Randomized Algorithms

Streaming II: second moments and distinct counts

Alex Gittens
Department of Computer Science, Rensselaer Polytechnic Institute

Fall 2026

Learning Objectives

By the end of this lecture, you should be able to:

1. Explain why F_2 measures concentration and F_0 measures the number of distinct keys.
2. Design and analyze the AMS random-sign estimator, including the role of 4-wise independence.
3. Amplify a constant-success estimator using averaging and a median.
4. Derive the Flajolet–Martin bitmap and its first-zero bounds; explain stochastic averaging.

Lecture Roadmap

Two questions about a stream

AMS: second moment from random signs

Flajolet–Martin: count distinct keys

What to remember

PART 1

Two questions about a stream

1

Same length, different patterns

A stream contains keys $a_1, \ldots, a_N \in U$, where $u = |U|$. Write $f_x = \#\{t : a_t = x\}$.

	eight different keys	some repeated	one key repeated
Nonzero frequencies	1, 1, 1, 1, 1, 1, 1, 1	3, 2, 1, 1, 1	8
$N = \sum_x f_x$	8	8	8
$F_0 = \#\{x : f_x > 0\}$	8	5	1
$F_2 = \sum_x f_x^2$	8	16	64

Same length, different patterns

A stream contains keys $a_1, \ldots, a_N \in U$, where $u = |U|$. Write $f_x = \#\{t : a_t = x\}$.

	eight different keys	some repeated	one key repeated
Nonzero frequencies	1, 1, 1, 1, 1, 1, 1, 1	3, 2, 1, 1, 1	8
$N = \sum_x f_x$	8	8	8
$F_0 = \#\{x : f_x > 0\}$	8	5	1
$F_2 = \sum_x f_x^2$	8	16	64

Storing every observed key solves both problems—but uses $\Omega(F_0)$ counters or keys.
Question: Can we estimate F_0 and F_2 using memory logarithmic in the stream and universe sizes?

Why measure F_2? Think of a collision

Pick two positions I, J independently and uniformly from $\{1, \ldots, N\}$.

$$\Pr(a_I = a_J) = \sum_x \Pr(a_I = x) \Pr(a_J = x) = \sum_x \left(\frac{f_x}{N}\right)^2 = \frac{F_2}{N^2}.$$

Why measure F_2? Think of a collision

Pick two positions I, J independently and uniformly from $\{1, \ldots, N\}$.

$$\Pr(a_I = a_J) = \sum_x \Pr(a_I = x) \Pr(a_J = x) = \sum_x \left(\frac{f_x}{N}\right)^2 = \frac{F_2}{N^2}.$$

For the three streams: collision probabilities are $1/8$, $1/4$, and 1.
Interpretation: F_2 detects concentration even when two streams have equal total traffic.

Why measure F_2? Think of a collision

Pick two positions I, J independently and uniformly from $\{1, \ldots, N\}$.

$$\Pr(a_I = a_J) = \sum_x \Pr(a_I = x) \Pr(a_J = x) = \sum_x \left(\frac{f_x}{N}\right)^2 = \frac{F_2}{N^2}.$$

For the three streams: collision probabilities are $1/8$, $1/4$, and 1.
Interpretation: F_2 detects concentration even when two streams have equal total traffic. **Our design constraints:**

- One pass; process each arrival quickly.
- Error and failure targets: $\varepsilon \in (0, 1)$ and $\delta \in (0, 1)$.
- Space in *bits*, depending on $\log u$, $\log(N + 1)$, $1/\varepsilon$, $\log(1/\delta)$, not linearly on F_0.

PART 2

AMS: second moment from random signs

2

What does an accurate answer mean?

For insertion-only streams, $N \leq F_2 \leq N^2$.

What does an accurate answer mean?

For insertion-only streams, $N \le F_2 \le N^2$. An additive error εN^2 is almost useless if $F_2 \approx N$. A scale-adaptive goal is **relative error**:

$$\Pr\left(|\widehat{F}_2 - F_2| > \varepsilon F_2\right) \le \delta.$$

What does an accurate answer mean?

For insertion-only streams, $N \le F_2 \le N^2$. An additive error εN^2 is almost useless if $F_2 \approx N$. A scale-adaptive goal is **relative error**:

$$\Pr\left(|\widehat{F}_2 - F_2| > \varepsilon F_2\right) \le \delta.$$

We will first construct an estimator that is correct *on average*, then ask how reliably it concentrates.

A randomized identity for a sum of squares

Suppose we know numbers a, b, c but only want their sum of squares. Choose independent fair signs $s_1, s_2, s_3 \in \{-1, +1\}$, and define

$$Z = s_1 a + s_2 b + s_3 c.$$

A randomized identity for a sum of squares

Suppose we know numbers a, b, c but only want their sum of squares. Choose independent fair signs $s_1, s_2, s_3 \in \{-1, +1\}$, and define

$$Z = s_1 a + s_2 b + s_3 c.$$

Squaring introduces the wanted squares and unwanted cross terms:

$$Z^2 = a^2 + b^2 + c^2 + 2s_1 s_2 ab + 2s_1 s_3 ac + 2s_2 s_3 bc.$$

A randomized identity for a sum of squares

Suppose we know numbers a, b, c but only want their sum of squares. Choose independent fair signs $s_1, s_2, s_3 \in \{-1, +1\}$, and define

$$Z = s_1 a + s_2 b + s_3 c.$$

Squaring introduces the wanted squares and unwanted cross terms:

$$Z^2 = a^2 + b^2 + c^2 + 2s_1 s_2 ab + 2s_1 s_3 ac + 2s_2 s_3 bc.$$

Each cross term has expectation zero, since $\mathbb{E}[s_i s_j] = 0$ for $i \neq j$.

$$\boxed{\mathbb{E}[Z^2] = a^2 + b^2 + c^2.}$$

Idea: Random signs remove the cross terms *in expectation*.

Turn the identity into a streaming algorithm

Choose a fixed random sign $s(x) \in \{-1, +1\}$ for each key x.

AMS: one copy

Maintain an integer Z, initially zero.

On arrival of x: **Z** $\leftarrow Z + s(x)$.

At query time: output $X = Z^2$.

Turn the identity into a streaming algorithm

Choose a fixed random sign $s(x) \in \{-1, +1\}$ for each key x.

AMS: one copy

Maintain an integer Z, initially zero.

On arrival of x: $\mathbf{Z} \leftarrow Z + s(x)$.

At query time: output $X = Z^2$.

After the stream, the contribution of key x is $s(x)$ repeated f_x times:

$$Z = \sum_{x \in U} s(x) f_x.$$

Turn the identity into a streaming algorithm

Choose a fixed random sign $s(x) \in \{-1, +1\}$ for each key x.

AMS: one copy

Maintain an integer Z, initially zero.

On arrival of x: $\mathbf{Z} \leftarrow Z + s(x)$.

At query time: output $X = Z^2$.

After the stream, the contribution of key x is $s(x)$ repeated f_x times:

$$Z = \sum_{x \in U} s(x) f_x.$$

Store a *seed* describing s, not a table of u signs. We will choose a 4-wise independent family.

AMS: a worked example

Let $(f_A, f_B, f_C) = (4, 2, 1)$, so $F_2 = 21$.

s_A	s_B	s_C	$Z = 4s_A + 2s_B + s_C$	Z^2
$+$	$+$	$+$	7	49
$+$	$+$	$-$	5	25
$+$	$-$	$+$	3	9
$+$	$-$	$-$	1	1

AMS: a worked example

Let $(f_A, f_B, f_C) = (4, 2, 1)$, so $F_2 = 21$.

s_A	s_B	s_C	$Z = 4s_A + 2s_B + s_C$	Z^2
$+$	$+$	$+$	7	49
$+$	$+$	$-$	5	25
$+$	$-$	$+$	3	9
$+$	$-$	$-$	1	1

Each row represents two assignments: flipping every sign changes Z to $-Z$ but leaves Z^2 unchanged.

$$\frac{49 + 25 + 9 + 1}{4} = 21 = F_2.$$

Discussion: Why is one copy not enough for a reliable answer?

Why the estimate is unbiased

For any frequency vector,

$$X = Z^2 = \sum_x f_x^2 + \sum_{x \neq y} f_x f_y s(x) s(y).$$

Why the estimate is unbiased

For any frequency vector,

$$X = Z^2 = \sum_x f_x^2 + \sum_{x \neq y} f_x f_y s(x) s(y).$$

For distinct keys $x \neq y$, fair pairwise-independent signs give $\mathbb{E}[s(x)s(y)] = \mathbb{E}s(x)\,\mathbb{E}s(y) = 0$.

$$\boxed{\mathbb{E}X = F_2.}$$

Why the estimate is unbiased

For any frequency vector,

$$X = Z^2 = \sum_x f_x^2 + \sum_{x \neq y} f_x f_y s(x) s(y).$$

For distinct keys $x \neq y$, fair pairwise-independent signs give $\mathbb{E}[s(x)s(y)] = \mathbb{E}s(x)\,\mathbb{E}s(y) = 0$.

$$\boxed{\mathbb{E}X = F_2.}$$

But unbiasedness does *not* say that a single sample is accurate. To control deviations, we need $\mathrm{Var}(X) = \mathbb{E}[Z^4] - F_2^2$.

Which fourth-moment terms survive?

Expand

$$Z^4 = \sum_{i,j,k,\ell} f_i f_j f_k f_\ell s(i)s(j)s(k)s(\ell).$$

Four-wise independent fair signs make the expected product zero whenever any index occurs an odd number of times.

Which fourth-moment terms survive?

Expand

$$Z^4 = \sum_{i,j,k,\ell} f_i f_j f_k f_\ell s(i)s(j)s(k)s(\ell).$$

Four-wise independent fair signs make the expected product zero whenever any index occurs an odd number of times.

Multiplicity pattern	Representative term	Mean sign product
$1+1+1+1$	i, j, k, ℓ all distinct	0
$2+1+1$	i, i, j, k (three distinct)	0
$3+1$	i, i, i, j	0
4	i, i, i, i	1
$2+2$	i, i, j, j	1

Which fourth-moment terms survive?

Expand

$$Z^4 = \sum_{i,j,k,\ell} f_i f_j f_k f_\ell s(i)s(j)s(k)s(\ell).$$

Four-wise independent fair signs make the expected product zero whenever any index occurs an odd number of times.

Multiplicity pattern	Representative term	Mean sign product
$1+1+1+1$	i, j, k, ℓ all distinct	0
$2+1+1$	i, i, j, k (three distinct)	0
$3+1$	i, i, i, j	0
4	i, i, i, i	1
$2+2$	i, i, j, j	1

Only the final two patterns contribute. This is *why* 4-wise independence is sufficient.

Fourth moments: count the surviving terms

All four indices equal contribute $\sum_i f_i^4 = F_4$.

Fourth moments: count the surviving terms

All four indices equal contribute $\sum_i f_i^4 = F_4$. For $i \neq j$, there are $\binom{4}{2} = 6$ arrangements of i, i, j, j:

$$\mathbb{E}Z^4 = F_4 + 6 \sum_{i<j} f_i^2 f_j^2.$$

Fourth moments: count the surviving terms

All four indices equal contribute $\sum_i f_i^4 = F_4$. For $i \neq j$, there are $\binom{4}{2} = 6$ arrangements of i, i, j, j:

$$\mathbb{E}Z^4 = F_4 + 6 \sum_{i<j} f_i^2 f_j^2.$$

Compare with the expansion

$$F_2^2 = \left(\sum_i f_i^2 \right)^2 = F_4 + 2 \sum_{i<j} f_i^2 f_j^2.$$

Fourth moments: count the surviving terms

All four indices equal contribute $\sum_i f_i^4 = F_4$. For $i \neq j$, there are $\binom{4}{2} = 6$ arrangements of i, i, j, j:

$$\mathbb{E}Z^4 = F_4 + 6\sum_{i<j} f_i^2 f_j^2.$$

Compare with the expansion

$$F_2^2 = \left(\sum_i f_i^2\right)^2 = F_4 + 2\sum_{i<j} f_i^2 f_j^2.$$

Therefore

$$\mathbb{E}Z^4 = F_4 + 3(F_2^2 - F_4) = 3F_2^2 - 2F_4.$$

$$\boxed{\mathrm{Var}(Z^2) = 2(F_2^2 - F_4) \leq 2F_2^2.}$$

Averaging gives constant success

Use s independently seeded AMS copies $X_1, \dots, X_s$ and average:

$$Y = \frac{1}{s} \sum_{j=1}^{s} X_j, \qquad \mathbb{E}Y = F_2, \qquad \mathrm{Var}(Y) \le \frac{2F_2^2}{s}.$$

Averaging gives constant success

Use s independently seeded AMS copies $X_1, \dots, X_s$ and average:

$$Y = \frac{1}{s} \sum_{j=1}^{s} X_j, \qquad \mathbb{E}Y = F_2, \qquad \mathrm{Var}(Y) \le \frac{2F_2^2}{s}.$$

Chebyshev gives

$$\Pr(|Y - F_2| > \varepsilon F_2) \le \frac{2}{s\varepsilon^2}.$$

Averaging gives constant success

Use s independently seeded AMS copies $X_1, \dots, X_s$ and average:

$$Y = \frac{1}{s} \sum_{j=1}^{s} X_j, \qquad \mathbb{E}Y = F_2, \qquad \mathrm{Var}(Y) \le \frac{2F_2^2}{s}.$$

Chebyshev gives

$$\Pr(|Y - F_2| > \varepsilon F_2) \le \frac{2}{s\varepsilon^2}.$$

Taking $s = \lceil 8/\varepsilon^2 \rceil$ gives failure probability at most $1/4$. **Lesson:** Relative accuracy depends on the ratio $\mathrm{Var}(X)/(\mathbb{E}X)^2$; for AMS it is at most 2.

Amplify success by taking a median

Create r independent groups. In each group average $s = \lceil 8/\varepsilon^2 \rceil$ sketches, producing $Y_1, \ldots, Y_r$.

$$\boxed{\widehat{F}_2 = \text{median}(Y_1, \ldots, Y_r).}$$

Amplify success by taking a median

Create r independent groups. In each group average $s = \lceil 8/\varepsilon^2 \rceil$ sketches, producing $Y_1, \ldots, Y_r$.

$$\boxed{\widehat{F}_2 = \text{median}(Y_1, \ldots, Y_r).}$$

Every group is bad with probability at most $1/4$. The median can be bad only if at least half the groups are bad.

Amplify success by taking a median

Create r independent groups. In each group average $s = \lceil 8/\varepsilon^2 \rceil$ sketches, producing $Y_1, \ldots, Y_r$.

$$\boxed{\widehat{F}_2 = \mathrm{median}(Y_1, \ldots, Y_r).}$$

Every group is bad with probability at most $1/4$. The median can be bad only if at least half the groups are bad. By a Chernoff bound,

$$\Pr(\text{median is bad}) \le e^{-r/12}.$$

Take $r = \lceil 12 \ln(1/\delta) \rceil$. **Key pattern:** variance control $\rightarrow$ constant success $\rightarrow$ exponentially small failure.

AMS: the theorem and what is actually stored

Relative-error guarantee

Using $O(\varepsilon^{-2} \log(1/\delta))$ independent signed counters, AMS returns $(1 \pm \varepsilon)F_2$ with probability at least $1 - \delta$.

AMS: the theorem and what is actually stored

Relative-error guarantee

Using $O(\varepsilon^{-2} \log(1/\delta))$ independent signed counters, AMS returns $(1 \pm \varepsilon)F_2$ with probability at least $1 - \delta$.

Each counter satisfies $|Z| \leq N$, requiring $O(\log(N+1))$ bits. A 4-wise independent sign function over a universe of size u can be represented with $O(\log u)$ seed bits.

$$\text{Total space } = O\big(\varepsilon^{-2} \log(1/\delta)[\log u + \log(N+1)]\big) \text{ bits.}$$

AMS: the theorem and what is actually stored

Relative-error guarantee

Using $O(\varepsilon^{-2}\log(1/\delta))$ independent signed counters, AMS returns $(1 \pm \varepsilon)F_2$ with probability at least $1 - \delta$.

Each counter satisfies $|Z| \le N$, requiring $O(\log(N+1))$ bits. A 4-wise independent sign function over a universe of size u can be represented with $O(\log u)$ seed bits.

$$\text{Total space } = O\big(\varepsilon^{-2}\log(1/\delta)[\log u + \log(N+1)]\big) \text{ bits.}$$

N need not be known in advance: grow the integer counters as the stream grows.

PART 3

Flajolet–Martin: count distinct keys

3

Count unique visitors, not total visits

A repeated key should not change $D = F_0$. How can randomness make repetitions harmless?

Count unique visitors, not total visits

A repeated key should not change $D = F_0$. How can randomness make repetitions harmless? Assign every key x a *fixed* independent uniform hash value $h(x) \in (0, 1)$. **Crucial detail:** If the same key arrives again, it gets the same hash value.

Count unique visitors, not total visits

A repeated key should not change $D = F_0$. How can randomness make repetitions harmless? Assign every key x a *fixed* independent uniform hash value $h(x) \in (0, 1)$. **Crucial detail:** If the same key arrives again, it gets the same hash value. Let M be the minimum hash value among the D distinct observed keys. **Question:** What does a very small value of M tell us about D?

Why the minimum reveals the number of keys

For $0 \leq t \leq 1$, $M > t$ means *all* D distinct hashes exceed t:

$$\boxed{\Pr(M > t) = (1 - t)^D.}$$

Why the minimum reveals the number of keys

For $0 \le t \le 1$, $M > t$ means *all* D distinct hashes exceed t:

$$\boxed{\Pr(M > t) = (1-t)^D.}$$

Integrating the tail,

$$\mathbb{E}M = \int_0^1 (1-t)^D \, dt = \frac{1}{D+1}.$$

Why the minimum reveals the number of keys

For $0 \le t \le 1$, $M > t$ means *all* D distinct hashes exceed t:

$$\boxed{\Pr(M > t) = (1-t)^D.}$$

Integrating the tail,

$$\mathbb{E}M = \int_0^1 (1-t)^D \, dt = \frac{1}{D+1}.$$

So M is typically on the order of $1/D$. A minimum near 0.001 suggests about a thousand distinct keys, not ten. **Caution:** $1/M - 1$ is not unbiased; indeed $\mathbb{E}[\,1/M] = \infty$ under continuous ideal hashing. We use the minimum to motivate a better statistic.

From tiny hashes to leading zeros

View the hash as an infinite fair-bit expansion, e.g.

$$h(x) = 0.\underbrace{000}_{\rho(x)=3}1011\ldots \qquad \Longrightarrow \qquad 2^{-4} \le h(x) < 2^{-3}.$$

From tiny hashes to leading zeros

View the hash as an infinite fair-bit expansion, e.g.

$$h(x) = 0.\underbrace{000}_{\rho(x)=3}1011\ldots \qquad \Longrightarrow \qquad 2^{-4} \le h(x) < 2^{-3}.$$

Let $\rho(x)$ count the initial zero bits before the first 1. For $\rho(x) \ge r$ we need the first r bits to be zero:

$$\boxed{\Pr(\rho(x) \ge r) = 2^{-r}.}$$

From tiny hashes to leading zeros

View the hash as an infinite fair-bit expansion, e.g.

$$h(x) = 0.\underbrace{000}_{\rho(x)=3}1011\ldots \qquad \Longrightarrow \qquad 2^{-4} \le h(x) < 2^{-3}.$$

Let $\rho(x)$ count the initial zero bits before the first 1. For $\rho(x) \ge r$ we need the first r bits to be zero:

$$\boxed{\Pr(\rho(x) \ge r) = 2^{-r}.}$$

For $\rho(x) = r$ we need r zeros *followed by a one*:

$$\boxed{\Pr(\rho(x) = r) = 2^{-(r+1)}.}$$

How rare are long zero runs?

Out of D independently hashed distinct keys, the expected number with $\rho(x) \geq r$ is

$$\lambda_r = D\, 2^{-r}.$$

How rare are long zero runs?

Out of D independently hashed distinct keys, the expected number with $\rho(x) \geq r$ is

$$\lambda_r = D\,2^{-r}.$$

For $D = 1024$	$r = 5$	$r = 8$	$r = 10$	$r = 12$	$r = 15$
Expected keys with $\rho \geq r$	32	4	1	1/4	1/32

How rare are long zero runs?

Out of D independently hashed distinct keys, the expected number with $\rho(x) \geq r$ is

$$\lambda_r = D\,2^{-r}.$$

For $D = 1024$	$r = 5$	$r = 8$	$r = 10$	$r = 12$	$r = 15$
Expected keys with $\rho \geq r$	32	4	1	1/4	1/32

The largest observed ρ, denoted $R_{\max}$, should therefore lie near $\log_2 D$. In fact,

$$\Pr(R_{\max} < r) = (1 - 2^{-r})^D.$$

One extreme hash is unstable

A first attempt is $\widehat{D} \approx 2^{R_{\max}}$.

One extreme hash is unstable

A first attempt is $\widehat{D} \approx 2^{R_{\max}}$. Suppose the largest number of leading zeros changes from 10 to 12 because of one unusually small hash:

$$2^{10} = 1024 \qquad \longrightarrow \qquad 2^{12} = 4096.$$

One extreme hash is unstable

A first attempt is $\widehat{D} \approx 2^{R_{\max}}$. Suppose the largest number of leading zeros changes from 10 to 12 because of one unusually small hash:

$$2^{10} = 1024 \qquad \longrightarrow \qquad 2^{12} = 4096.$$

The true number of distinct keys could have increased by only one! **Design question:**

Can we remember more than just the single most extreme hash, while still using very little memory?

FM: remember which rare levels were seen

Keep a bitmap $B[0], B[1], \ldots, B[L-1]$, initially all zero.

Flajolet–Martin bitmap

On arrival of key x: set $\mathbf{B}[\rho(x)] \leftarrow 1$.

At query time: find R, the index of the *first unset bit*.

FM: remember which rare levels were seen

Keep a bitmap $B[0], B[1], \ldots, B[L-1]$, initially all zero.

Flajolet–Martin bitmap

On arrival of key x: set $\mathbf{B}[\rho(x)] \leftarrow 1$.

At query time: find R, the index of the *first unset bit*.

Example: arrivals with zero counts 0, 2, 1, 0, 4, 2, 1 produce

position r	0	1	2	3	4	5	6	7
$B[r]$	1	1	1	**0**	1	0	0	0

$R = 3$.

FM: remember which rare levels were seen

Keep a bitmap $B[0], B[1], \ldots, B[L-1]$, initially all zero.

Flajolet–Martin bitmap

On arrival of key x: set $\mathbf{B}[\rho(x)] \leftarrow 1$.

At query time: find R, the index of the *first unset bit*.

Example: arrivals with zero counts 0, 2, 1, 0, 4, 2, 1 produce

position r	0	1	2	3	4	5	6	7
$B[r]$	1	1	1	**0**	1	0	0	0

$R = 3$.

A repeated key hashes to the same level and sets the same bit. The bitmap thus depends only on the set of observed keys.

Why do the bitmap's early positions fill first?

For any single key, $\Pr(\rho(x) = r) = p_r = 2^{-(r+1)}$.

Why do the bitmap's early positions fill first?

For any single key, $\Pr(\rho(x) = r) = p_r = 2^{-(r+1)}$. A level stays empty only if all D keys miss it:

$$\boxed{\Pr(B[r] = 0) = (1 - p_r)^D = (1 - 2^{-(r+1)})^D.}$$

Why do the bitmap's early positions fill first?

For any single key, $\Pr(\rho(x) = r) = p_r = 2^{-(r+1)}$. A level stays empty only if all D keys miss it:

$$\boxed{\Pr(B[r] = 0) = (1 - p_r)^D = (1 - 2^{-(r+1)})^D.}$$

Write $\lambda_r = Dp_r = D/2^{r+1}$ for its expected number of hits.

$$\lambda_r \gg 1: \quad \Pr(B[r] = 0) \le e^{-\lambda_r} \quad \text{(almost surely filled)},$$
$$\lambda_r \ll 1: \quad \Pr(B[r] = 1) \le \lambda_r \quad \text{(usually empty)}.$$

Why do the bitmap's early positions fill first?

For any single key, $\Pr(\rho(x) = r) = p_r = 2^{-(r+1)}$. A level stays empty only if all D keys miss it:

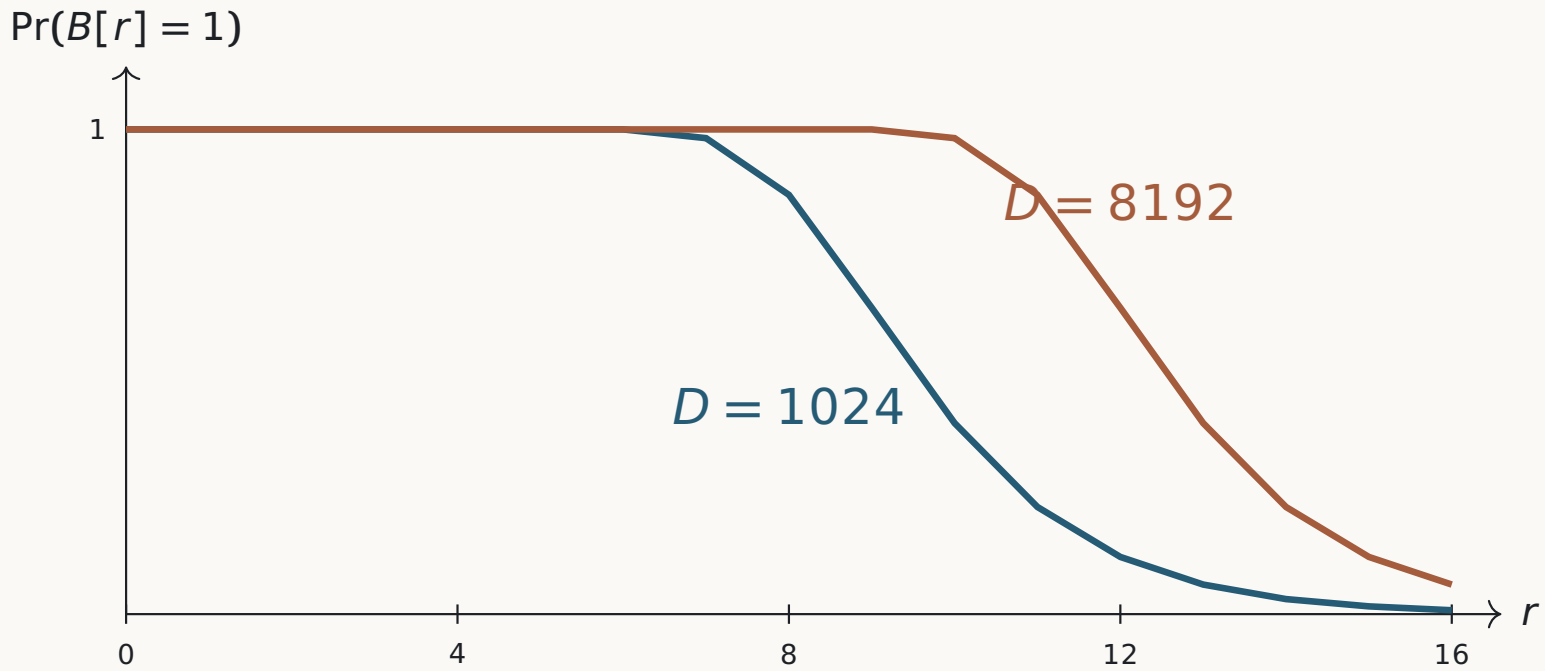
$$\boxed{\Pr(B[r] = 0) = (1 - p_r)^D = (1 - 2^{-(r+1)})^D.}$$

Write $\lambda_r = Dp_r = D/2^{r+1}$ for its expected number of hits.

$$\lambda_r \gg 1: \quad \Pr(B[r] = 0) \le e^{-\lambda_r} \quad \text{(almost surely filled)},$$
$$\lambda_r \ll 1: \quad \Pr(B[r] = 1) \le \lambda_r \quad \text{(usually empty)}.$$

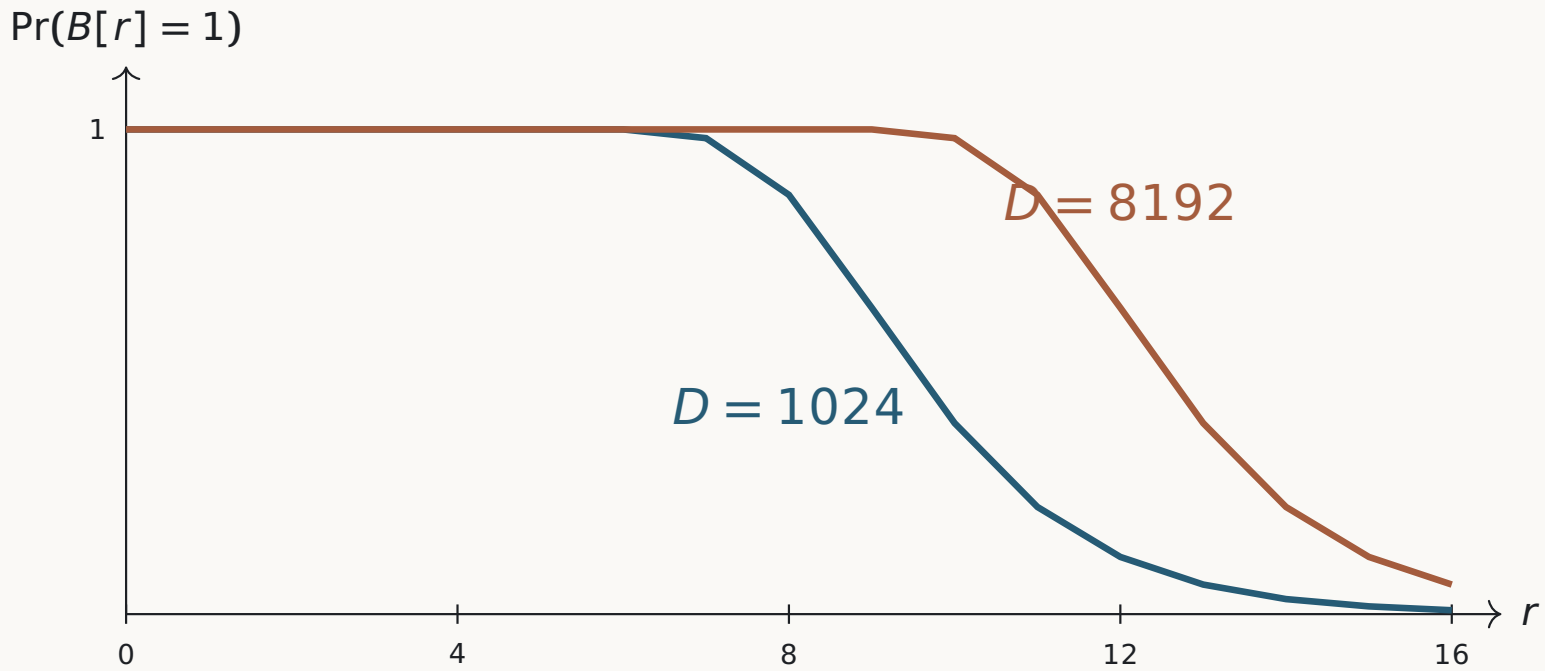
The occupancy transition occurs when $D/2^{r+1} \approx 1$, i.e., near $r \approx \log_2 D$.

A picture of the bitmap transition



Multiplying D by 8 shifts the occupancy transition roughly three positions right.

A picture of the bitmap transition



Multiplying D by 8 shifts the occupancy transition roughly three positions right. But the *first empty* position is influenced by all preceding bits. We need to bound that event separately.

Lower tail: the first empty bit is not too early

Let $R = \min\{j : B[j] = 0\}$. If $R < r$, at least one of bits $0, \ldots, r-1$ is empty.

Lower tail: the first empty bit is not too early

Let $R = \min\{j : B[j] = 0\}$. If $R < r$, at least one of bits $0, \ldots, r-1$ is empty. Using a union bound and $(1-p)^D \le e^{-Dp}$,

$$\Pr(R < r) \le \sum_{j=0}^{r-1} e^{-D/2^{j+1}}.$$

Lower tail: the first empty bit is not too early

Let $R = \min\{j : B[j] = 0\}$. If $R < r$, at least one of bits $0, \ldots, r-1$ is empty. Using a union bound and $(1-p)^D \le e^{-Dp}$,

$$\Pr(R < r) \le \sum_{j=0}^{r-1} e^{-D/2^{j+1}}.$$

Set $a = D/2^r$. The exponents in the sum are $a, 2a, 4a, \ldots$:

$$\boxed{\Pr(R < r) \le \sum_{k \ge 0} e^{-2^k a} \le \frac{e^{-a}}{1 - e^{-a}}.}$$

Take $r = \lfloor \log_2 D \rfloor - c$. Then $a \ge 2^c$, so the chance of seeing an empty bit *that early* is extremely small.

Upper tail: the first empty bit is not too late

If $R > r$, then bit r must have been hit by some key.

Upper tail: the first empty bit is not too late

If $R > r$, then bit r must have been hit by some key. A union bound over the D distinct keys gives

$$\boxed{\Pr(R > r) \le \Pr(B[r] = 1) \le D\, 2^{-(r+1)}.}$$

Upper tail: the first empty bit is not too late

If $R > r$, then bit r must have been hit by some key. A union bound over the D distinct keys gives

$$\boxed{\Pr(R > r) \le \Pr(B[r] = 1) \le D\, 2^{-(r+1)}.}$$

Take $r = \lceil \log_2 D \rceil + c$: then

$$\Pr(R > r) \le 2^{-c-1}.$$

Upper tail: the first empty bit is not too late

If $R > r$, then bit r must have been hit by some key. A union bound over the D distinct keys gives

$$\boxed{\Pr(R > r) \le \Pr(B[r] = 1) \le D\, 2^{-(r+1)}.}$$

Take $r = \lceil \log_2 D \rceil + c$: then

$$\Pr(R > r) \le 2^{-c-1}.$$

Together with the preceding slide, these inequalities show that

$$\boxed{R = \log_2 D + O(1) \quad \text{with high constant probability.}}$$

Thus 2^R gives a *constant-factor* indication of the distinct count, not yet an arbitrary-accuracy guarantee.

A calibrated FM estimate—and its limitations

The original FM bitmap estimator takes the form

$$\widehat{D} = 2^R/\phi, \qquad \phi \approx 0.7735.$$

A calibrated FM estimate—and its limitations

The original FM bitmap estimator takes the form

$$\widehat{D} = 2^R/\phi, \qquad \phi \approx 0.7735.$$

The factor ϕ comes from a finer asymptotic analysis of the location of R; our bounds establish the scale, *not* this calibration constant. A change of one in R doubles or halves 2^R, so a single bitmap is noisy.

A calibrated FM estimate—and its limitations

The original FM bitmap estimator takes the form

$$\widehat{D} = 2^R/\phi, \qquad \phi \approx 0.7735.$$

The factor ϕ comes from a finer asymptotic analysis of the location of R; our bounds establish the scale, *not* this calibration constant. A change of one in R doubles or halves 2^R, so a single bitmap is noisy. **Three caveats, specific to FM:**

- The analysis idealizes independently uniform hashes of distinct keys.
- A finite bitmap needs a sufficiently long hash and an overflow convention.
- A bitmap cannot undo arbitrary deletions: setting a bit loses multiplicity information.

Reduce noise using independent bitmaps

Maintain m bitmaps using independent hashes and obtain first-zero positions $R_1, \ldots, R_m$.

Reduce noise using independent bitmaps

Maintain m bitmaps using independent hashes and obtain first-zero positions $R_1, \ldots, R_m$. Average the *positions*, then exponentiate:

$$\bar{R} = \frac{1}{m} \sum_{j=1}^{m} R_j, \qquad \widehat{D}_{\text{geom}} = \frac{2^{\bar{R}}}{\phi}.$$

Reduce noise using independent bitmaps

Maintain m bitmaps using independent hashes and obtain first-zero positions $R_1, \ldots, R_m$. Average the *positions*, then exponentiate:

$$\bar{R} = \frac{1}{m} \sum_{j=1}^{m} R_j, \qquad \widehat{D}_{\text{geom}} = \frac{2^{\bar{R}}}{\phi}.$$

This is the geometric mean of the individual estimates $2^{R_j}/\phi$. **Why not just average 2^{R_j}?** Exponentiation makes unusually large R_j disproportionately influential. **Tradeoff:** each stream update must compute m independent hashes and update m bitmaps. Can we get several approximate independent experiments for the cost of one hash?

Stochastic averaging: one hash, many buckets

Choose $m = 2^b$ buckets. Hash each key once to a long bit string.

$$\underbrace{101}_{\text{bucket } j=5} \;\Big|\; \underbrace{00010110\ldots}_{\rho=3 \text{ in remaining bits}}$$

Stochastic averaging: one hash, many buckets

Choose $m = 2^b$ buckets. Hash each key once to a long bit string.

$$\underbrace{101}_{\text{bucket } j=5} \;\Big|\; \underbrace{00010110\ldots}_{\rho=3 \text{ in remaining bits}}$$

One update

Use the first b bits to choose bucket $j \in \{0, \ldots, m-1\}$.

Count leading zeros in the remaining bits, say r.

Set only $\mathbf{B}_j[r] \leftarrow 1$.

Stochastic averaging: one hash, many buckets

Choose $m = 2^b$ buckets. Hash each key once to a long bit string.

$$\underbrace{101}_{\text{bucket } j=5} \;\Big|\; \underbrace{00010110\ldots}_{\rho=3 \text{ in remaining bits}}$$

One update

Use the first b bits to choose bucket $j \in \{0, \ldots, m-1\}$.

Count leading zeros in the remaining bits, say r.

Set only $\mathbf{B}_j[r] \leftarrow 1$.

Each distinct key chooses one bucket, so a bucket receives on average D/m distinct keys. Repeats choose the same bucket and level.
Advantage: one hash calculation and one bitmap update per arrival, regardless of m.

A complete stochastic-averaging example

Suppose $m = 4$ buckets, and the first-zero positions after all updates are

bucket j	0	1	2	3
R_j	3	4	2	3
rough distinct-count scale 2^{R_j}	8	16	4	8

A complete stochastic-averaging example

Suppose $m = 4$ buckets, and the first-zero positions after all updates are

bucket j	0	1	2	3
R_j	3	4	2	3
rough distinct-count scale 2^{R_j}	8	16	4	8

The geometric mean of the four scales is

$$2^{\bar{R}} = 2^{(3+4+2+3)/4} = 2^3 = 8.$$

A complete stochastic-averaging example

Suppose $m = 4$ buckets, and the first-zero positions after all updates are

bucket j	0	1	2	3
R_j	3	4	2	3
rough distinct-count scale 2^{R_j}	8	16	4	8

The geometric mean of the four scales is

$$2^{\bar{R}} = 2^{(3+4+2+3)/4} = 2^3 = 8.$$

Each bucket sees about D/m keys. Scaling by m gives the global estimate

$$\boxed{\widehat{D} \approx m \frac{2^{\bar{R}}}{\phi} = \frac{4 \cdot 8}{0.7735} \approx 41.4.}$$

This is an asymptotic FM calibration; small buckets, finite hashes, and the randomness of bucket sizes require additional care for a formal accuracy guarantee.

What does stochastic averaging buy us?

	independent bitmaps	stochastic averaging
Hash evaluations per update	m	1
Bitmap updates per arrival	m	1
Expected distinct keys per bitmap	D	D/m
Repetitions of the same key	same level in each bitmap	same bucket and level

What does stochastic averaging buy us?

	independent bitmaps	stochastic averaging
Hash evaluations per update	m	1
Bitmap updates per arrival	m	1
Expected distinct keys per bitmap	D	D/m
Repetitions of the same key	same level in each bitmap	same bucket and level

Both use m bitmaps, but stochastic averaging trades independent copies of the *whole stream* for a random partition of distinct keys. The estimator

is most informative when enough distinct keys populate the buckets. Precise finite-sample relative-error bounds need more analysis than the elementary single-bitmap bounds we proved.

FM: space and assumptions

For L levels and m buckets, the bitmap uses mL bits, plus the hash seed.

FM: space and assumptions

For L levels and m buckets, the bitmap uses mL bits, plus the hash seed. $L = O(\log D)$ levels cover the typical occupied range in the idealized model; practical implementations choose a fixed hash length and handle exhaustion/overflow explicitly. **Keep these distinctions clear:**

- Storing an index R costs $O(\log L)$ bits, but storing a *bitmap* costs L bits.
- Ideal independent hashes simplify the theory; finite pseudorandom hashes need their own guarantees.
- Stochastic averaging improves *update time*, while averaging many buckets is intended to improve precision.

FM: space and assumptions

For L levels and m buckets, the bitmap uses mL bits, plus the hash seed. $L = O(\log D)$ levels cover the typical occupied range in the idealized model; practical implementations choose a fixed hash length and handle exhaustion/overflow explicitly. **Keep these distinctions clear:**

- Storing an index R costs $O(\log L)$ bits, but storing a *bitmap* costs L bits.
- Ideal independent hashes simplify the theory; finite pseudorandom hashes need their own guarantees.
- Stochastic averaging improves *update time*, while averaging many buckets is intended to improve precision.

Modern distinct sketches further compress and calibrate the bucket summaries (HyperLogLog: appendix).

Two sketches, two ideas

	F_2: AMS	F_0: FM
Core trick	Random signs cancel cross terms	Rare hash prefixes expose count scale
State	Signed counters	Bitmap of observed zero-run levels
Proof idea	Fourth-moment bound; Chebyshev	Occupancy; union bounds
Noise reduction	Average, then median	Multiple bitmaps; stochastic averaging
Update behavior	Signed additive update	Set one bit; repeats do nothing

Two sketches, two ideas

	F_2: AMS	F_0: FM
Core trick	Random signs cancel cross terms	Rare hash prefixes expose count scale
State	Signed counters	Bitmap of observed zero-run levels
Proof idea	Fourth-moment bound; Chebyshev	Occupancy; union bounds
Noise reduction	Average, then median	Multiple bitmaps; stochastic averaging
Update behavior	Signed additive update	Set one bit; repeats do nothing

Check your understanding: If each key in a stream appears a second time, how do F_0, F_2, and the two sketches change?

Questions?

Appendix: further directions

- Higher moments: a sampling estimator and why its variance can be large.
- Other frequency statistics: stable sketches, entropy, and limits of naive expansion.
- Modern distinct counting: HyperLogLog and k-minimum values.

The appendix is supplementary; these results were not developed in the 110-minute lecture.

Many useful statistics are sums $\sum_x g(f_x)$

$g(t)$	$\sum_x g(f_x)$	what it is used for
$\mathbf{1}\{t > 0\}$	F_0	number of distinct keys
t^2	F_2	skew, collision probability, $\|f - f'\|_2^2$
t^k, k large	$F_k \approx (\max_x f_x)^k$	largest count $\approx F_k^{1/k}$
$t \ln t$	$N\big(\ln N - H(f/N)\big)$	entropy of the key distribution
$\min(t, T)$	capped count	ad reach, capped at T per user

Many useful statistics are sums $\sum_x g(f_x)$

$g(t)$	$\sum_x g(f_x)$	what it is used for
$\mathbf{1}\{t > 0\}$	F_0	number of distinct keys
t^2	F_2	skew, collision probability, $\|f - f'\|_2^2$
t^k, k large	$F_k \approx (\max_x f_x)^k$	largest count $\approx F_k^{1/k}$
$t \ln t$	$N\big(\ln N - H(f/N)\big)$	entropy of the key distribution
$\min(t, T)$	capped count	ad reach, capped at T per user

Entropy is a standard network-anomaly signal: a port scan spreads traffic over many destination ports (entropy up), a flood concentrates it (entropy down).

Many useful statistics are sums $\sum_x g(f_x)$

$g(t)$	$\sum_x g(f_x)$	what it is used for
$\mathbf{1}\{t > 0\}$	F_0	number of distinct keys
t^2	F_2	skew, collision probability, $\|f - f'\|_2^2$
t^k, k large	$F_k \approx (\max_x f_x)^k$	largest count $\approx F_k^{1/k}$
$t \ln t$	$N\big(\ln N - H(f/N)\big)$	entropy of the key distribution
$\min(t, T)$	capped count	ad reach, capped at T per user

Entropy is a standard network-anomaly signal: a port scan spreads traffic over many destination ports (entropy up), a flood concentrates it (entropy down).

Different frequency statistics require different estimators; there is no universal small-space sketch for arbitrary g.

Many useful statistics are sums $\sum_x g(f_x)$

$g(t)$	$\sum_x g(f_x)$	what it is used for
$\mathbf{1}\{t > 0\}$	F_0	number of distinct keys
t^2	F_2	skew, collision probability, $\|f - f'\|_2^2$
t^k, k large	$F_k \approx (\max_x f_x)^k$	largest count $\approx F_k^{1/k}$
$t \ln t$	$N\big(\ln N - H(f/N)\big)$	entropy of the key distribution
$\min(t, T)$	capped count	ad reach, capped at T per user

Entropy is a standard network-anomaly signal: a port scan spreads traffic over many destination ports (entropy up), a flood concentrates it (entropy down).

Different frequency statistics require different estimators; there is no universal small-space sketch for arbitrary g.

Questions for the rest of this part: can we estimate F_k for $k > 2$? Arbitrary g? What does it cost?

Can the sign trick give F_4?

We computed $\mathbb{E}[\mathbf{Z}^4] = 3F_2^2 - 2F_4$. So F_4 is in there, entangled with F_2^2.

Can the sign trick give F_4?

We computed $\mathbb{E}[\mathbf{Z}^4] = 3F_2^2 - 2F_4$. So F_4 is in there, entangled with F_2^2.

Try $\widehat{F}_4 = \frac{1}{2}\big(3\widehat{F_2^2} - \widehat{\mathbb{E}\mathbf{Z}^4}\big)$. The trouble is scale. On an all-distinct stream of n keys,

$$F_4 = n, \qquad F_2^2 = n^2.$$

We would need to subtract two quantities of size n^2 and get their difference to within εn: relative accuracy ε/n on each, i.e. $\sim n^2/\varepsilon^2$ copies.

Can the sign trick give F_4?

We computed $\mathbb{E}[\mathbf{Z}^4] = 3F_2^2 - 2F_4$. So F_4 is in there, entangled with F_2^2.

Try $\widehat{F}_4 = \frac{1}{2}\big(3\widehat{F_2^2} - \widehat{\mathbb{E}\mathbf{Z}^4}\big)$. The trouble is scale. On an all-distinct stream of n keys,

$$F_4 = n, \qquad F_2^2 = n^2.$$

We would need to subtract two quantities of size n^2 and get their difference to within εn: relative accuracy ε/n on each, i.e. $\sim n^2/\varepsilon^2$ copies.

The sign trick worked for F_2 because the spread of $\mathbf{Z}^2$ is on the scale of F_2. For F_k, $k > 2$, signs produce spreads on the scale of $F_2^{k/2}$, which can be much larger than F_k. We need a different estimator.

A different idea: sample a position, look ahead

Pick a position J uniformly from $\{1, \ldots, N\}$. Let $x = a_J$ and

$$r = \#\{t \geq J : \ a_t = x\} \qquad \text{(occurrences of } x \text{ from } J \text{ to the end)}.$$

Output $X = N\,(r^k - (r-1)^k)$.

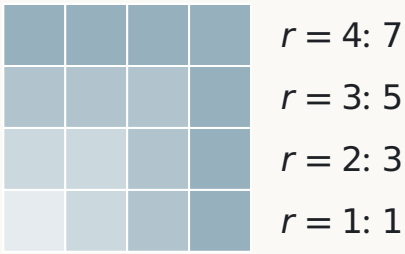
A different idea: sample a position, look ahead

Pick a position J uniformly from $\{1, \ldots, N\}$. Let $x = a_J$ and

$$r = \#\{t \geq J : \ a_t = x\} \qquad \text{(occurrences of } x \text{ from } J \text{ to the end).}$$

Output $X = N\,(r^k - (r-1)^k)$.

Why it is unbiased. The f_x occurrences of x have $r = f_x, f_x - 1, \ldots, 1$. Summing their contributions telescopes:



For $k = 2$, key A with $f_A = 4$: the four occurrences contribute $7, 5, 3, 1$, the L-shaped layers of a 4×4 square.

$$\mathbb{E}X = \frac{1}{N} \cdot N \sum_x \sum_{r=1}^{f_x} \left(r^k - (r-1)^k\right) = \sum_x f_x^k.$$

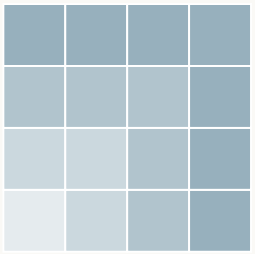
A different idea: sample a position, look ahead

Pick a position J uniformly from $\{1, \ldots, N\}$. Let $x = a_J$ and

$$r = \#\{t \geq J : \ a_t = x\} \qquad \text{(occurrences of } x \text{ from } J \text{ to the end).}$$

Output $X = N\,(r^k - (r-1)^k)$.

Why it is unbiased. The f_x occurrences of x have $r = f_x, f_x - 1, \ldots, 1$. Summing their contributions telescopes:



For $k = 2$, key A with $f_A = 4$: the four occurrences contribute $7, 5, 3, 1$, the L-shaped layers of a 4×4 square.

$$\mathbb{E}X = \frac{1}{N} \cdot N \sum_x \sum_{r=1}^{f_x} \left(r^k - (r-1)^k\right) = \sum_x f_x^k.$$

Nothing used $g(t) = t^k$ except $g(0) = 0$: $N\big(g(r) - g(r-1)\big)$ is unbiased for $\sum_x g(f_x)$. This is how entropy is estimated in streams, with $g(t) = t \ln t$.

Running it in one pass

We do not know N in advance, and r counts *future* occurrences. Both are fine:

One copy

Keep a sampled key x, a counter r, and the length t so far. On arrival of a_t:

- with probability $1/t$, set $x \leftarrow a_t$ and $r \leftarrow 1$ (reservoir sampling: x is a uniform position);
- otherwise, if $a_t = x$, set $r \leftarrow r + 1$.

Output $N\,(g(r) - g(r-1))$.

Running it in one pass

We do not know N in advance, and r counts *future* occurrences. Both are fine:

One copy

Keep a sampled key x, a counter r, and the length t so far. On arrival of a_t:

- with probability $1/t$, set $x \leftarrow a_t$ and $r \leftarrow 1$ (reservoir sampling: x is a uniform position);
- otherwise, if $a_t = x$, set $r \leftarrow r + 1$.

Output $N\,(g(r) - g(r-1))$.

Example: A, B, A, A, C, B, A, $k = 2$, $N = 7$.

J	1	2	3	4	5	6	7
a_J	A	B	A	A	C	B	A
r	4	2	3	2	1	1	1
$X = 7(2r-1)$	49	21	35	21	7	7	7

Average over J: $\frac{147}{7} = 21 = F_2$.

The price of higher moments

For $g(t) = t^k$ one can show (appendix)

$$\mathrm{Var}(X) \le k\, n^{1-1/k} F_k^2, \qquad n = \text{number of possible keys.}$$

Median of means then needs $O\big(k\, n^{1-1/k} \varepsilon^{-2} \log(1/\delta)\big)$ copies.

The price of higher moments

For $g(t) = t^k$ one can show (appendix)

$$\operatorname{Var}(X) \le k\, n^{1-1/k} F_k^2, \qquad n = \text{number of possible keys.}$$

Median of means then needs $O\big(k\, n^{1-1/k} \varepsilon^{-2} \log(1/\delta)\big)$ copies.

- For $k = 2$ this is $\sqrt{n}$ copies, versus $O(1)$ for AMS signs. The sign trick is special to F_2.
- For $k > 2$, polynomial space is unavoidable: any algorithm needs $\Omega(n^{1-2/k})$ bits (via communication complexity), and $\tilde{O}(n^{1-2/k})$ is achievable (Indyk–Woodruff).

The price of higher moments

For $g(t) = t^k$ one can show (appendix)

$$\mathrm{Var}(X) \le k\, n^{1-1/k} F_k^2, \qquad n = \text{number of possible keys.}$$

Median of means then needs $O\big(k\, n^{1-1/k} \varepsilon^{-2} \log(1/\delta)\big)$ copies.

- For $k = 2$ this is $\sqrt{n}$ copies, versus $O(1)$ for AMS signs. The sign trick is special to F_2.
- For $k > 2$, polynomial space is unavoidable: any algorithm needs $\Omega(n^{1-2/k})$ bits (via communication complexity), and $\tilde{O}(n^{1-2/k})$ is achievable (Indyk–Woodruff).

Intuition: as k grows, F_k is dominated by the few heaviest keys. Estimating it is close to *finding* the maximum frequency, which a small sketch cannot do when the heaviest key is only modestly above the rest.

The price of higher moments

For $g(t) = t^k$ one can show (appendix)

$$\mathrm{Var}(X) \le k\, n^{1-1/k} F_k^2, \qquad n = \text{number of possible keys.}$$

Median of means then needs $O\big(k\, n^{1-1/k} \varepsilon^{-2} \log(1/\delta)\big)$ copies.

- For $k = 2$ this is $\sqrt{n}$ copies, versus $O(1)$ for AMS signs. The sign trick is special to F_2.
- For $k > 2$, polynomial space is unavoidable: any algorithm needs $\Omega(n^{1-2/k})$ bits (via communication complexity), and $\tilde{O}(n^{1-2/k})$ is achievable (Indyk–Woodruff).

Intuition: as k grows, F_k is dominated by the few heaviest keys. Estimating it is close to *finding* the maximum frequency, which a small sketch cannot do when the heaviest key is only modestly above the rest.

So: moments $0 < k \le 2$ are cheap (polylog space); moments $k > 2$ are not.

F_p for $0 < p \leq 2$: replace signs with stable variabl

Why is $p \leq 2$ the cheap range? Signs mimic Gaussians, and Gaussians have this property:

$$G_x \text{ i.i.d. } \mathcal{N}(0,1) \quad \Longrightarrow \quad \sum_x f_x G_x \sim \|f\|_2 \cdot \mathcal{N}(0,1).$$

A linear sketch whose distribution is a known shape scaled by $\|f\|_2$.

F_p for $0 < p \leq 2$: replace signs with stable variabl

Why is $p \leq 2$ the cheap range? Signs mimic Gaussians, and Gaussians have this property:

$$G_x \text{ i.i.d. } \mathcal{N}(0,1) \quad \Longrightarrow \quad \sum_x f_x G_x \;\sim\; \|f\|_2 \cdot \mathcal{N}(0,1).$$

A linear sketch whose distribution is a known shape scaled by $\|f\|_2$.

p-stable laws do the same for $\|f\|_p$, and exist exactly for $0 < p \leq 2$ (Cauchy is $p = 1$):

$$\sum_x f_x S_x \;\sim\; \|f\|_p \cdot S.$$

Estimate $\|f\|_p$ by $\operatorname{median}_i |Z_i| / \operatorname{median} |S|$ over $O(\varepsilon^{-2} \log(1/\delta))$ copies (Indyk).

F_p for $0 < p \leq 2$: replace signs with stable variabl

Why is $p \leq 2$ the cheap range? Signs mimic Gaussians, and Gaussians have this property:

$$G_x \text{ i.i.d. } \mathcal{N}(0,1) \quad \Longrightarrow \quad \sum_x f_x G_x \;\sim\; \|f\|_2 \cdot \mathcal{N}(0,1).$$

A linear sketch whose distribution is a known shape scaled by $\|f\|_2$.

p-stable laws do the same for $\|f\|_p$, and exist exactly for $0 < p \leq 2$ (Cauchy is $p = 1$):

$$\sum_x f_x S_x \;\sim\; \|f\|_p \cdot S.$$

Estimate $\|f\|_p$ by $\operatorname{median}_i |Z_i| / \operatorname{median} |S|$ over $O(\varepsilon^{-2} \log(1/\delta))$ copies (Indyk).

For $p > 2$ no such distribution exists, matching the lower bound on the previous slide.

What about Taylor series?

If $g(t) = \sum_k c_k t^k$, then $\sum_x g(f_x) = \sum_k c_k F_k$. Estimate each moment and add?

What about Taylor series?

If $g(t) = \sum_k c_k t^k$, then $\sum_x g(f_x) = \sum_k c_k F_k$. Estimate each moment and add? Three problems:

- F_k for $k > 2$ costs $n^{1-2/k}$ space, and high-order terms need large k.
- f_x ranges over $[1, N]$; a series in t will not be accurate across that whole range for $g(t) = t \ln t$.
- Terms with alternating signs cancel, so relative errors on each F_k become huge absolute errors in the sum.

What about Taylor series?

If $g(t) = \sum_k c_k t^k$, then $\sum_x g(f_x) = \sum_k c_k F_k$. Estimate each moment and add? Three problems:

- F_k for $k > 2$ costs $n^{1-2/k}$ space, and high-order terms need large k.
- f_x ranges over $[1, N]$; a series in t will not be accurate across that whole range for $g(t) = t \ln t$.
- Terms with alternating signs cancel, so relative errors on each F_k become huge absolute errors in the sum.

The approximation-theory idea is right; the variable we expand in is wrong.

Expand in the moment order instead

Entropy needs $\sum_x f_x \ln f_x$. Since $\frac{d}{dp} f^p = f^p \ln f$,

$$\sum_x f_x \ln f_x = \frac{d}{dp} F_p \Big|_{p=1} \approx \frac{F_{1+\eta} - F_{1-\eta}}{2\eta}.$$

Both moments are in the cheap range $p \leq 2$.

Expand in the moment order instead

Entropy needs $\sum_x f_x \ln f_x$. Since $\frac{d}{dp} f^p = f^p \ln f$,

$$\sum_x f_x \ln f_x = \frac{d}{dp} F_p \Big|_{p=1} \approx \frac{F_{1+\eta} - F_{1-\eta}}{2\eta}.$$

Both moments are in the cheap range $p \le 2$.
The catch: dividing by 2η amplifies sketch error by $1/\eta$, while small η is needed for a small truncation error. Harvey–Nelson–Onak balance this with polynomial interpolation at several p near 1, giving polylog-space entropy estimation.

Expand in the moment order instead

Entropy needs $\sum_x f_x \ln f_x$. Since $\frac{d}{dp} f^p = f^p \ln f$,

$$\sum_x f_x \ln f_x = \frac{d}{dp} F_p \Big|_{p=1} \approx \frac{F_{1+\eta} - F_{1-\eta}}{2\eta}.$$

Both moments are in the cheap range $p \le 2$.
The catch: dividing by 2η amplifies sketch error by $1/\eta$, while small η is needed for a small truncation error. Harvey–Nelson–Onak balance this with polynomial interpolation at several p near 1, giving polylog-space entropy estimation.

More generally, Braverman–Ostrovsky characterize which g admit polylog-space sketches; roughly, g must grow at most quadratically and not change too abruptly.

Sampling F_k: variance bound

With $X = N(r^k - (r-1)^k)$ and J uniform,

$$\mathbb{E}[X^2] = N \sum_x \sum_{r=1}^{f_x} \left(r^k - (r-1)^k\right)^2.$$

Use $r^k - (r-1)^k \le k\, r^{k-1} \le k f_x^{k-1}$ on one factor, then telescope the other:

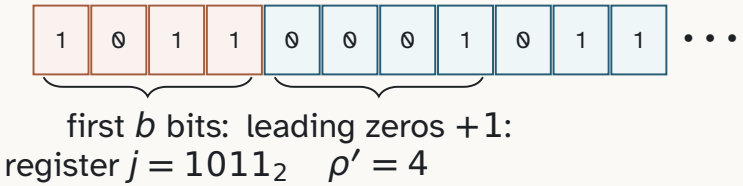
$$\mathbb{E}[X^2] \le N \sum_x k f_x^{k-1} \cdot f_x^k = k\, F_1 F_{2k-1}.$$

A Hölder-type inequality gives $F_1 F_{2k-1} \le n^{1-1/k} F_k^2$ for any $f \in \mathbb{N}^n$. Hence

$$\operatorname{Var}(X) \le \mathbb{E}[X^2] \le k\, n^{1-1/k} F_k^2.$$

HyperLogLog: one hash, many registers

Hash each key to a 64-bit string and split it:



- There are $m = 2^b$ registers $M_1, \ldots, M_m$, all 0 initially.
- On key x: $M_j \leftarrow \max(M_j, \rho')$.
- Each register sees about D/m distinct keys, so 2^{M_j} is a rough (factor-2) estimate of D/m.

This is the "largest ρ" estimator from the main lecture, run m times in parallel for the cost of one hash. Each register needs only 6 bits.

HyperLogLog: harmonic mean

Each 2^{M_j} estimates D/m, but its error is lopsided: one lucky key can double a register, while a register can never fall far *below* its typical value. A few large outliers would dominate an arithmetic mean.

HyperLogLog: harmonic mean

Each 2^{M_j} estimates D/m, but its error is lopsided: one lucky key can double a register, while a register can never fall far *below* its typical value. A few large outliers would dominate an arithmetic mean.
The harmonic mean damps large values:

$$\widehat{D} = \alpha_m \, m \cdot \underbrace{\frac{m}{\sum_{j=1}^{m} 2^{-M_j}}}_{\text{harmonic mean of } 2^{M_j}}, \qquad \alpha_m \approx 0.7213/(1 + 1.079/m).$$

(α_m corrects the bias, as ϕ did for FM.)

HyperLogLog: harmonic mean

Each 2^{M_j} estimates D/m, but its error is lopsided: one lucky key can double a register, while a register can never fall far *below* its typical value. A few large outliers would dominate an arithmetic mean.
The harmonic mean damps large values:

$$\widehat{D} = \alpha_m \, m \cdot \underbrace{\frac{m}{\sum_{j=1}^{m} 2^{-M_j}}}_{\text{harmonic mean of } 2^{M_j}}, \qquad \alpha_m \approx 0.7213/(1 + 1.079/m).$$

(α_m corrects the bias, as ϕ did for FM.)

- Relative standard error $\approx 1.04/\sqrt{m}$. With $m = 2^{14}$: about 0.8%, in 12 KB.
- **Small counts:** when many registers are still 0, switch to *linear counting*, $\widehat{D} = m \ln(m/V)$ with V the number of zero registers (a balls-in-bins estimate).
- **Mergeable:** the sketch of a union of two streams is the entrywise max of their registers. Distributed systems count distinct keys per machine and merge.

KMV: retain k minimum hashes

The darts picture also gives a simpler estimator with clean analysis.

Keep the k smallest distinct hash values seen. If v_k is the k-th smallest, then k of the D darts landed in $[0, v_k]$, so $D \approx k/v_k$. The unbiased version is

$$\widehat{D} = \frac{k-1}{v_k},$$

with relative standard error about $1/\sqrt{k-2}$.

KMV: retain k minimum hashes

The darts picture also gives a simpler estimator with clean analysis.

Keep the k smallest distinct hash values seen. If v_k is the k-th smallest, then k of the D darts landed in $[0, v_k]$, so $D \approx k/v_k$. The unbiased version is

$$\widehat{D} = \frac{k-1}{v_k},$$

with relative standard error about $1/\sqrt{k-2}$.

Trade-off against HyperLogLog: KMV stores k hash values of ~ 64 bits each, HyperLogLog stores m registers of 6 bits each, for similar accuracy. In exchange, KMV supports intersections and other set operations more naturally.