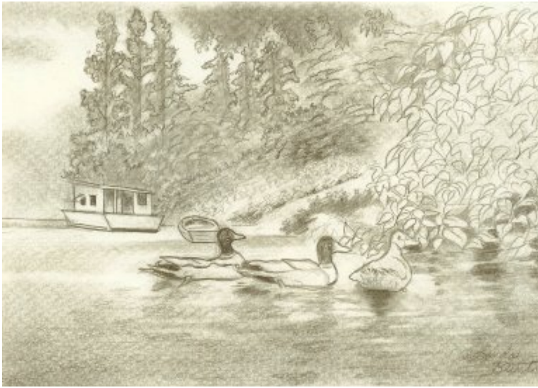
Introduction to Data Stream Processing

Sumit Ganguly

IIT Kanpur



Outline

Data Sources

Many applications have data sources that send data continuously and at fast speeds.

1. Network switch data: sequence of records with schema: *(source-IP, dest-IP, port, Packet-Type, DATA)*.
2. Web-Server: access data
3. Supermarket transaction data, financial markets data,
4. Satellite imagery (or other imagery) data,
5. Sensor network data, etc..

Type of analysis

There are generally two kinds of analyses.

1. Deep Analysis on stored data, typically, data mining applications.
2. Very fast online analysis, with some probability of error. Goal is to find outliers of some kind.

We will look at the second class. Applications needing continuous analysis for detection of anomalies, extremal scenarios, etc.. Some examples are

Data Stream Application Queries

Queries:

- Is there a denial of service attack in progress? (Network Monitoring)
- Is any IP-range sending/receiving much more traffic than usual? (Network Monitoring).
- From images, say quickly if an image is likely to be "similar" to known cases of problem images. Problem images can be (a) adverse weather disturbance, (b) known pathological medical images, etc..

Data Streams: Model

- Analysis has to be continuous and must keep pace with the data.
- Not enough time to store on secondary storage, and update secondary indices and then process.
- Analyze on the fly: Keep a summary, called *sketch*, in memory/cache.
- Update sketch corresponding to each stream record.
- Answer queries from the sketch on demand in real-time.

Data Streams: Model

- Item domain $[n] = \{1, 2, \dots, n\}$, n is large: e.g., $2^{64} \dots 2^{256} \dots$, IP-addresses, pairs of IP-addresses, URL's etc.
- Stream = sequence of updates of the form (*item*, *change in frequency*) $\equiv (i, v)$.

$$(1,1)\ (4,1)\ (5,3)\ (7,1)\ (5,-1)\ (5,2)\ (7,2)\ (6,1)\ (1,-1) \dots$$

Frequency Vector

	0	0	0	1	4	1	3	0	0	...
Items	1	2	3	4	5	6	7	8	9	

- initially $f = 0$.
- When (i, v) arrives: $f_i \leftarrow f_i + v$.

Global: $f_i = \sum_{(i,v) \in \text{ stream}} v$.

Data Streaming: Algorithmic Model

- Single pass over stream (Online algorithm).
- Sublinear storage: e.g., $o(n)$, $O(\sqrt{n})$, or, better $\log n$, $\log^2 n$, etc..
- Fast processing per arriving stream record.
  - Approximate processing (almost always necessary).
  - Randomized computation (almost always necessary).
- Multi-pass computations, e.g., for graph streaming applications.

Outline

Problem Definition

- The second moment of the frequency vector f is defined as

$$F_2 = \sum_{i \in [n]} |f_i|^2 = \|f\|_2^2 \ .$$

- We are given accuracy parameter ϵ.
- Deterministic Solution: Return $\hat{F}_2$ such that $\hat{F}_2 \in (1 \pm \epsilon) F_2$. Requires $\Omega(n)$ space (later).
- Randomized Solution: $\hat{F}_2 \in (1 \pm \epsilon) F_2$ with probability $1 - \delta$, δ is failure probability parameter.
- Two solutions: Alon, Matias, Szegedy and ℓ_2 dimensionality reduction: Johnson-Lindenstrauss.

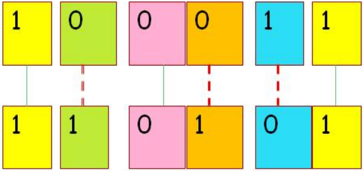
Outline

Basic notions of codes

- Weight *wt* of a binary vector *v* is defined the number of positions in *v* with 1. For e.g.,

$$wt(1\ 0\ 0\ 1) = 2,\ wt(1\ 0\ 0\ 0) = 1$$

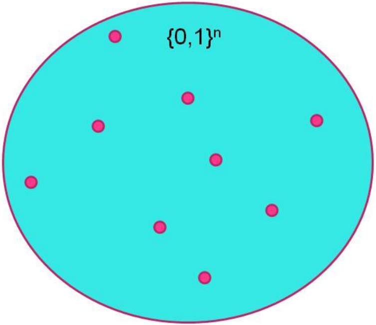
- Let y and z be n-dimensional vectors. Then, the Hamming distance $d_H(y, z) =$ number of positions i where $y_i \neq z_i$.
- It is a metric, $d_H \geq 0$, symmetric and satisfies the triangle inequality $d(x, y) + d(y, z) \geq d(x, z)$.



Hamming Distance is the number of pairings in red, which is 3

Codes..basics

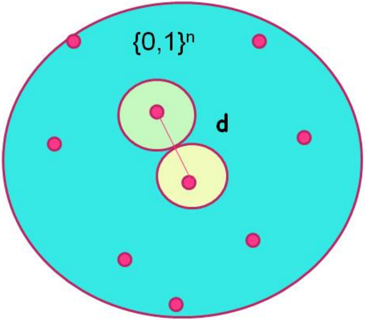
- A (binary) code is a set of binary vectors. It is sometimes useful to visualize a code as some subset of $\{0,1\}^n$. Each point "codes" or "represents" some input vector in $\{0,1\}^k$, where, $k \leq n$.



Points in red are the codes

Minimum distance of code

The smallest Hamming distance between any pair of codewords is called the *distance* of the code. Radius $=$smallest integer less than distance of code /2.



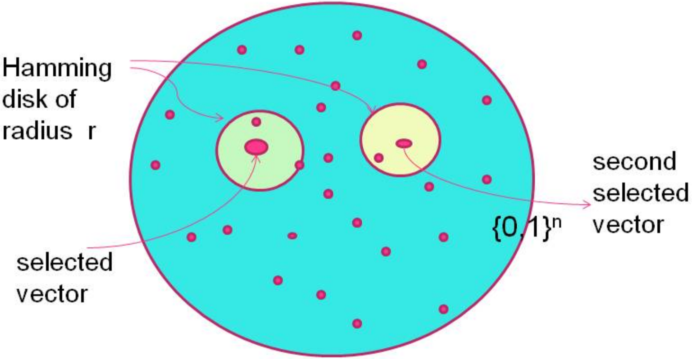
Smallest pair-wise distance is the code distance.
Error correcting radius < d/2.
Decoding method: map to nearest code-word (by Hamming Distance).

A useful code for us

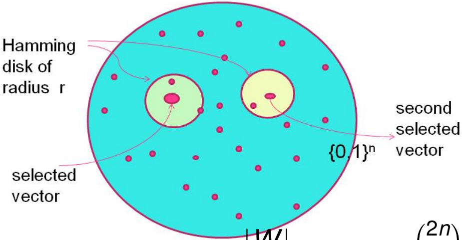
- Connection: existence of some codes can show non-existence of some algorithms>
- Consider a code $C \subset \{0,1\}^{2n}$ with vectors of weight n and radius$(C) \geq n/4$.
- Such codes of size $|C| = 2^{\Omega(n)}$ exist. Let us see.

Existence of code

- Consider set W of vectors from $\{0,1\}^{2n}$ of weight n.
- Choose any $y \in W$. Remove all vectors from Y that are in ball of vectors centered at y and radius $< dn$: $B_{dn}(y)$. Choose y_1 from remaining set, remove $B_{dn}(y)$, so on...



Code size calculation

- 

$$|C| = \frac{|W|}{|B_{n/4}|} = \frac{\binom{2n}{n}}{\sum_{1 \le r \le n/8} \binom{n}{r}^2} = 2^{cn}$$

  $B_{n/4}$ is size of Hamming ball of radius $n/4$ centered at some vector.

- This is a special case of the Gilbert-Varshamov bound in coding theory.
- For $y \in C$, $F_2(y) = \|y\|_2^2 = wt(y) = n$.

Deterministic evaluation of F_2

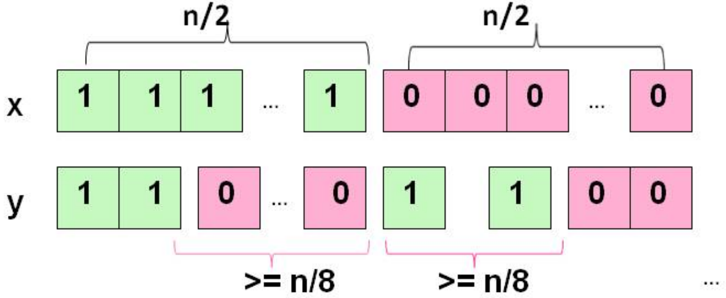
Proof idea

We would like to show that for any deterministic algorithm that estimates F_2 correctly to within 1 ± 0.01 must map distinct vectors from the code C to distinct memory images.

- Choose x, y two vectors from the code C.
- Suppose there is a deterministic algorithm A that gives $\hat{F}_2 \in (1 \pm 0.01)F_2$.
- If x and y are mapped to the same memory image by A, then, $x + x$ is mapped to the same image as $y + x$.

Lower bound-II

- Clearly, $\|2x\|^2 = 4\|x\|^2 = 4n$.
- Since $d_H(x, y) > n/4$:
  $\|x + y\|_2^2 < 4(n - n/8) + n/4 = 4n - n/4$.



Both x and y have n/2 1's and differ in at least n/4 positions. So among the positions where x is 1, there are at least n/8 positions where y is 0 and vice-versa.

x+y :

1. at most n/2 –n/8 positions with 2 (both are 1).
2. at least n/4 positions with 1 .
3. $||x+y||^2 \leq 4(n/2-n/8) + n/4 = 2n - n/4$

Two cases

- One case: $\|x + x\|_2^2$:

$$\hat{F}_2 \geq \|x + y\|_2^2(1 - 0.01) \geq 4n(0.99)$$

- Other case: $\|x + y\|_2^2$:

$$\hat{F}_2 < \|x + x\|_2^2(1 + 0.1) \leq (4n - n/4)(1.01)$$

Lower bound: Inference

- So either $\|2x\|^2$ is not computed within 1 ± 0.01 or $\|x+y\|^2$ is not computed within 1 ± 0.01.
- Algorithm A makes a mistake for either x or y. So distinct elements of C must be mapped to distinct images.
- A requires $\log|\text{Codesize}| = \log 2^{cn} = \Omega(n)$ bits.

F_2: Randomized approximate problem definition

- Modified problem: Given ϵ and δ, design an algorithm that returns $\hat{F}_2$ satisfying

$$|\hat{F}_2 - F_2| \leq \epsilon F_2 \text{ with prob. } 1 - \delta \quad .$$

Outline

Recap: Independence of Random Variables

The random variables $X_1, X_2, \ldots, X_n$ are said to be independent if their joint probability distribution function equals the product of their individual probability distributions. That is, for any choice of values $a_1, a_2, \ldots, a_n$,

$$\Pr\{X_1 = a_1 \wedge X_2 = a_2 \wedge \ldots \wedge X_n = a_n\}$$
$$= \Pr\{X_1 = a_1\} \times \Pr\{X_2 = a_2\} \times \ldots \times \Pr\{X_n = a_n\} \quad .$$

k-wise Limited Independence

$\{X_1, X_2, \ldots, X_n\}$ a family of random variables are k-wise independent if for distinct indices $1 \leq i_1, i_2, \ldots, i_k \leq n$ and $a_1 \in \mathrm{support}(X_{i_1}), \ldots, a_k \in \mathrm{support}(X_{i_k})$.

$$\begin{aligned}\Pr\left\{X_{i_1} = a_1 \wedge X_{i_2} = a_2 \wedge \ldots \wedge X_{i_k} = a_k\right\} \\ = \Pr\left\{X_{i_1} = a_1\right\}\Pr\left\{X_{i_2} = a_2\right\} \ldots \Pr\left\{X_{i_k} = a_k\right\} \quad .\end{aligned}$$

k-wise independence: Implications

- Product of expectation of any k of X_i's is the product of individual expectations.
  $\mathbb{E}\left[X_{i_1} \ldots X_{i_k}\right] = \mathbb{E}\left[X_{i_1}\right] \mathbb{E}\left[X_{i_2}\right] \ldots \mathbb{E}\left[X_{i_k}\right]$.
- k-wise independence implies $k-1$-wise independence.
- Space and randomness efficient: suffices for most applications (we will see this now).

k-wise independent hash functions

[Wegman Carter 81]

- $\mathcal{H}$ is a finite family of functions mapping $[n] \to [b]$, usually $b \ll n$.
- Pick random member $h \in \mathcal{H}$ with prob. $1/|\mathcal{H}|$.
- $\mathcal{H}$ is k-wise independent if for any $x_1, \ldots, x_k$ distinct, and any $b_1, \ldots, b_k \in [m]$,

$$\Pr_{h \in \mathcal{H}} \{(h(x_1) = b_1) \wedge (h(x_2) = b_2) \ldots \wedge (h(x_k) = b_k)\}$$

$$=$$

$$\Pr_{h \in \mathcal{H}} \{h(x_1) = b_1\} \cdot \Pr_{h \in \mathcal{H}} \{h(x_2 = b_2)\} \cdots \times \Pr_{h \in \mathcal{H}} \{h(x_k) = b_k\} \quad .$$

Hash Family: Degree $k-1$ polynomials

- $\mathbb{F}$ is a finite field of size at least n.
- $\mathcal{H}_k$: all k-tuples $(a_0, \ldots, a_{k-1})$ over $\mathbb{F}$, viewed as a degree $k-1$ polynomial $h(x)$ over $\mathbb{F}$:

$$h(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_{k-1} x^{k-1} \quad .$$

- The family $\mathcal{H}_k$ is k-wise independent. Why?

Hash Family: Degree $k-1$ polynomials

- $\mathbb{F}$ is a finite field of size at least n.
- $\mathcal{H}_k$: all k-tuples $(a_0, \ldots, a_{k-1})$ over $\mathbb{F}$, viewed as a degree $k-1$ polynomial $h(x)$ over $\mathbb{F}$:

$$h(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_{k-1} x^{k-1} \quad .$$

- The family $\mathcal{H}_k$ is k-wise independent. Why?
  - $|H| = |F|^k$.
  - Count number of solutions to $h(x_i) = b_i$: $a_0 = b_i - a_1 x_i - a_2 x_i^2 - \ldots$
  - so # solutions is $|F|^{k-1}$. So, $\Pr\{h(x_i) = b_i\} = 1/|F|$.
  - Count number of solutions to $h(x_i) = b_i, i = 1, \ldots, k$. This is 1. Joint probability is $1/|F|^k$.

Hash Family: Degree $k-1$ polynomials

- $\mathbb{F}$ is a finite field of size at least n.
- $\mathcal{H}_k$: all k-tuples $(a_0, \ldots, a_{k-1})$ over $\mathbb{F}$, viewed as a degree $k-1$ polynomial $h(x)$ over $\mathbb{F}$:

$$h(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_{k-1} x^{k-1} \quad .$$

- The family $\mathcal{H}_k$ is k-wise independent. Why?
  - $|H| = |F|^k$.
  - Count number of solutions to $h(x_i) = b_i$: $a_0 = b_i - a_1 x_i - a_2 x_i^2 - \ldots$
  - so # solutions is $|F|^{k-1}$. So, $\Pr\{h(x_i) = b_i\} = 1/|F|$.
  - Count number of solutions to $h(x_i) = b_i, i = 1, \ldots, k$. This is 1. Joint probability is $1/|F|^k$.
- Space and randomness: store $a_0, \ldots, a_{k-1}$: $O(k \log n)$ bits.

Outline

Linear Rademacher Sketch

- For $i \in [n]$, $\xi_i \in \{-1, 1\}$ randomly with probability 1/2 each: *Rademacher* random variables.
- Let ξ_i's be 4-wise independent.
- Implementation: Choose h at random from the family of cubic polynomials over $\mathbb{F}_{2^r}$, where, $n \le 2^r < 2n$.

$$\xi(u) = \begin{cases} 1 & \text{if last bit of } h(u) = 1 \\ -1 & \text{otherwise.} \end{cases}$$

- A sketch is a random counter:

$$X = \sum_{i=1}^{n} f_i \xi(i) \quad .$$

- Easily updated corresponding to stream updates (i, v):

$$X := X + v \cdot \xi(i) \quad .$$

Sketches

$X = \sum_i f_i \xi(i)$. Recall: $\xi_i \in \{-1, 1\}$ with prob. $1/2$ each.

$$\mathbb{E}\left[X^2\right] = \mathbb{E}\left[\left(\sum_{i=1}^{n} f_i \xi(i)\right)^2\right] = \mathbb{E}\left[\sum_{i=1}^{n} f_i^2 + 2 \sum_{1 \leq i < j \leq n} f_i f_j \xi(i) \xi(j)\right]$$
$$= \sum_{i=1}^{n} f_i^2 = F_2$$

using linearity of expectation, pair-wise independence and symmetry around 0 of $\xi(i)$'s.

Sketch:Variance

$$
\begin{aligned}
\mathbb{E}\left[X^4\right] = \mathbb{E}\left[\left(\sum_{i=1}^{n} f_i \xi(i)\right)^4\right] &= \sum_{i=1}^{n} f_i^4 + \sum_{i \neq j} 4 f_i^3 f_j \mathbb{E}\left[(\xi(i))^3 \xi(j)\right] \\
&+ \sum_{i,j \text{ distinct}} 6 f_i^2 f_j^2 \mathbb{E}\left[\xi(i)^2 \xi(j)^2\right] + \sum_{i,j,k \text{ distinct}} 12 f_i^2 f_j f_k \mathbb{E}\left[\xi(i)^2 \xi(j) \xi(k)\right] \\
&+ \sum_{i,j,k,l \text{ distinct}} 4! f_i f_j f_k f_l \mathbb{E}\left[\xi(i)\xi(j)\xi(k)\xi(l)\right]
\end{aligned}
$$

Expectation of up to four-wise products of $\xi(j)$'s is the product of the corresponding expectations. So,

$$\mathbb{E}\left[X^4\right] = \sum_{i=1}^{n} f_i^4 + \sum_{i<j} 6 f_i^2 f_j^2 \leq 3\Big(\sum_{i=1}^{n} f_i^2\Big)^2 = 3F_2^2 \ .$$

$$\mathrm{Var}[X^2] = \mathbb{E}\left[X^4\right] - (\mathbb{E}[X^2])^2 = 3F_2^2 - F_2^2 = 2F_2^2 \ .$$

Designing estimator for F_2 contd.

- Keep $t = 16/\epsilon^2$ independent sketches $X_1, X_2, \ldots X_t$.
- Return averages of squares: $Y = \left(X_1^2 + \ldots + X_t^2\right)/t$.
- So, $\mathbb{E}[Y] = \mathbb{E}\left[X_1^2\right] = F_2$.
- X_i^2's are independent, so, variance of their sum is the sum of their variances. So,

$$\mathrm{Var}\left[Y\right] = \frac{1}{t^2} \cdot t\mathrm{Var}\left[X_1^2\right] = 2F_2^2/t = \epsilon^2 F_2^2/8 \ .$$

Estimator for F_2

- Chebychev's inequality for any real valued variable Y,

$$\Pr\{|Y - \mathbb{E}[Y]| > \alpha\} < \mathrm{Var}\left[Y\right]/\alpha^2 \ .$$

- So, $\Pr\{|Y - F_2| > \epsilon F_2\} < \mathrm{Var}\left[Y\right]/(\epsilon^2 F_2^2) = 1/8$, or, $|Y - F_2| < \epsilon F_2$ with probability $7/8$.

Boosting confidence using median

- Let A be a randomized algorithm.
- On input I, correct value is $Y(I)$.
- Suppose A on input I returns (random) numeric value $\hat{Y}(I)$. and the following guarantee:

$$\Pr\left\{|\hat{Y}(I) - Y(I)| < \epsilon Y(I)\right\} \geq \frac{7}{8}$$

- To boost confidence to $1 - \delta$, run A independently on I $s = O(\log \frac{1}{\delta})$ times to obtain

$$\hat{Y}_1(I), \ldots, \hat{Y}_s(I) \quad .$$

- Now return

$$M = \text{med}\{\hat{Y}_1(I), \ldots, \hat{Y}_s(I)\}$$

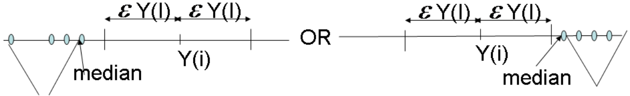
Boosting using Median-II

- Upper bound the probability that median M is "bad", that is, $|M - Y| > \epsilon Y$.
- Define indicator variable $X_i = 0$ if the jth run of A gives a "good answer" and is 1 otherwise.

$$X_j = \begin{cases} 0 & \text{if} \quad |\hat{Y}_j(I) - Y(I)| < \epsilon Y(I) \\ 1 & \text{otherwise.} \end{cases}$$

$$\Pr\{X_j = 1\} \leq \frac{1}{8}$$

- Let $X = X_1 + X_2 + \ldots + X_s$: count number of "bad" answers.
- $\mathbb{E}[X] \leq s/8$.
- M is "bad" implies there are at least 1/2 of the X_j's that are "bad", i.e., $X \geq \frac{s}{2}$.



Boosting with median: Analysis

Chernoff's bound

Let $X_1, \ldots, X_t$ be independent random variables taking values from $\{0, 1\}$ with $\mathbb{E}[X_i] = p_i$. Let $X = X_1 + X_2 + \ldots + X_t$ and $\mu = p_1 + \ldots + p_t$. Then, for $0 < \epsilon < 1$,

$$\Pr\{X > (1+\epsilon)\mu\} < e^{-\mu\epsilon^2/3}$$
$$\Pr\{X < (1-\epsilon)\mu\} < e^{-\mu\epsilon^2/2} \quad .$$

- By Chernoff's bound, with high probability, X should concentrate close to $\mathbb{E}[X] = s/8$.

$$\Pr\{X \geq s/2\} \leq \Pr\{X \geq s/4\} \leq e^{-s/24} \quad .$$

This is at most δ if $s = O(\log \frac{1}{\delta})$.

F_2 estimation with high confidence

- Maintain $s = O(log(1/\delta))$ groups of $t = 16/\epsilon^2$ independent sketches X_j^r, $j = 1, 2, \ldots, t$, $r = 1, 2, \ldots, s$.
- In each group r, take average

$$Y_r = \mathrm{avg}_{j=1}^{t}(X_j^r)^2, \quad r = 1, 2, \ldots s \ .$$

- Return median of the averages

$$\hat{F}_2 = \mathrm{med}_{r=1}^{s} Y_j \ .$$

- Property:

$$\Pr\left\{|\hat{F}_2 - F_2| < \epsilon F_2\right\} \geq 1 - \delta \ .$$

AMS: Resources consumed

Space:

- Let $|f_i| \leq m$. Each sketch $\sum_i f_i \xi(i)$ can be stored in $\log(mn)$ bits.
- Space = $O(\frac{1}{\epsilon^2} \log(1/\delta)) \times \log(mn)$ bits.

Time to process stream update (i, v):

- Each sketch is updated.
- Requires evaluating degree 3 polynomial over $\mathbb{F}$: $O(1)$ simple field operations, total $O(\log(1/\delta)/\epsilon^2)$.

Randomness:

- Each sketch requires $4 \log n$ random bits, total $O(\log(n)/\epsilon^2)$.

Outline

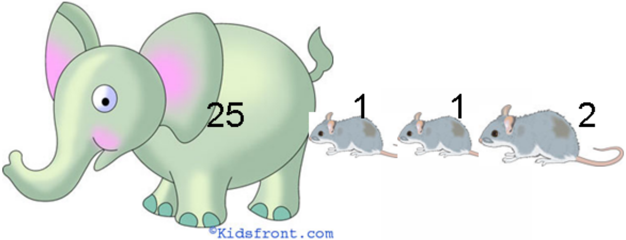
Heavy Hitters: Illustration

Heavy Hitters are items with large absolute frequencies (Elephants)

stream: $(1, 10)(2, 1)(3, 1)(4, 2)(1, 10)\ldots$

frequency vector

25	1	1	2	0	∘ ∘ ∘
1	2	3	4	5	



Elephants and mice

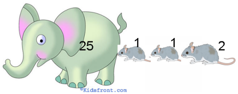
Heavy Hitter Problem: Find the elephants

Applications

- Among the most popular applications of data streaming.
    1. Find the IP-addresses that send the most traffic.
    2. Find source-IP, dest-IP pairs that send the most traffic to each other.
    3. Find the most visited web sites.

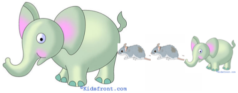
    $\vdots$

Heavy Hitters: Definition



- ℓ_p heavy hitters with threshold parameter $\phi \in (0,1)$:
  $HH_p^{\phi}(f) = \left\{ i \in [n] : |f_i|^p > \phi \sum_{j=1}^{n} |f_j|^p \right\}$.
- Given ϕ, can we find the set HH_p^{ϕ} in low space (close to $O(\frac{1}{\phi})$).
- Finding HH_p^{ϕ} EXACTLY requires $\Omega(n)$ space [KSP02].
  Consider $HH_1^{1/2}$: i s.t. $|f_i| > F_1/2$, Majority problem.
  Consider $2n$-dimensional binary vectors f with $wt = n$. Add n to coordinate i and test for majority. Now, i is majority iff f_i was 1 earlier. Vector is recovered. Requires $\log \binom{2n}{n} = \Omega(n)$ bits.

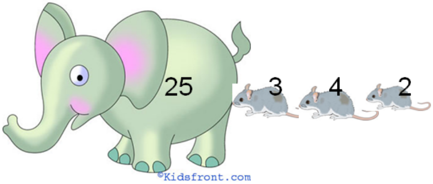
Approximate Heavy Hitters: Definition



- Approximate heavy hitters: $\text{ApproxHH}_p^{\phi,\phi'}$. ϕ is upper threshold, $\phi' < \phi < 1$ is lower threshold.
- Return (any) set S such that
  1. $S \supset HH_p^{\phi}$: Do not miss i with $|f_i|^p > \phi F_p$.
  2. $S \subset HH_p^{\phi'}$: Do not include i with $|f_i|^p < \phi' F_p$.
- Uncertainty allows low space algorithms. Space approx. $\tilde{O}(1/(\phi - \phi'))$.

ℓ_p Point Query/Estimating Frequencies

- Point query: Estimate frequency of any item i. Cannot be done exactly in $o(n)$ space. Allow bounded error, for any query point i, $\hat{f}_i^p = f_i^p \pm \phi F_p$, .



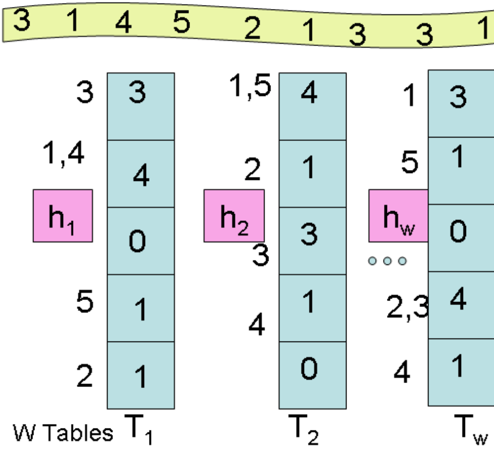
Frequency Vector

Estimated frequencies

Outline

Count-Min Sketch: Basic algorithm

- w hash tables $T_1, T_2, \ldots, T_w$.
- Each table T_j:
  1. B buckets.
  2. hash fn. $h_j : [n] \rightarrow [B]$.
  3. $h_j \in_R$ pair-wise indep. family.
- h_j's independent.
- UPDATE(i, v): $T_j[h_j(i)]$ **+=** v, $j = 1, 2, \ldots, w$.
- ESTIMATE(i):



All non-negative frequencies
$\hat{f}_i = \min_{j=1}^{w} T_j[h_j(i)]$

General frequencies
$\hat{f}_i = \text{median}_{j=1}^{w} T_j[h_j(i)]$

Analysis

- $$\mathbb{E}\Big[\big|T_l[h_l(i)] - f_i\big|\Big] = \mathbb{E}\Big[\big|\sum_{\substack{i \neq k \\ h_l(i)=h_l(k)}} f_k\big|\Big] \leq \sum_{i \neq k} \frac{|f_k|}{B} = \frac{F_1 - |f_i|}{B}$$

  by pair-wise independence of hash family of h_l.

Analysis

- $$\mathbb{E}\left[\left|T_l[h_l(i)] - f_i\right|\right] = \mathbb{E}\left[\left|\sum_{\substack{i \neq k \\ h_l(i)=h_l(k)}} f_k\right|\right] \leq \sum_{i \neq k} \frac{|f_k|}{B} = \frac{F_1 - |f_i|}{B}$$

  by pair-wise independence of hash family of h_l.

- **Markov's inequality**:

  $$\Pr\{X \geq a\} \leq \mathbb{E}[X]/a, \quad X \text{ non-negative random variable.} \quad .$$

  Using Markov's inequality,

  $$\Pr\left\{\left|T_l[h_l(i)] - f_i\right| > 4F_1/B\right\} \leq 1/4 \ .$$

Analysis

- $$\mathbb{E}\left[\left|T_l[h_l(i)] - f_i\right|\right] = \mathbb{E}\left[\left|\sum_{\substack{i \neq k \\ h_l(i)=h_l(k)}} f_k\right|\right] \leq \sum_{i \neq k} \frac{|f_k|}{B} = \frac{F_1 - |f_i|}{B}$$

  by pair-wise independence of hash family of h_l.
- **Markov's inequality**:

  $$\Pr\{X \geq a\} \leq \mathbb{E}[X]/a, \quad X \text{ non-negative random variable.}$$

  Using Markov's inequality,

  $$\Pr\left\{\left|T_l[h_l(i)] - f_i\right| > 4F_1/B\right\} \leq 1/4 .$$
- Taking median from estimates of $w = O(\log(1/\delta))$ tables

  $$\Pr\left\{\left|\text{median}_{l=1}^{w} T_l[h_l(i)] - f_i\right| > 4F_1/B\right\} < \delta .$$

Analysis

- $$\mathbb{E}\left[\left|T_l[h_l(i)] - f_i\right|\right] = \mathbb{E}\left[\left|\sum_{\substack{i \neq k \\ h_l(i)=h_l(k)}} f_k\right|\right] \le \sum_{i \neq k} \frac{|f_k|}{B} = \frac{F_1 - |f_i|}{B}$$

  by pair-wise independence of hash family of h_l.
- **Markov's inequality**:

  $$\Pr\{X \ge a\} \le \mathbb{E}[X]/a, \quad X \text{ non-negative random variable.}$$

  Using Markov's inequality,

  $$\Pr\left\{\left|T_l[h_l(i)] - f_i\right| > 4F_1/B\right\} \le 1/4 \; .$$

- Taking median from estimates of $w = O(\log(1/\delta))$ tables

  $$\Pr\left\{\left|\text{median}_{l=1}^{w} T_l[h_l(i)] - f_i\right| > 4F_1/B\right\} < \delta \; .$$

- Space: $O(B \log(1/\delta))$ counters. Update time: $O(\log(1/\delta))$. Randomness: $2 \log n \times O(\log(1/\delta))$ bits .

Count-Min-Sketch: ℓ_1 Approx. Heavy-Hitters

- Solve $HH_1^{\phi,\phi'}$. Need all i: $|f_i| > \phi F_1$, no i: $|f_i| < \phi' F_1$.

Count-Min-Sketch: ℓ_1 Approx. Heavy-Hitters

- Solve $HH_1^{\phi,\phi'}$. Need all i: $|f_i| > \phi F_1$, no i: $|f_i| < \phi' F_1$.
- Assume F_1 is known (otherwise, use $\hat{F}_1$, adjust constants.)

COUNT-MIN-Sketch: ℓ_1 Approx. Heavy-Hitters

- Solve $HH_1^{\phi,\phi'}$. Need all i: $|f_i| > \phi F_1$, no i: $|f_i| < \phi' F_1$.
- Assume F_1 is known (otherwise, use $\hat{F}_1$, adjust constants.)
- Keep $B = \lceil 8/(\phi - \phi') \rceil$ buckets per table, and $w = O(\log(n/\delta))$ buckets.

Count-Min-Sketch: ℓ_1 Approx. Heavy-Hitters

- Solve $HH_1^{\phi,\phi'}$. Need all i: $|f_i| > \phi F_1$, no i: $|f_i| < \phi' F_1$.
- Assume F_1 is known (otherwise, use $\hat{F}_1$, adjust constants.)
- Keep $B = \lceil 8/(\phi - \phi') \rceil$ buckets per table, and $w = O(\log(n/\delta))$ buckets.
- Iterate over domain $[n]$ and obtain $\hat{f}_i$ for each i.

Count-Min-Sketch: ℓ_1 Approx. Heavy-Hitters

- Solve $HH_1^{\phi,\phi'}$. Need all i: $|f_i| > \phi F_1$, no i: $|f_i| < \phi' F_1$.
- Assume F_1 is known (otherwise, use $\hat{F}_1$, adjust constants.)
- Keep $B = \lceil 8/(\phi - \phi') \rceil$ buckets per table, and $w = O(\log(n/\delta))$ buckets.
- Iterate over domain $[n]$ and obtain $\hat{f}_i$ for each i.
- Return i with $\hat{f}_i \geq ((\phi + \phi')/2)\hat{F}_1$.

COUNT-MIN-Sketch: ℓ_1 Approx. Heavy-Hitters

- Solve $HH_1^{\phi,\phi'}$. Need all i: $|f_i| > \phi F_1$, no i: $|f_i| < \phi' F_1$.
- Assume F_1 is known (otherwise, use $\hat{F}_1$, adjust constants.)
- Keep $B = \lceil 8/(\phi - \phi') \rceil$ buckets per table, and $w = O(\log(n/\delta))$ buckets.
- Iterate over domain $[n]$ and obtain $\hat{f}_i$ for each i.
- Return i with $\hat{f}_i \geq ((\phi + \phi')/2)\hat{F}_1$.
- Error in estimation $\Delta = ((\phi - \phi')/2)F_1$.
  - if $|f_i| > \phi F_1$, then, $|\hat{f}_i| > |f_i| - \Delta > ((\phi + \phi')/2)F_1$.
  - if $|f_i| < \phi' F_1$, then, $|\hat{f}_i| < \phi' F_1 + \Delta < (\phi + \phi')/2)F_1$.

COUNT-MIN-Sketch: ℓ_1 Approx. Heavy-Hitters

- Solve $HH_1^{\phi,\phi'}$. Need all i: $|f_i| > \phi F_1$, no i: $|f_i| < \phi' F_1$.
- Assume F_1 is known (otherwise, use $\hat{F}_1$, adjust constants.)
- Keep $B = \lceil 8/(\phi - \phi') \rceil$ buckets per table, and $w = O(\log(n/\delta))$ buckets.
- Iterate over domain $[n]$ and obtain $\hat{f}_i$ for each i.
- Return i with $\hat{f}_i \geq ((\phi + \phi')/2)\hat{F}_1$.
- Error in estimation $\Delta = ((\phi - \phi')/2)F_1$.
  - if $|f_i| > \phi F_1$, then, $|\hat{f}_i| > |f_i| - \Delta > ((\phi + \phi')/2)F_1$.
  - if $|f_i| < \phi' F_1$, then, $|\hat{f}_i| < \phi' F_1 + \Delta < (\phi + \phi')/2)F_1$.
- But, domain is large and iteration becomes expensive.

Group Testing Overview: Bit tester

- General idea: Each heavy-hitter is a majority item in its bucket with probability $3/4$.
- Problem: find majority item in a bucket if there is one. Following works for non-negative frequencies. If no majority item, gives a false positive.

Assume: Non-negative frequencies

Ex. 1

	Bit1	Bit2		Bit4		Bit6
0	7	8	4	10	1	4
1	3	2	6	0	9	6
Majority Item: found	0	0	1	0	1	1

L1 majority structure

1. Finds true majority item if one exists.
2. May report a false positive.

Ex. 2

	Bit1	Bit2		Bit4		Bit6
0	7	8	4	5	1	4
1	3	2	6	5	9	6
Majority Item: Amibguous	0	0	1	?	1	1

Group testing for ℓ_2 majority

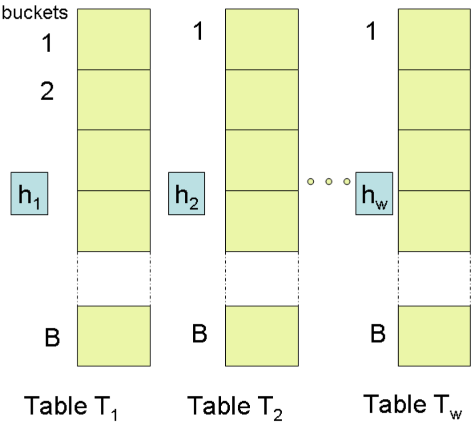
- ℓ_2 majority: $|f_i|^2 > F_2/2$, strengthen to $|f_i|^2 > F_2/4$. Use twice the size of hash table.
- Keep $O(1)$ AMS/Gaussian sketches for each bit position. Allows estimation of sub-stream mapping to a bucket/bit position/bit-value to accuracy of $1 \pm 1/8$ (say) with constant probability $7/8$ say.
- For majority item, each bit position is correctly found with constant probability say 3/4.
- So, with very high probability, $2/3$rd bits are correctly recovered (Chernoff's bound).
- Instead of using bit positions, use an error-correcting code for i: $C(i)$ that can correct $1/3$ fraction of bits [GLPS10].

ℓ_2 point query & HH: CountSketch[CCF-C02]

- CountSketch structure:
  1. w tables $T_1, \ldots, T_w$.
  2. $h_j : [n] \to [B]$ corresponding to T_j.
  3. h_j randomly chosen from a pair-wise indep. family.
  4. $h_1, \ldots, h_w$ are independently chosen.
  5. Sketch fn. $\xi_j : [n] \to \{-1, +1\}$ corresponding to T_j, 4-wise independent.
  6. $$T_j[b] = \sum_{i : h_j(i) = b} f_i \xi_j(i)$$

$b = 1, \ldots, B$, $j = 1, 2, \ldots, w$.

Each bucket keeps AMS sketch of sub-stream mapping to it



Countsketch Structure

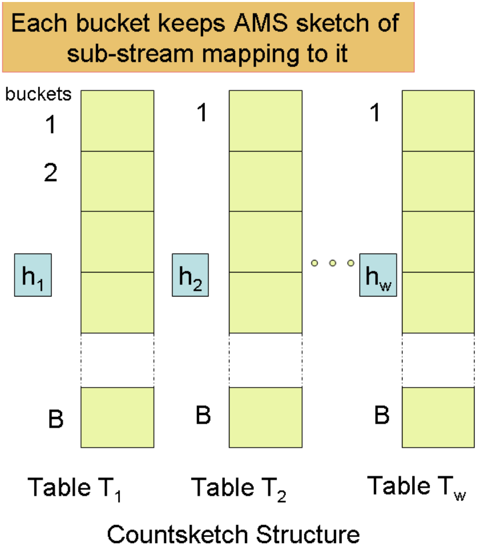
CountSketch structure

- Update(i, v):

  for $j = 1$ **to** w {
  $T_j[h_j(i)]$ **+=** $v \cdot \xi_j(i)$
  }

- Estimate(i):

  $$\hat{f}_i = \text{med}_{j=1}^{w} T_j[h_j(i)] \cdot \xi_j(i) \quad .$$



Frequency recovery: Basic idea

- Median of estimates from each table: table l estimate $T_l[h_l(i)] \cdot \xi_l(i)$.
- $T_l[h_l(i)] \cdot \xi_l(i) = f_i + \sum_{k \neq i, h_l(i) = h_l(k)} f_k \xi_l(k) \xi_l(i)$.
- $\mathbb{E}\left[\text{Estimate from table } l\right] = f_i$.
- Variance is $O(F_2 - |f_i|^2/B)$. Better analysis: $F_2^{\text{res}}(B/8)/B$, conditional on i does not collide with any of the top-$B/8$ items. This holds with probability $7/8$. $F_2^{\text{res}}(k) = F_2$ of vector except for the top-k frequencies by absolute value.
- This gives,

$$|\hat{f}_i - f_i| \leq O\Big(\Big(\frac{F_2^{\text{res}}(B/8)}{B}\Big)^{1/2}\Big)$$

Which is better?

- We have shown estimators for PtQuery_1^ϕ and PtQuery_2^ϕ as follows.

$$\text{PtQuery}_1 : \quad |\hat{f}_i - f_i| \le \frac{F_1^{\text{res}}(k)}{k}, \text{ space } O(k \log(1/\delta))$$

$$\text{PtQuery}_2 : \quad |\hat{f}_i - f_i| \le \left(\frac{F_2^{\text{res}}(k)}{k}\right)^{1/2}, \text{ space } O(k \log(1/\delta)) \ .$$

- Both are close to being space-optimal.
- Which is more accurate (more than just constant factors)?
- We have,

$$\left(\frac{F_2^{\text{res}}(2k)}{2k}\right)^{1/2} \le \frac{F_1^{\text{res}}(k)}{2k}$$

So PtQuery_2 method implies $|\hat{f}_i - f_i| < O(F_1^{\text{res}}(k)/k)$.

Outline

A Dimensionality Reduction View

- Keep $s = O(\log m)$ tables for Fast-AMS or $O(\log m)$ groups, of $16/\epsilon^2$ sketches in each group.
- A sketch can be viewed as a map from frequency vectors to some sketch space: $sk : \mathbb{R}^n \mapsto \mathbb{R}^{O(\epsilon^{-2}\log(m))}$.
- m streams with frequency vectors $f^1, \ldots, f^m$.
- Sketch is linear: therefore,

$$sk(f^i - f^j) = sk(f^i) - sk(f^j) \ .$$

- So with probability $7/8$, we have

$$\|f^i - f^j\|_2 \in (1 \pm \epsilon)\text{Med}(sk(f^i) - sk(f^j)), \forall i, j.$$
$$\|f^i\| \in (1 \pm \epsilon)\text{Med}(sk(f^i)), \forall i$$

- But Med is not an ℓ_2 norm in the sketch space.

Dimensionality Reduction: Metric Space view

- A discrete metric space (X, d_X): X is a finite set of points, $d_X(x, y)$ gives distance between points x and y in X. d_X function satisfies metric properties.
- (X, d_X) embeds into (Y, d_Y) with distortion D if there exists $f : X \to Y$ and a scaling constant c such that

$$c \cdot d_X(x, y) \leq d_Y(f(x), f(y)) \leq c \cdot D \cdot d_X(x, y), \quad \forall x, y \in X \quad .$$

Johnson-Lindenstrauss (J-L) Lemma

- For any $0 < \epsilon < 1$ and a set S of m points from $\mathbb{R}^n$, there exists a mapping $f : \mathbb{R}^n \to \mathbb{R}^t$ where, $t = O(\epsilon^{-2} \log m)$ s.t.

$$(1 - \epsilon)\|x - y\|_2 \leq \|f(x) - f(y)\|_2 \leq \|x - y\|_2, \forall x, y \in S \quad .$$

- Follows from:
  There exists a probabilistic mapping $\mu : \mathbb{R}^n \to \mathbb{R}^t$, for $t = O(\epsilon^{-2} \log(1/\delta))$ with μ distributed as $\mathcal{D}$, such that for any unit vector $\|x\|_2 = 1$,

$$\Pr_{\mu \sim D} \left\{ |\|\mu(x)\|_2^2 - 1| \leq \epsilon \right\} \geq 1 - \delta$$

Usefulness of Embeddability

- ϵ-distortion implies: nearest neighbors are approximately preserved.
- *k*-d trees and other ℓ_2-based geometric data structures can be used in much fewer dimensions.
- Time complexity of most geometric algorithms, including NN, is exponential in dimension.
- A basic step in reducing this “curse of dimensionality”.
- We now see the basic set up of J-L Lemma.

Normal Distribution

- Gaussian distribution (Normal distribution):
  $X \sim N(\mu, \sigma^2)$. $\mathbb{E}[X] = \mu$, $\text{Var}\left[X\right] = \sigma^2$.
- Density function: $f_X(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$.
- Standard Normal distribution: $N(0, 1)$.
- Stability: Sum of independent normally distributed variates is normally distributed.
  $X_i \sim N(\mu_i, \sigma_i^2)$, $i = 1, 2, \ldots, k$, X_i's independent. Then,

$$X_1 + \ldots + X_k \sim N(\mu_1 + \ldots + \mu_k, \sigma_1^2 + \ldots + \sigma_k^2) \ .$$

Gamma distribution

- Gamma(k, θ), $k =$ shape parameter, $\theta =$ scale factor (non-negative). Density function:

$$f(x; k, \theta) = \frac{1}{\theta^k \Gamma(k)} x^{k-1} e^{-x/\theta} \quad .$$

- $\mathbb{E}[X] = k\theta$.
- If $X \sim N(0, \sigma^2)$, then, $X^2 \sim$ Gamma$(1/2, 2\sigma^2)$.
- Scaling: $X \sim$ Gamma(k, θ), then, $aX \sim$ Gamma$(k, a\theta)$.
- Sum of indepdendent Gamma variates is Gamma distributed **if** scale factors are same. Let $X_i \sim$ Gamma(k_i, θ) and independent. Then,

$$X_1 + \ldots + X_r \sim \text{Gamma}(k_1 + k_2 + \ldots + k_r, \theta) \quad .$$

Application to estimating F_2: Gaussian sketches

- Let $\xi(j) \sim N(0,1)$ for $j \in [n]$.
- $\xi(j)$'s are (fully) independent. Ignore randomness/space/time required for now.
- Consider sketch

$$X = \sum_{i=1}^{n} f_i \xi(i) \quad .$$

- By stability property of normal distr.

$$X \sim N(0, F_2) \quad .$$

- Problem reduces to: Estimate variance of X.

Gaussian sketches

- Let $X_1, X_2, \ldots, X_t$ be independent Gaussian sketches.
- $Y = X_1^2 + \ldots + X_t^2$.

Gaussian sketches

- Let $X_1, X_2, \ldots, X_t$ be independent Gaussian sketches.
- $Y = X_1^2 + \ldots + X_t^2$.
- Each $X_j^2 \sim$ Gamma $(1/2, 2F_2)$. So $Y \sim$ Gamma$(t/2, 2F_2)$.

Gaussian sketches

- Let $X_1, X_2, \ldots, X_t$ be independent Gaussian sketches.
- $Y = X_1^2 + \ldots + X_t^2$.
- Each $X_j^2 \sim$ Gamma $(1/2, 2F_2)$. So $Y \sim$ Gamma$(t/2, 2F_2)$.
- $\mathbb{E}[Y] = tF_2$. Need Tail probability: $\Pr\{Y > (1 \pm \epsilon)tF_2\}$.

Gaussian sketches

- $Y = X_1^2 + \ldots + X_t^2$.
- Each $X_j^2 \sim$ Gamma $(1/2, 2F_2)$. So $Y \sim$ Gamma$(t/2, 2F_2)$.
- $\mathbb{E}[Y] = tF_2$. Need Tail probability: $\Pr\{Y > (1 \pm \epsilon)tF_2\}$.
- Property: If $Y \sim$ *Gamma*(t, θ). Then, for $0 < \epsilon < 1$,

$$\Pr\{Y \in (1 \pm \epsilon)\mathbb{E}[Y]\} \leq \frac{2e^{-\epsilon^2 t/6}}{\epsilon\sqrt{2\pi(t-1)}}.$$

Gaussian sketches

- $Y = X_1^2 + \ldots + X_t^2$.
- Each $X_j^2 \sim$ Gamma $(1/2, 2F_2)$. So $Y \sim$ Gamma$(t/2, 2F_2)$.
- $\mathbb{E}[Y] = tF_2$. Need Tail probability: $\Pr\{Y > (1 \pm \epsilon)tF_2\}$.
- Property: If $Y \sim Gamma(t, \theta)$. Then, for $0 < \epsilon < 1$,

$$\Pr\{Y \in (1 \pm \epsilon)\mathbb{E}[Y]\} \leq \frac{2e^{-\epsilon^2 t/6}}{\epsilon\sqrt{2\pi(t-1)}}.$$

- Let $t = O(\epsilon^{-2}\log(m))$. Then,

$$\frac{Y}{t} \in (1 \pm \epsilon)F_2, \text{ with prob. } 1 - \frac{1}{8m^2} \quad .$$

Another view of mapping: J-L Lemma

- $t \times n$ matrix A, entries $z_{i,j}$ drawn from $N(0,1)$ i.i.d.

$$A = \begin{bmatrix} z_{1,1} & z_{1,2} & \cdots & z_{1,n} \\ z_{2,1} & z_{2,1} & \cdots & z_{2,n} \\ & \vdots & \vdots & \\ z_{t,1} & z_{t,2} & \cdots & z_{t,n} \end{bmatrix}$$

- $x \in \mathbb{R}^n$, $x \mapsto Ax$, $\|Ax\|_2 \in (1 \pm \epsilon)\|x\|_2$ with prob. $1 - 1/m^{O(1)}$.
- By linearity, $A(x-y) = Ax - Ay$.
- Let $t = O(\epsilon^{-2} \log m)$. For any set S of m points,

$$\|Ax - Ay\|_2 \in (1 \pm \epsilon)\|x - y\|_2, \quad \forall x, y \in S$$

with probability $1 - 1/m^2$.

J-L matrices

- Let $\mathcal{D}$ be a distribution over matrices in such that

$$\Pr_{A \sim \mathcal{D}} \left\{ \|Ax\|_2^2 \in (1 \pm \epsilon)\|x\|_2^2 \right\} \geq 1 - 1/n^2 \ .$$

- Examples:
  1. Matrices with Rademacher (random ± 1) entries and (slightly sparse) Rademacher [Achlioptas 01]. Matrices with entries from distributions with sub-Gaussian[1]

[1] Sub-gaussian with expectation μ and variance σ^2:
$\Pr\{X > \lambda\} \leq e^{-\Omega(\lambda^2/\sigma^2)}$.

Fast J-L

- Computing Ax requires $O(tn)$ time. Can this be done faster? [Ailon-Chazelle]
- Write

$$A = PH_nD$$

  - D is $n \times n$ diagonal matrix with random ± 1 entries. Assume $n = 2^r$.
  - H_n is the $n \times n$ Hadamard matrix: Orthonormal and

$$H_n = \frac{1}{\sqrt{2}} \begin{bmatrix} H_{n/2} & H_{n/2} \\ H_{n/2} & -H_{n/2} \end{bmatrix}$$

    Due to recursive nature, H_nx can be computed in time $O(n \log n)$.

Outline

Small moments: problem and results

- $F_p = \sum_{i=1}^{n} |f_i|^p$. Restrict attention to $p \in (0, 2)$.
- F_0: number of items with non-zero frequency. "Count-distinct" queries.
- Deterministic PTAS requires $\Omega(n)$ space [AMS96].
- Lower Bounds: $(\epsilon^{-2} \log(\epsilon M))$ [IW03,Wood04,KNW10]

Estimate F_p, $0 < p < 2$, p-stable sketches [Indyk00]

- Unit scale p-stable distributions $\mathrm{St}(p, 1)$.
- Property of p-stability: if s_i's are unit p-stable and independent, then, (i) as_i has distribution $\mathrm{St}(p, |a|)$ for scalar a, and, (ii)

$$X = f_1 s_1 + f_2 s_2 + \ldots + f_n s_n \text{ is distributed as}$$

$$\mathrm{St}\Big(p, \left(|f_1|^p + \ldots |f_p|^p\right)^{1/p}\Big) \ .$$

- X is a p-stable random variable with scale factor $F_p^{1/p}$.
- If $Z \sim \mathrm{St}(p, 1)$ so X has same distribution as $F_p^{1/p} Z$, or $|X|^p$ has the same distribution as $F_p |Z|^p$.
- So, $\mathrm{med}(|X|^p) = F_p \mathrm{med}(|Z|^p)$.

Small F_p: Median

- Median method [Indyk00]: Make $t = O(1/\epsilon^2)$ independent observations $X_1, \ldots, X_t$.

$$\hat{F}_p = \frac{\text{med}(|X_1|^p, \ldots, |X_t|^p)}{\text{med}(|Z|^p)} \quad .$$

X is a scaling of Z, $|X|^p \;\sim\; F_p |Z|^p$.

Small F_p: Median

- Median method [Indyk00]: Make $t = O(1/\epsilon^2)$ independent observations $X_1, \ldots, X_t$.

  $$\hat{F}_p = \frac{\text{med}(|X_1|^p, \ldots, |X_t|^p)}{\text{med}(|Z|^p)} \quad .$$

  X is a scaling of Z, $|X|^p \sim F_p |Z|^p$.
- Lifschitz property of density function: there is *at least* $c \cdot \epsilon$ probability mass in each of the ranges:
  1. ϵ times the median and right of median: $|Z| \in [\text{med}(|Z|), (1+\epsilon)\text{med}(|Z|)]$.
  2. ϵ times median and left of median: $|Z| \in [(1-\epsilon)\text{med}(|Z|), \text{med}(|Z|)]$.

Small F_p: Median

- Median method [Indyk00]: Make $t = O(1/\epsilon^2)$ independent observations $X_1, \ldots, X_t$.

$$\hat{F}_p = \frac{\text{med}(|X_1|^p, \ldots, |X_t|^p)}{\text{med}(|Z|^p)} \quad .$$

  X is a scaling of Z, $|X|^p \; \sim \; F_p |Z|^p$.

- Lifschitz property of density function: there is *at least c* $\cdot \, \epsilon$ probability mass in each of the ranges:
  1. ϵ times the median and right of median: $|Z| \in [\text{med}(|Z|), (1+\epsilon)\text{med}(|Z|)]$.
  2. ϵ times median and left of median: $|Z| \in [(1-\epsilon)\text{med}(|Z|), \text{med}(|Z|)]$.
- Out of $t = d/\epsilon^2$ independent trials, by Chernoff bounds, the number of X_j's such that $|X|_j^p > (1+\epsilon)\text{med}(|X|^p)$ is $\exp\{-t(1/2 - c\epsilon)(c\epsilon)^2\} \geq 15/16$, by choosing $t = d/\epsilon^2$ appropriately.
- Similarly, for the left part.

Geometric Means Estimator [Li 2006]

- **Fact:** $X \sim \mathrm{St}(p, F_p^{1/p})$ then

$$\mathbb{E}\left[|X|^q\right] = C_{p,q} \cdot F_p^{q/p}, \qquad -1 < q < p \ .$$

- GM Estimator:

$$\hat{F}_p^{\mathrm{GM}}(r) = C'_{p,r} |X_1|^{p/r} |X_2|^{p/r} \ldots |X_r|^{p/r}, \quad r \geq 3$$

- Concentration: $\left|\hat{F}_p^{\mathrm{GM}}(r) - F_p\right| \leq \epsilon F_p$ with prob. 7/8 for $r = O(\frac{1}{\epsilon^2})$.

Reducing Randomness

- The stable variables are assumed independent.
- Space requirement is $O(\log(nm)) \times O\left(\frac{1}{\epsilon^2}\right)$. Time $R = O(nM)$.
- Use Nisan's PRG for fooling bounded space S computations [Indyk00] requiring time R. $O(S \log R)$ random bits suffices.

KNW 2010: Log Cosine Estimator

- $X \sim S(p, F_p^{1/p})$.
- $\mathbb{E}\left[e^{itX}\right] = e^{-|t|^p F_p} = \mathbb{E}\left[cos(tX)\right]$, [Levy1930s].
- Estimator:

$$C_s(t) = \frac{1}{s}(\cos(tX_1)) + \ldots + \cos(tX_r))$$
$$\hat{F}_p = \frac{1}{|t|^p} \log \frac{1}{C_s(t)}$$

- Choose t so that $(1 + O(\epsilon))e^{-1} \leq C_s(t) \leq (1 - O(\epsilon))e^{-1/8}$.
- $\hat{F}_p$ concentrates within $(1 \pm \epsilon)F_p$ with high probability.
- $O(\log(1/\epsilon))$-wise independence suffices [KNW10] (complicated).

Outline

$F_p : p > 2$, current status

- Problem: Estimate $F_p = \sum_{i=1}^{n} |f_i|^p$, $p > 2$.
- Randomized space lower bound: $\Omega(n^{1-2/p}\epsilon^{-2/p}[B-YJKS02] + (1/\epsilon^2)\log n[Wood04])$.
- Current best upper bound: $O(n^{1-2/p}\epsilon^{-2}\log(nm))$, $p = 2 + \Omega(1)$ [IW05, BGKS06, AKS11,..].

IW05: Basic idea

- Level-wise structure, $l = 0, 1, \ldots, \log M$.
- All items map to level 0, items are sub-sampled with probability $1/2$ to map to level 1, further sub-sampled with probability $1/2$ to also map to level 2, and so on.
- Keep ℓ_2 heavy-hitter structure: $H_l = HH_2^{\phi/4, \phi/8}$ at each level l. Any update (i, v) is inserted to levels H_0 through $H_{l(i)}$ if $l(i)$ is the "highest level" that i is sampled into.
- Let $k = O(1/\phi)$. $F_{2,l}^{res}(k)$ is the sum of squares of frequencies of items at level l (all but top-k) of items that are sampled into level l.
- Note that $\mathbb{E}\left[F_{2,l}^{res}(k)\right] \leq F_2^{\text{res}}(k)/2^l$ and $F_{2,l}^{res}(k) \leq 2F_2^{\text{res}}(k)/2^l$ with probability $1 - \delta$ using independence of hash function $O(\log(1/\delta))$.

Algorithm: Basic idea

- Let $F_2 \le \hat{F}_2 \le (1 + 1/20)F_2$ using standard methods.
- Basic Idea: Find heavy-hitters from the *HH* structure at each level l and their frequency estimates.
- Divide items into groups:

$$G_0 : |f_i|^2 \ge \phi \hat{F}_2, G_l : |f_i|^2 \in \left[\hat{F}_2/2^l, \hat{F}_2/2^{l-1}\right)$$

- Sampled groups: $\bar{G}_0, \ldots, \bar{G}_{\log m}$. $\bar{G}_0 : \hat{f}_i^2 \ge \phi \hat{F}_2$
- $\bar{G}_l : \phi \hat{F}_2/2^l \le \hat{f}_i^2 < \phi \hat{F}_2/2^{l-1}$ and i maps to level l.
- Estimator: Collect items into sample groups, estimate and scale.

$$\hat{F}_p = \sum_{\text{levels l}} \sum_{i \in \bar{G}_l} 2^l \hat{f}_i^p \; .$$

High Moments: Algorithm parameters

- When is an estimate reliable? [Simple version] if $\hat{f}_i \in (1 \pm \epsilon/(10p))f_i$. Then, $|\hat{f}_i|^p \in (1 \pm \epsilon/10)|f_i|^p$.
- Let $\epsilon' = \epsilon/(10p)$.
- We keep $HH_2^{\phi,\phi'}$ at each level. So error at level l is $((\phi - \phi')/2)F_{2,l} \leq ((\phi - \phi')/2)\hat{F}_2/2^{l-1}$ (w.h.p).
- Let $\phi' = \phi - \epsilon'\phi/2$. Then, error at level l is $\epsilon'\phi F_2/2^{l+1}$.

Details

- $\hat{f}_i^l =$ estimate for f_i obtained from level l HH.
- i could be discovered as a heavy-hitter at multiple levels.
- Divide G_l range $[\hat{F}_2/2^l, \hat{F}_2/2^{l-1}]$ into 3 regions:
  1. $$\text{mid}(G_l) : f_i^2 \in \left[\frac{\hat{F}_2}{(2^l(1-\epsilon'))}, \frac{\hat{F}_2}{(2^l(1+\epsilon'))}\right] .$$
     Items here are discovered as heavy only at level l (whp).
  2. $$\text{lmargin}(G_l) : f_i^2 \in \left[\frac{\hat{F}_2}{2^l}, \frac{\hat{F}_2}{(2^l(1-\epsilon'))}\right] .$$
     Items here may be classified into $\bar{G}_l$ or to $\bar{G}_{l+1}$.
  3. rmargin(G_l) : symmetric case for right margin.
- Convention: For each item i, we consider the estimate returned from the lowest level l' where $\hat{f}_i^{l'} \geq \phi \hat{F}_2/(2^{l'}(1+\epsilon'))$.

Items in "middle"

- Let $i \in G_l$.
- Probability $i \in \bar{G}_l =$ probability that i maps to level l $= 1/2^l$.

Items in margin

- Suppose $i \in \text{lmargin}(G_l)$.
- If i does not map to level l, then, $\hat{f}_i^{l'} < \hat{F}_2/2^{l'}$.
- If i maps to level l, $\hat{f}_i^l$ is a reliable estimate for f_i.
- In this case, if $\hat{f}_i^l \geq \hat{F}_2/2^l$, then, i is placed in group $\bar{G}_l$.
- If $\hat{f}_i^l < \hat{F}_2/2^l$ AND i also maps to level $l+1$, then i is placed in $\bar{G}_{l+1}$.

Items in margin

- So, for $i \in \text{lmargin}(G_l)$, the probability that i is included in $\bar{G}_{l+1}$ is

$$\Pr\left\{i \in \bar{G}_{l+1} \mid i \text{ maps to level } l\right\}$$

$$\Pr\left\{i \text{ maps to level } l+1 \text{ and } \hat{f}_i^l < \hat{F}_2/2^l \mid i \text{ maps to level } l\right\}$$

$$= \Pr\left\{\hat{f}_i^l < \hat{F}_2/2^l \mid i \text{ maps to level } l\right\} \times$$

$$\Pr\left\{i \text{ maps to level } l+1 \mid i \text{ maps to level } l\right\}$$

$$= \left(1 - \Pr\left\{\hat{f}_i^l \geq \hat{F}_2/2^l \mid i \text{ maps to level } l\right\}\right)(1/2)$$

$$= 1/2 - (1/2)\Pr\left\{i \in \bar{G}_l \mid i \text{ maps to level } l\right\}$$

or, multiplying by $\Pr\left\{i \text{ maps to level } l\right\}$,

$$2\Pr\left\{i \in \bar{G}_{l+1}\right\} + \Pr\left\{i \in \bar{G}_l \mid i \text{ maps to level l}\right\} = 1/2^l$$

gives a basic equation for analysis. [rest is straightforward].

Expectation

-
$$\hat{F}_p = \sum_{\text{levels } l} \sum_{i \in \bar{G}_l} 2^l \hat{f}_i^p$$
$$\in \sum_i \left(1 \pm \frac{\epsilon}{10}\right) |f_i|^p \sum_{\text{levels } l} 2^l x_{il}$$
where, x_{il} indicates 1 if $i \in \bar{G}_l$ and 0 otherwise. Taking expectation,
$$\mathbb{E}\left[\hat{F}_p\right] = \sum_i \left(1 \pm \frac{\epsilon}{10}\right) |f_i|^p = (1 \pm \epsilon/10) F_p \quad .$$
- Variance: calculating directly (and as before),
$$\mathrm{Var}\left[\hat{F}_p\right] = \sum_{i \in G_0} \epsilon^2 f_i^{2p} + \sum_{i \in G_l, l \geq 1} 2^l |f_i|^{2p}$$

Variance

- Since, $|f_i|^2 \leq \phi \hat{F}_2 / 2^l$,

$$\sum_{i \in G_l, l \geq 1} 2^l |f_i|^{2p} \leq \sum_{i \in G_l, l \geq 1} (2F_2) \phi |f_i|^{2p-2} \leq 2(\phi F_2) F_{2p-2}$$

- Some inequalities:

$$F_{2p-2} = \sum_i |f_i|^{2p-2} \leq (\max_i |f_i|)^{p-2}) \sum_i |f_i|^p \leq F_p^{2-2/p},$$

$$(F_2/n)^{1/2} \leq (F_p/n)^{1/p}, \text{ or, } F_2 \leq n^{1-2/p} F_p^{2/p}$$

- Combining: $\mathrm{Var}\left[\hat{F}_p\right] \leq \phi n^{1-2/p} F_p^2$.
- So ϕ should be $\epsilon^{-2} n^{1-2/p}$.

Space

- $\phi = \epsilon^{-2} n^{1-2/p}$. Also need accuracy of $\epsilon\phi$ at each level.
- A calculation shows table sizes at each level is $O(\epsilon^{-2-4/p} n^{1-2/p})$. (can be reduced to $\epsilon^{-2} n^{1-2/p}$).
- Number of levels $\log m$, can be reduced to $\log n$ because higher levels contribute less than ϵ mass to F_p. Further reduced to $O(1)$ levels [AKO10]
- Number of tables per level is $O(\log n)$.
- Space $O(n^{1-2/p} \epsilon^{-2-4/p} \log(m) \log(n))$ words.
- Randomness: can be reduced to use $O(\log n)$ bits.

Outline

Compressed Sensing: Problem and Motivation

- x is n-dimensional vector, e.g., image.
- We wish to recover a "close" approximation of x, but by making $m \ll n$ observations of x.
- Observations are linear: Ax, where A is a measurement matrix.
- Closeness of approximation: close to the best k-sparse approximation of x.
- k-sparse vector: has at most k non-zero entries.
- Sparse revovery with ℓ_p/ℓ_q guarantees. Return $\hat{x}$ such that

  $$\|x - \hat{x}\|_p \leq C \min_{k\text{-sparse} x'} \|x - x'\|_q$$

  where, C is a small constant.
- x^* achieving $\min_{k\text{-sparse} x'} \|x - x'\|_q$ has the top-k values $|x_i|$.

Overview

- Sparse ℓ_1/ℓ_2 guarantees such that

$$\|x - \hat{x}\|_2 \leq \frac{C}{\sqrt{k}} \min_{k\text{-sparse} x'} \|x - x'\|_1$$

  and C is a small constant.
- $\hat{x}$ itself need not be k-sparse (minor point).

Overview

- Sparse ℓ_1/ℓ_2 guarantees such that

$$\|x - \hat{x}\|_2 \leq \frac{C}{\sqrt{k}} \min_{k\text{-sparse} x'} \|x - x'\|_1$$

  and C is a small constant.
- $\hat{x}$ itself need not be k-sparse (minor point).
- Consider LP:

$$\begin{aligned} &\min \|x^*\|_1 \\ &\text{s.t. } Ax = Ax^* \end{aligned}$$

Overview

- Sparse ℓ_1/ℓ_2 guarantees such that

$$\|x - \hat{x}\|_2 \leq \frac{C}{\sqrt{k}} \min_{k\text{-sparse} x'} \|x - x'\|_1$$

  and C is a small constant.
- $\hat{x}$ itself need not be k-sparse (minor point).
- Consider LP:

$$\min \|x^*\|_1$$
$$\text{s.t. } Ax = Ax^*$$

- This is an LP.

$$\min t_1 + t_2 + \ldots + t_n$$
$$s.t. - t_i \leq x_i^* \leq t_i$$
$$Ax = Ax^*$$

Motivation for LP

$$\min \|x^*\|_1$$
$$\text{s.t. } Ax = Ax^*$$

- Seems mysterious at first.
- Actual goal should have been to minimize $\|x\|_0$.
- For carefully chosen A, this has (almost) the same effect.
- Choice of $\|x^*\|_1$ crucial, $\|x^*\|_2$ doesn't work.

Statement of theorem

- **Theorem** [CRT06, D06] If each entry of $A_{m \times n}$ is i.i.d. $N(0,1)$ and $m = \Theta(k \log(n/k))$, then with high probability (over the randomness of A) the output x' of LP satisfies:

$$\|x - x'\|_2 \leq \frac{C}{\sqrt{k}} \min_{k\text{-sparse} x''} \|x - x''\|_1 \quad .$$

- Remarks:
  1. "One sketch for all": guarantee is deterministic (construction is probabilistic).
  2. $N(0,1)$ not crucial: distributions satisfying $J - L$ also work.
  3. ℓ_1/ℓ_2 mixed guarantee essential: no similar guarantee possible for ℓ_2/ℓ_2.

Restricted Isometry Property

- A matrix is (k, δ)-RIP if for every k-sparse x

$$(1-\delta)\|x\|_2 \leq \|Ax\|_2 \leq (1+\delta)\|x\|_2$$

- Property 1: If each entry of $A_{m\times n}$ is i.i.d. $N(0,1)$ and $m = O(k\log(n/k))$, then, A is $(k, 1/3)$-RIP.
- Property 2: A $(4k, 1/3)$-RIP matrix A implies: the output x' of LP satisfies:

$$\|x - x'\|_2 \leq \frac{C}{\sqrt{k}} \min_{k\text{-sparse} x''} \|x - x''\|_1 \quad .$$

Normal i.i.d. matrices are RIP whp

- Suffice to assume that $\|x\|_2 = 1$.
- We will take the union bound over all k-subsets T of $\{1, \ldots, n\}$ such that support$(x) = T$. There are $\binom{n}{k}$ such sets.
- Consider A_T: the columns of A corresponding to positions in T and $x' = x_T$ similarly. So x' is k-dimensional and A_T is $m \times k$.
- We need to show that with probability $1 - 1/(8\binom{n}{k})$, for any x' on a k-dimensional unit ball B, we have,

$$2/3 \leq \|Ax'\|_2 \leq 4/3$$

k-dimensional unit ball preservation

- An ϵ-net N of a set B is a subset of B such that for any $x' \in B$, there exists $x_1 \in N$ such that $\|x - x'\| < \epsilon$. Let $\epsilon = 1/7$.
- Fact: there exists an ϵ-net for unit k-dimensional ball of size $(1/\epsilon)^{\Theta(k)}$.
- By J-L Lemma, for all points in N, we have,

$$7/8 \leq \|Ax\|_2 \leq 8/7, \text{ with prob. } 1 - e^{-\Theta(m)} \quad .$$

- To extend to all $x' \in B$, we write $\Delta = x' - x_1$. Now, $\|\Delta\| < 1/7$. Normalize Δ. Recurse.
- So, $x' = x_1 + b_2 x_2 + \ldots +$ such that
  1. all $x_i \in N$
  2. $b_i \leq 1/7^i$.

k dim unit ball preservation

- So, we get,

$$\|Ax'\|_2 \le \sum_{i \ge 0} b_i \|Ax_i\| \le \sum_{i \ge 0} (8/7)(1/7)^i \le (8/7)(7/6) = 4/3$$

- The other case (lower bound on $\|Ax'\|_2$ is similar.
- So, for x' in the unit k-dimensional ball, we get $2/3 \le \|Ax'\|_2 \le 4/3$
- Failure probability using union bound:

$$\binom{n}{k} 7^{O(k)} e^{-\Theta(m)} = (n/k)^{\Theta(k)} e^{-\Theta(m)}$$

which is at most $1/8$ if $m = \Theta(k \log(n/k)$.