum.

function CMS-INITIALIZE(n, ε, δ)
  $k \leftarrow \left\lceil \frac{2}{\varepsilon} \right\rceil$, $t \leftarrow \left\lceil \log_2 \frac{1}{\delta} \right\rceil$
  for $i \in [t]$ **do**
    $C_i[0, \ldots, k-1] \leftarrow 0$
    pick 2-independent hash function $h_i : [n] \rightarrow [k]$

function CMS-PROCESS(x, Δ)
  for $i \in [t]$ **do** $C_i[h_i(x)] \leftarrow C_i[h_i(x)] + \Delta$

function CMS-QUERY(x)
  return $\min_{i \in [t]} C_i[h_i(x)]$ ▷ Returns $\hat{f}_x$

Count-Min Sketch: Pseudocode

function CMS-INITIALIZE(n, ε, δ)
    $k \leftarrow \left\lceil \frac{2}{\varepsilon} \right\rceil$, $t \leftarrow \left\lceil \log_2 \frac{1}{\delta} \right\rceil$
    for $i \in [t]$ **do**
        $C_i[0, \ldots, k-1] \leftarrow 0$
        pick 2-independent hash function $h_i : [n] \to [k]$

function CMS-PROCESS(x, Δ)
    for $i \in [t]$ **do** $C_i[h_i(x)] \leftarrow C_i[h_i(x)] + \Delta$

function CMS-QUERY(x)
    return $\min_{i \in [t]} C_i[h_i(x)]$ $\triangleright$ Returns $\hat{f}_x$

Finally, we return the *minimum* of the accumulated values rather than the median.

Count-Min Sketch: Analysis

Theorem

Given any $\varepsilon, \delta > 0$, *and any* $x \in [n]$, *let* $\hat{f}_x = \textsc{cms-query}(x)$. *Then* $\hat{f}_x \geq f_x$ *and*

$$\Pr\left[\hat{f}_x - f_x \geq \varepsilon \|\mathbf{f}_{-x}\|_1\right] \leq \delta$$

and the data structure uses $\mathcal{O}(\frac{1}{\varepsilon} \log \frac{1}{\delta}(\log n + \log u))$ *bits of space.*

The space is better by a factor of $\frac{1}{\varepsilon}$, but the error guarantee is in terms of $\|\mathbf{f}_{-x}\|_1$ instead of $\|\mathbf{f}_{-x}\|_2$. What difference does that make?

Count-Min Sketch: Analysis

Theorem

Given any $\varepsilon, \delta > 0$, and any $x \in [n]$, let $\hat{f}_x = \textsc{cms-query}(x)$. Then $\hat{f}_x \geq f_x$ and

$$\Pr\left[\hat{f}_x - f_x \geq \varepsilon \|\mathbf{f}_{-x}\|_1\right] \leq \delta$$

and the data structure uses $\mathcal{O}(\frac{1}{\varepsilon} \log \frac{1}{\delta}(\log n + \log u))$ bits of space.

The space is better by a factor of $\frac{1}{\varepsilon}$, but the error guarantee is in terms of $\|\mathbf{f}_{-x}\|_1$ instead of $\|\mathbf{f}_{-x}\|_2$. What difference does that make?

Note that for any n-dimensional vector $\mathbf{f}$, and any $p \geq 1$ we define the ℓ_p norm as

$$\|\mathbf{f}\|_p = \left(\sum_{i=1}^{n} |f_i|^p\right)^{\frac{1}{p}}$$

In particular, the two norms we use are defined as

$$\|\mathbf{f}\|_1 = \sum_{i=1}^{n} |f_i| \qquad \text{(Manhattan norm)}$$

$$\|\mathbf{f}\|_2 = \sqrt{\sum_{i=1}^{n} |f_i|^2} \qquad \text{(Euclidean norm)}$$

As $p \to \infty$, the ℓ_p norm approaches the following norm

$$\|\mathbf{f}\|_\infty = \max_{i=1}^{n} |f_i| \qquad \text{(Maximum norm)}$$

Count-Min Sketch: Analysis

Theorem

Given any $\varepsilon, \delta > 0$*, and any* $x \in [n]$*, let* $\hat{f}_x = \textsc{cms-query}(x)$*. Then* $\hat{f}_x \geq f_x$ *and*

$$\Pr\left[\hat{f}_x - f_x \geq \varepsilon \|\mathbf{f}_{-x}\|_1\right] \leq \delta$$

and the data structure uses $\mathcal{O}(\frac{1}{\varepsilon} \log \frac{1}{\delta}(\log n + \log u))$ *bits of space.*

The space is better by a factor of $\frac{1}{\varepsilon}$, but the error guarantee is in terms of $\|\mathbf{f}_{-x}\|_1$ instead of $\|\mathbf{f}_{-x}\|_2$. What difference does that make?

Count-Min Sketch: Analysis

Theorem

Given any $\varepsilon, \delta > 0$, *and any* $x \in [n]$, *let* $\hat{f}_x = \textsc{cms-query}(x)$. *Then* $\hat{f}_x \geq f_x$ *and*

$$\Pr\left[\hat{f}_x - f_x \geq \varepsilon \|\mathbf{f}_{-x}\|_1\right] \leq \delta$$

and the data structure uses $\mathcal{O}(\frac{1}{\varepsilon} \log \frac{1}{\delta}(\log n + \log u))$ *bits of space.*

The space is better by a factor of $\frac{1}{\varepsilon}$, but the error guarantee is in terms of $\|\mathbf{f}_{-x}\|_1$ instead of $\|\mathbf{f}_{-x}\|_2$. What difference does that make?

$\|\mathbf{f}\|_2 \leq \|\mathbf{f}\|_1 \leq \sqrt{n}\|\mathbf{f}\|_2$, so $\|\cdot\|_2$ is always at least as good as $\|\cdot\|_1$ and sometimes much better.

Count-Min Sketch: Proof

Let $X_i = C_i[h_i(x)] - f_x$ and $B_{iy} = [h_i(x) = h_i(y)]$.

Then $X_i = \sum_{y \neq x} f_y B_{iy}$ and $\mathbb{E}[B_{iy}] = \frac{1}{k}$. So $\mathbb{E}[X_i] = \sum_{y \neq x} f_y \frac{1}{k} = \frac{\|\mathbf{f}_{-x}\|_1}{k}$. By Markov, we then have

$$\Pr[X_i \geq \varepsilon \|\mathbf{f}_{-x}\|_1] \leq \frac{\mathbb{E}[X_i]}{\varepsilon \|\mathbf{f}_{-x}\|_1} = \frac{1}{k\varepsilon} \leq \frac{1}{2}$$

Now

$$\begin{aligned} \hat{f}_x - f_x \geq \varepsilon \|\mathbf{f}_{-x}\|_1 &\iff \min_{i \in [t]} X_i \geq \varepsilon \|\mathbf{f}_{-x}\|_1 \\ &\iff \forall i \in [t] : X_i \geq \varepsilon \|\mathbf{f}_{-x}\|_1 \end{aligned}$$

So

$$\Pr\left[\hat{f}_x - f_x \geq \varepsilon \|\mathbf{f}_{-x}\|_1\right] \leq \frac{1}{2^t} \leq \delta$$

Count-Min Sketch: Proof

Let $X_i = C_i[h_i(x)] - f_x$ and $B_{iy} = [h_i(x) = h_i(y)]$. Then $X_i = \sum_{y \neq x} f_y B_{iy}$ and $\mathbb{E}[B_{iy}] = \frac{1}{k}$. So $\mathbb{E}[X_i] = \sum_{y \neq x} f_y \frac{1}{k} = \frac{\|\mathbf{f}_{-x}\|_1}{k}$. By Markov, we then have

$$\Pr[X_i \geq \varepsilon \|\mathbf{f}_{-x}\|_1] \leq \frac{\mathbb{E}[X_i]}{\varepsilon \|\mathbf{f}_{-x}\|_1} = \frac{1}{k\varepsilon} \leq \frac{1}{2}$$

Now

$$\begin{aligned} \hat{f}_x - f_x \geq \varepsilon \|\mathbf{f}_{-x}\|_1 &\iff \min_{i \in [t]} X_i \geq \varepsilon \|\mathbf{f}_{-x}\|_1 \\ &\iff \forall i \in [t] : X_i \geq \varepsilon \|\mathbf{f}_{-x}\|_1 \end{aligned}$$

So

$$\Pr\left[\hat{f}_x - f_x \geq \varepsilon \|\mathbf{f}_{-x}\|_1\right] \leq \frac{1}{2^t} \leq \delta$$

By definition,

$$\begin{aligned} C_i[h_i(x)] &= \sum_{\substack{j \in [n] \\ h_i(x_j) = h_i(x)}} \Delta_j = \sum_{j \in [n]} \Delta_j [h_i(x_j) = h_i(x)] = \sum_{j \in [n]} \Delta_j B_{ix_j} \\ &= \sum_{y \in [n]} \sum_{j \in I_y} \Delta_j B_{iy} \\ &= \sum_{y \in [n]} B_{iy} \sum_{j \in I_y} \Delta_j \\ &= \sum_{y \in [n]} B_{iy} f_y \end{aligned}$$

Thus

$$\begin{aligned} C_i[h_i(x)] - f_x &= C_i[h_i(x)] - f_x B_{ix} && \text{(Since } B_{ix} = 1\text{)} \\ &= \left(\sum_{y \in [n]} B_{iy} f_y\right) - f_x B_{ix} && \text{(From above)} \\ &= \sum_{y \neq x} B_{iy} f_y \end{aligned}$$

Count-Min Sketch: Proof

Since $h_i : [n] \to [k]$ is 2-independent.

Let $X_i = C_i[h_i(x)] - f_x$ and $B_{iy} = [h_i(x) = h_i(y)]$. Then $X_i = \sum_{y \neq x} f_y B_{iy}$ and $\mathbb{E}[B_{iy}] = \frac{1}{k}$. So $\mathbb{E}[X_i] = \sum_{y \neq x} f_y \frac{1}{k} = \frac{\|\mathbf{f}_{-x}\|_1}{k}$. By Markov, we then have

$$\Pr[X_i \geq \varepsilon \|\mathbf{f}_{-x}\|_1] \leq \frac{\mathbb{E}[X_i]}{\varepsilon \|\mathbf{f}_{-x}\|_1} = \frac{1}{k\varepsilon} \leq \frac{1}{2}$$

Now

$$\begin{aligned} \hat{f}_x - f_x \geq \varepsilon \|\mathbf{f}_{-x}\|_1 &\iff \min_{i \in [t]} X_i \geq \varepsilon \|\mathbf{f}_{-x}\|_1 \\ &\iff \forall i \in [t] : X_i \geq \varepsilon \|\mathbf{f}_{-x}\|_1 \end{aligned}$$

So

$$\Pr\left[\hat{f}_x - f_x \geq \varepsilon \|\mathbf{f}_{-x}\|_1\right] \leq \frac{1}{2^t} \leq \delta$$

Count-Min Sketch: Proof

Let $X_i = C_i[h_i(x)] - f_x$ and $B_{iy} = [h_i(x) = h_i(y)]$.
Then $X_i = \sum_{y \neq x} f_y B_{iy}$ and $\mathbb{E}[B_{iy}] = \frac{1}{k}$. So
$\mathbb{E}[X_i] = \sum_{y \neq x} f_y \frac{1}{k} = \frac{\|\mathbf{f}_{-x}\|_1}{k}$. By Markov, we then have

$$\Pr[X_i \geq \varepsilon \|\mathbf{f}_{-x}\|_1] \leq \frac{\mathbb{E}[X_i]}{\varepsilon \|\mathbf{f}_{-x}\|_1} = \frac{1}{k\varepsilon} \leq \frac{1}{2}$$

Now

$$\begin{aligned} \hat{f}_x - f_x \geq \varepsilon \|\mathbf{f}_{-x}\|_1 &\iff \min_{i \in [t]} X_i \geq \varepsilon \|\mathbf{f}_{-x}\|_1 \\ &\iff \forall i \in [t] : X_i \geq \varepsilon \|\mathbf{f}_{-x}\|_1 \end{aligned}$$

So

$$\Pr\left[\hat{f}_x - f_x \geq \varepsilon \|\mathbf{f}_{-x}\|_1\right] \leq \frac{1}{2^t} \leq \delta$$

Count-Min Sketch: Proof

Let $X_i = C_i[h_i(x)] - f_x$ and $B_{iy} = [h_i(x) = h_i(y)]$. Then $X_i = \sum_{y \neq x} f_y B_{iy}$ and $\mathbb{E}[B_{iy}] = \frac{1}{k}$. So $\mathbb{E}[X_i] = \sum_{y \neq x} f_y \frac{1}{k} = \frac{\|\mathbf{f}_{-x}\|_1}{k}$. By Markov, we then have

$$\Pr[X_i \geq \varepsilon \|\mathbf{f}_{-x}\|_1] \leq \frac{\mathbb{E}[X_i]}{\varepsilon \|\mathbf{f}_{-x}\|_1} = \frac{1}{k\varepsilon} \leq \frac{1}{2}$$

Now

$$\begin{aligned} \hat{f}_x - f_x \geq \varepsilon \|\mathbf{f}_{-x}\|_1 &\iff \min_{i \in [t]} X_i \geq \varepsilon \|\mathbf{f}_{-x}\|_1 \\ &\iff \forall i \in [t] : X_i \geq \varepsilon \|\mathbf{f}_{-x}\|_1 \end{aligned}$$

So

$$\Pr\left[\hat{f}_x - f_x \geq \varepsilon \|\mathbf{f}_{-x}\|_1\right] \leq \frac{1}{2^t} \leq \delta$$

Count-Min Sketch: Proof

Let $X_i = C_i[h_i(x)] - f_x$ and $B_{iy} = [h_i(x) = h_i(y)]$. Then $X_i = \sum_{y \neq x} f_y B_{iy}$ and $\mathbb{E}[B_{iy}] = \frac{1}{k}$. So $\mathbb{E}[X_i] = \sum_{y \neq x} f_y \frac{1}{k} = \frac{\|\mathbf{f}_{-x}\|_1}{k}$. By Markov, we then have

$$\Pr[X_i \geq \varepsilon \|\mathbf{f}_{-x}\|_1] \leq \frac{\mathbb{E}[X_i]}{\varepsilon \|\mathbf{f}_{-x}\|_1} = \frac{1}{k\varepsilon} \leq \frac{1}{2}$$

Now

$$\begin{aligned} \hat{f}_x - f_x \geq \varepsilon \|\mathbf{f}_{-x}\|_1 &\iff \min_{i \in [t]} X_i \geq \varepsilon \|\mathbf{f}_{-x}\|_1 \\ &\iff \forall i \in [t] : X_i \geq \varepsilon \|\mathbf{f}_{-x}\|_1 \end{aligned}$$

So

$$\Pr\left[\hat{f}_x - f_x \geq \varepsilon \|\mathbf{f}_{-x}\|_1\right] \leq \frac{1}{2^t} \leq \delta$$

Count-Min Sketch: Proof

Let $X_i = C_i[h_i(x)] - f_x$ and $B_{iy} = [h_i(x) = h_i(y)]$. Then $X_i = \sum_{y \neq x} f_y B_{iy}$ and $\mathbb{E}[B_{iy}] = \frac{1}{k}$. So $\mathbb{E}[X_i] = \sum_{y \neq x} f_y \frac{1}{k} = \frac{\|\mathbf{f}_{-x}\|_1}{k}$. By Markov, we then have

$$\Pr[X_i \geq \varepsilon \|\mathbf{f}_{-x}\|_1] \leq \frac{\mathbb{E}[X_i]}{\varepsilon \|\mathbf{f}_{-x}\|_1} = \frac{1}{k\varepsilon} \leq \frac{1}{2}$$

By definition of $k = \lceil \frac{2}{\varepsilon} \rceil$

Now

$$\hat{f}_x - f_x \geq \varepsilon \|\mathbf{f}_{-x}\|_1 \iff \min_{i \in [t]} X_i \geq \varepsilon \|\mathbf{f}_{-x}\|_1$$
$$\iff \forall i \in [t] : X_i \geq \varepsilon \|\mathbf{f}_{-x}\|_1$$

So

$$\Pr\left[\hat{f}_x - f_x \geq \varepsilon \|\mathbf{f}_{-x}\|_1\right] \leq \frac{1}{2^t} \leq \delta$$

Count-Min Sketch: Proof

Let $X_i = C_i[h_i(x)] - f_x$ and $B_{iy} = [h_i(x) = h_i(y)]$. Then $X_i = \sum_{y \neq x} f_y B_{iy}$ and $\mathbb{E}[B_{iy}] = \frac{1}{k}$. So $\mathbb{E}[X_i] = \sum_{y \neq x} f_y \frac{1}{k} = \frac{\|\mathbf{f}_{-x}\|_1}{k}$. By Markov, we then have

$$\Pr[X_i \geq \varepsilon \|\mathbf{f}_{-x}\|_1] \leq \frac{\mathbb{E}[X_i]}{\varepsilon \|\mathbf{f}_{-x}\|_1} = \frac{1}{k\varepsilon} \leq \frac{1}{2}$$

Now

$$\hat{f}_x - f_x \geq \varepsilon \|\mathbf{f}_{-x}\|_1 \iff \min_{i \in [t]} X_i \geq \varepsilon \|\mathbf{f}_{-x}\|_1$$

$$\iff \forall i \in [t] : X_i \geq \varepsilon \|\mathbf{f}_{-x}\|_1$$

So

$$\Pr\left[\hat{f}_x - f_x \geq \varepsilon \|\mathbf{f}_{-x}\|_1\right] \leq \frac{1}{2^t} \leq \delta$$

Count-Min Sketch: Proof

Let $X_i = C_i[h_i(x)] - f_x$ and $B_{iy} = [h_i(x) = h_i(y)]$. Then $X_i = \sum_{y \neq x} f_y B_{iy}$ and $\mathbb{E}[B_{iy}] = \frac{1}{k}$. So $\mathbb{E}[X_i] = \sum_{y \neq x} f_y \frac{1}{k} = \frac{\|\mathbf{f}_{-x}\|_1}{k}$. By Markov, we then have

$$\Pr[X_i \geq \varepsilon \|\mathbf{f}_{-x}\|_1] \leq \frac{\mathbb{E}[X_i]}{\varepsilon \|\mathbf{f}_{-x}\|_1} = \frac{1}{k\varepsilon} \leq \frac{1}{2}$$

Now

$$\begin{aligned} \hat{f}_x - f_x \geq \varepsilon \|\mathbf{f}_{-x}\|_1 &\iff \min_{i \in [t]} X_i \geq \varepsilon \|\mathbf{f}_{-x}\|_1 \\ &\iff \forall i \in [t] : X_i \geq \varepsilon \|\mathbf{f}_{-x}\|_1 \end{aligned}$$

So

$$\Pr\left[\hat{f}_x - f_x \geq \varepsilon \|\mathbf{f}_{-x}\|_1\right] \leq \frac{1}{2^t} \leq \delta$$

Count-Min Sketch: Proof

Let $X_i = C_i[h_i(x)] - f_x$ and $B_{iy} = [h_i(x) = h_i(y)]$. Then $X_i = \sum_{y \neq x} f_y B_{iy}$ and $\mathbb{E}[B_{iy}] = \frac{1}{k}$. So $\mathbb{E}[X_i] = \sum_{y \neq x} f_y \frac{1}{k} = \frac{\|\mathbf{f}_{-x}\|_1}{k}$. By Markov, we then have

$$\Pr[X_i \geq \varepsilon \|\mathbf{f}_{-x}\|_1] \leq \frac{\mathbb{E}[X_i]}{\varepsilon \|\mathbf{f}_{-x}\|_1} = \frac{1}{k\varepsilon} \leq \frac{1}{2}$$

Now

$$\begin{aligned}\hat{f}_x - f_x \geq \varepsilon \|\mathbf{f}_{-x}\|_1 &\iff \min_{i \in [t]} X_i \geq \varepsilon \|\mathbf{f}_{-x}\|_1 \\ &\iff \forall i \in [t] : X_i \geq \varepsilon \|\mathbf{f}_{-x}\|_1\end{aligned}$$

So

$$\Pr\left[\hat{f}_x - f_x \geq \varepsilon \|\mathbf{f}_{-x}\|_1\right] \leq \frac{1}{2^t} \leq \delta$$

Since all the h_i are independent, the probability of the events for all i happening is the product of the events for each of the t events happening.

Count-Min Sketch: Proof

Let $X_i = C_i[h_i(x)] - f_x$ and $B_{iy} = [h_i(x) = h_i(y)]$. Then $X_i = \sum_{y \neq x} f_y B_{iy}$ and $\mathbb{E}[B_{iy}] = \frac{1}{k}$. So $\mathbb{E}[X_i] = \sum_{y \neq x} f_y \frac{1}{k} = \frac{\|\mathbf{f}_{-x}\|_1}{k}$. By Markov, we then have

$$\Pr[X_i \geq \varepsilon \|\mathbf{f}_{-x}\|_1] \leq \frac{\mathbb{E}[X_i]}{\varepsilon \|\mathbf{f}_{-x}\|_1} = \frac{1}{k\varepsilon} \leq \frac{1}{2}$$

Now

$$\begin{aligned} \hat{f}_x - f_x \geq \varepsilon \|\mathbf{f}_{-x}\|_1 &\iff \min_{i \in [t]} X_i \geq \varepsilon \|\mathbf{f}_{-x}\|_1 \\ &\iff \forall i \in [t] : X_i \geq \varepsilon \|\mathbf{f}_{-x}\|_1 \end{aligned}$$

So

$$\Pr\left[\hat{f}_x - f_x \geq \varepsilon \|\mathbf{f}_{-x}\|_1\right] \leq \frac{1}{2^t} \leq \delta$$

By definition of $t = \lceil \log_2 \frac{1}{\delta} \rceil$

Summary

- We have seen different streaming models: Basic, cash register, and turnstile.
- We have looked at the frequency estimation problem in each of these models.
- Misra-Gries is a good deterministic algorithm for the basic and cash-register models.
- Count-Min Sketch is a better randomized algorithm for the cash-register model.
- Count Sketch is an even better randomized algorithm for the more general turnstile model (but costs more space).
- As part of Count Sketch we saw the "median trick", of using the median of independent unbiased estimates.
- Next time: Fun :)

Summary

- We have seen different streaming models: Basic, cash register, and turnstile.
- We have looked at the frequency estimation problem in each of these models.
- Misra-Gries is a good deterministic algorithm for the basic and cash-register models.
- Count-Min Sketch is a better randomized algorithm for the cash-register model.
- Count Sketch is an even better randomized algorithm for the more general turnstile model (but costs more space).
- As part of Count Sketch we saw the "median trick", of using the median of independent unbiased estimates.
- Next time: Fun :)

Summary

- We have seen different streaming models: Basic, cash register, and turnstile.
- We have looked at the frequency estimation problem in each of these models.
- Misra-Gries is a good deterministic algorithm for the basic and cash-register models.
- Count-Min Sketch is a better randomized algorithm for the cash-register model.
- Count Sketch is an even better randomized algorithm for the more general turnstile model (but costs more space).
- As part of Count Sketch we saw the "median trick", of using the median of independent unbiased estimates.
- Next time: Fun :)

Summary

- We have seen different streaming models: Basic, cash register, and turnstile.
- We have looked at the frequency estimation problem in each of these models.
- Misra-Gries is a good deterministic algorithm for the basic and cash-register models.
- Count-Min Sketch is a better randomized algorithm for the cash-register model.
- Count Sketch is an even better randomized algorithm for the more general turnstile model (but costs more space).
- As part of Count Sketch we saw the "median trick", of using the median of independent unbiased estimates.
- Next time: Fun :)

Summary

- We have seen different streaming models: Basic, cash register, and turnstile.
- We have looked at the frequency estimation problem in each of these models.
- Misra-Gries is a good deterministic algorithm for the basic and cash-register models.
- Count-Min Sketch is a better randomized algorithm for the cash-register model.
- Count Sketch is an even better randomized algorithm for the more general turnstile model (but costs more space).
- As part of Count Sketch we saw the "median trick", of using the median of independent unbiased estimates.
- Next time: Fun :)

Summary

- We have seen different streaming models: Basic, cash register, and turnstile.
- We have looked at the frequency estimation problem in each of these models.
- Misra-Gries is a good deterministic algorithm for the basic and cash-register models.
- Count-Min Sketch is a better randomized algorithm for the cash-register model.
- Count Sketch is an even better randomized algorithm for the more general turnstile model (but costs more space).
- As part of Count Sketch we saw the "median trick", of using the median of independent unbiased estimates.
- Next time: Fun :)

Summary

- We have seen different streaming models: Basic, cash register, and turnstile.
- We have looked at the frequency estimation problem in each of these models.
- Misra-Gries is a good deterministic algorithm for the basic and cash-register models.
- Count-Min Sketch is a better randomized algorithm for the cash-register model.
- Count Sketch is an even better randomized algorithm for the more general turnstile model (but costs more space).
- As part of Count Sketch we saw the "median trick", of using the median of independent unbiased estimates.
- Next time: Fun :)

Summary

- We have seen different streaming models: Basic, cash register, and turnstile.
- We have looked at the frequency estimation problem in each of these models.
- Misra-Gries is a good deterministic algorithm for the basic and cash-register models.
- Count-Min Sketch is a better randomized algorithm for the cash-register model.
- Count Sketch is an even better randomized algorithm for the more general turnstile model (but costs more space).
- As part of Count Sketch we saw the "median trick", of using the median of independent unbiased estimates.
- Next time: Fun :)