BPTree: improved space for insertion-only ℓ_2 heavy hitters

Jelani Nelson

Harvard

March 19, 2018

joint work with Vladimir Braverman (Johns Hopkins), Stephen Chestnut (G-Research), Nikita Ivkin (Johns Hopkins), Zhengyu Wang (Harvard), and David Woodruff (CMU)

Finding frequent items

A (fake) search engine query log from Nov 7th:

18:58:02	`gmail`
18:59:12	`mlb playoffs`
19:07:40	`wiki trump`
19:07:42	`cream of wheat wiki`
19:07:58	`p vs np`
19:09:37	`aa flight status 1597`
19:10:14	`halloween costumes`
19:10:18	`wiki van emde boas`
19:11:28	`edx wiki`

Finding frequent items

A (fake) search engine query log from Nov 7th:

18:58:02	`gmail`
18:59:12	`mlb playoffs`
19:07:40	**`wiki`** `trump`
19:07:42	`cream of wheat` **`wiki`**
19:07:58	`p vs np`
19:09:37	`aa flight status 1597`
19:10:14	`halloween costumes`
19:10:18	**`wiki`** `van emde boas`
19:11:28	`edx` **`wiki`**

Finding heavy hitters

Problem: Given stream of items (e.g. words) coming from some universe $\mathcal{U}$ (e.g. English dictionary), report a small list $L \subset \mathcal{U}$ containing all "frequent" items

- "frequent/heavy" depends on some input parameter ε
- for word $i \in \mathcal{U}$, "heavy" means $x_i > \varepsilon \|x\|_2$, where $x \in \mathbb{R}^{|\mathcal{U}|}$ has x_i equal to # occurrences of word i in stream

Finding heavy hitters

Problem: Given stream of items (e.g. words) coming from some universe $\mathcal{U}$ (e.g. English dictionary), report a small list $L \subset \mathcal{U}$ containing all "frequent" items

- "frequent/heavy" depends on some input parameter ε
- for word $i \in \mathcal{U}$, "heavy" means $x_i > \varepsilon \|x\|_2$, where $x \in \mathbb{R}^{|\mathcal{U}|}$ has x_i equal to # occurrences of word i in stream
- trivial solution: use $n = |\mathcal{U}|$ words of memory

Finding heavy hitters

Problem: Given stream of items (e.g. words) coming from some universe $\mathcal{U}$ (e.g. English dictionary), report a small list $L \subset \mathcal{U}$ containing all "frequent" items

- "frequent/heavy" depends on some input parameter ε
- for word $i \in \mathcal{U}$, "heavy" means $x_i > \varepsilon \|x\|_2$, where $x \in \mathbb{R}^{|\mathcal{U}|}$ has x_i equal to # occurrences of word i in stream
- trivial solution: use $n = |\mathcal{U}|$ words of memory
- **Goal:** using $\ll n$ memory, output small such L (e.g. $|L| \leq O(1/\varepsilon^2)$, which is max possible # of heavy items)

Finding heavy hitters

Problem: Given stream of items (e.g. words) coming from some universe $\mathcal{U}$ (e.g. English dictionary), report a small list $L \subset \mathcal{U}$ containing all "frequent" items

- "frequent/heavy" depends on some input parameter ε
- for word $i \in \mathcal{U}$, "heavy" means $x_i > \varepsilon\|x\|_2$, where $x \in \mathbb{R}^{|\mathcal{U}|}$ has x_i equal to # occurrences of word i in stream
- trivial solution: use $n = |\mathcal{U}|$ words of memory
- **Goal:** using $\ll n$ memory, output small such L (e.g. $|L| \leq O(1/\varepsilon^2)$), which is max possible # of heavy items)
- Henceforth: $k := 1/\varepsilon^2$, want to find (ℓ_2-approximate) "top-k"

Finding heavy hitters

Problem: Given stream of items (e.g. words) coming from some universe $\mathcal{U}$ (e.g. English dictionary), report a small list $L \subset \mathcal{U}$ containing all "frequent" items

- "frequent/heavy" depends on some input parameter ε
- for word $i \in \mathcal{U}$, "heavy" means $x_i > \varepsilon\|x\|_2$, where $x \in \mathbb{R}^{|\mathcal{U}|}$ has x_i equal to # occurrences of word i in stream
- trivial solution: use $n = |\mathcal{U}|$ words of memory
- **Goal:** using $\ll n$ memory, output small such L (e.g. $|L| \le O(1/\varepsilon^2)$), which is max possible # of heavy items)
- Henceforth: $k := 1/\varepsilon^2$, want to find (ℓ_2-approximate) "top-k"
- Could define in terms of $\|x\|_p$ for other p, but known $f(k) \cdot n^{o(1)}$ space possible iff $p \le 2$ [BarYossef-Jayram-Kumar-Sivakumar'04], and up to slight change in problem defn can black-box solve ℓ_p HH optimally using optimal ℓ_q algo. if $p < q$ [Jowhari-Sağlam-Tardos'11].

Problem Statement

Problem name: "ℓ_2 heavy hitters in insertion-only streams"

Definition

Index $i \in [n]$ is a *k-heavy hitter* (or k-HH) if $|x_i| > \frac{1}{\sqrt{k}}\|x\|_2$

Problem Statement

Problem name: "ℓ_2 heavy hitters in insertion-only streams"

Definition

Index $i \in [n]$ is a *k-heavy hitter* (or k-HH) if $|x_i| > \frac{1}{\sqrt{k}}\|x\|_2$

`query()`: Must output $L \subseteq [n]$ s.t.

(1) $|L| = O(k)$, and

(2) L contains every k-HH

Works on heavy hitters

- sampling (folklore)
- Frequent [Misra-Gries'82]
- LossyCounting [Singh-Motwani'02]
- SpaceSaving [Metwally-Agrawal-ElAbbadi'05]
- SampleAndHold [Estan-Varghese'03]
- Multi-stage bloom filters [Chabchoub-Fricker-Mohamed'09]
- Sketch-guided sampling [Kumar-Xu'06]
- CountMin sketch [Cormode-Muthukrishnan'05]
- CountMin sketch with dyadic trick [Cormode-Muthukrishnan'05]
- CountSketch [Charikar-Chen-FarachColton'02]
- CountSketch with codes [Pagh'13]
- HSS (Hierarchical CountSketch) [Cormode-Hadjieleftheriou'08]
- CountSieve [Braverman-Chestnut-Ivkin-Woodruff'16]
- BDW [Bhattacharyya-Dey-Woodruff'16]
- BPTree [Braverman-Chestnut-Ivkin-Nelson-Wang-Woodruff'17]
- ExpanderSketch [Larsen-Nelson-Nguyễn-Thorup'16]

- sampling (folklore)
- Frequent [Misra-Gries'82]
- LossyCounting [Singh-Motwani'02]
- SpaceSaving [Metwally-Agrawal-ElAbbadi'05]
- SampleAndHold [Estan-Varghese'03]
- Multi-stage bloom filters [Chabchoub-Fricker-Mohamed'09]
- Sketch-guided sampling [Kumar-Xu'06]
- CountMin sketch [Cormode-Muthukrishnan'05]
- CountMin sketch with dyadic trick [Cormode-Muthukrishnan'05]
- CountSketch [Charikar-Chen-FarachColton'02]
- CountSketch with codes [Pagh'13]
- HSS (Hierarchical CountSketch) [Cormode-Hadjieleftheriou'08]
- CountSieve [Braverman-Chestnut-Ivkin-Woodruff'16]
- BDW [Bhattacharyya-Dey-Woodruff'16]
- **BPTree [Braverman-Chestnut-Ivkin-Nelson-Wang-Woodruff'17]**
- ExpanderSketch [Larsen-Nelson-Nguyễn-Thorup'16]

Bounds attained for ℓ_2-heavy hitters

(k denotes $1/\varepsilon^2$)

Insertion-only

reference	data structure	space (words)
[Charikar, Chen, Farach-Colton'02]	CountSketch	$k \log n$
[Braverman, Chestnut, Ivkin, Woodruff'16]	CountSieve	$k \log k \log\log n$
[BCIW+Nelson+Wang'17]	BPTree	$k \log k$

(all for failure probability 1/100)

Bounds attained for ℓ_2-heavy hitters

(k denotes $1/\varepsilon^2$)

Insertion-only

reference	data structure	space (words)
[Charikar, Chen, Farach-Colton'02]	CountSketch	$k \log n$
[Braverman, Chestnut, Ivkin, Woodruff'16]	CountSieve	$k \log k \log\log n$
[BCIW+Nelson+Wang'17]	BPTree	$k \log k$

(all for failure probability 1/100)

OPEN: $O(k)$ words?

Insertion-only ℓ_2 heavy hitters: the BPTree

[Braverman-Chestnut-Ivkin-Nelson-Wang-Woodruff'17]

BPTree

Plan of attack

- **Defn.** $H \in [n]$ is **super-heavy** if $x_H^2 > 1000 \sum_{j \neq H} x_j^2$
- We will reduce finding $L \subset [n]$ containing all heavy hitters, $|L| = O(k)$, to the following problem:

BPTree

Plan of attack

- **Defn.** $H \in [n]$ is **super-heavy** if $x_H^2 > 1000 \sum_{j \neq H} x_j^2$
- We will reduce finding $L \subset [n]$ containing all heavy hitters, $|L| = O(k)$, to the following problem:

  if $\exists i$ super-heavy, find it with probability $9/10$

  (if no super-heavy item exists, then arbitrary output allowed)

BPTree

Plan of attack

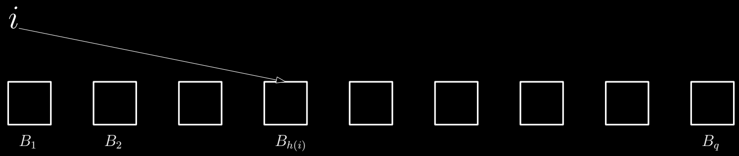
- **Defn.** $H \in [n]$ is **super-heavy** if $x_H^2 > 1000 \sum_{j \neq H} x_j^2$
- We will reduce finding $L \subset [n]$ containing all heavy hitters, $|L| = O(k)$, to the following problem:

  if $\exists i$ super-heavy, find it with probability $9/10$

  (if no super-heavy item exists, then arbitrary output allowed)
- If can solve "super-heavy" in space S, our final algorithm will have space $O(S \cdot k \log k) \implies$ want $S = O(1)$

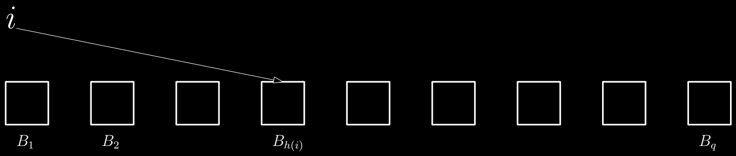
Reduction to finding super-heavy items

The reduction: $h : [n] \to [q]$ from 2-wise indep. family, $q = 5000k$



Reduction to finding super-heavy items

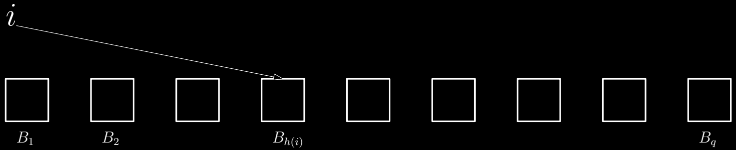
The reduction: $h : [n] \to [q]$ from 2-wise indep. family, $q = 5000k$



Let HH be set of ε-heavy hitters, and say $i \in HH$.

Reduction to finding super-heavy items

The reduction: $h : [n] \to [q]$ from 2-wise indep. family, $q = 5000k$



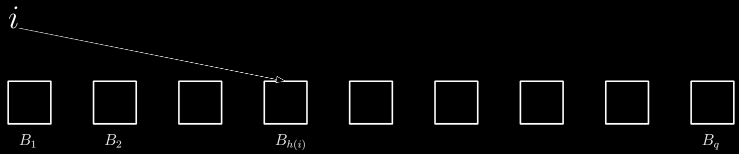
Let HH be set of ε-heavy hitters, and say $i \in HH$.

$$\mathbb{P}_h(\exists j \in HH \backslash \{i\},\ h(j) = h(i)) \le \frac{1}{5000} \quad \text{(i isolated from rest of HH)}$$

$$\mathbb{P}_h\Big(\sum_{\substack{j \notin HH \\ h(j)=h(i)}} x_j^2 \ge \frac{1}{1000k}\|x\|_2^2\Big) \le \frac{1}{5} \quad \text{(very little non-HH mass in $B_{h(i)}$)}$$

Reduction to finding super-heavy items

The reduction: $h : [n] \to [q]$ from 2-wise indep. family, $q = 5000k$



Let HH be set of ε-heavy hitters, and say $i \in HH$.

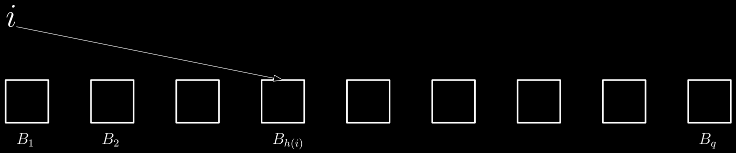
$$\mathop{\mathbb{P}}_{h}(\exists j \in HH \backslash \{i\},\ h(j) = h(i)) \le \frac{1}{5000} \quad \text{(i isolated from rest of HH)}$$

$$\mathop{\mathbb{P}}_{h}\Big(\sum_{\substack{j \notin HH \\ h(j)=h(i)}} x_j^2 \ge \frac{1}{1000k}\|x\|_2^2\Big) \le \frac{1}{5} \quad \text{(very little non-HH mass in $B_{h(i)}$)}$$

$\implies$ i is super-heavy in $B_{h(i)}$ with at least $\approx 4/5$ probability

Reduction to finding super-heavy items

The reduction: $h : [n] \to [q]$ from 2-wise indep. family, $q = 5000k$



Let HH be set of ε-heavy hitters, and say $i \in HH$.

$$\mathbb{P}_h(\exists j \in HH \backslash \{i\},\ h(j) = h(i)) \leq \frac{1}{5000} \quad \text{(i isolated from rest of HH)}$$

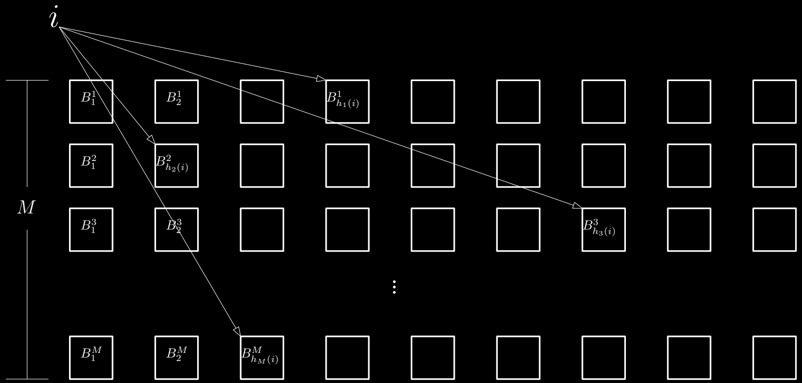
$$\mathbb{P}_h\Big(\sum_{\substack{j \notin HH \\ h(j)=h(i)}} x_j^2 \geq \frac{1}{1000k}\|x\|_2^2\Big) \leq \frac{1}{5}\quad \text{(very little non-HH mass in $B_{h(i)}$)}$$

$\implies$ i is super-heavy in $B_{h(i)}$ with at least $\approx 4/5$ probability

- Each B_r stores a data structure implementing a solution to the "super-heavy" problem w/ success prob. $\geq 9/10$, so we find i w.p. $\geq \frac{9}{10} \cdot (1 - \frac{1}{5} - \frac{1}{5000}) > \frac{7}{10}$

Final reduction

The reduction: $h_1, \ldots, h_M : [n] \to [q]$ from 2-wise indep. family, $q = 5000k$, $M = \Theta(\log k)$



Output
$L = \{i : i \text{ reported as super-heavy in at least half the rows}\}$

Analysis: Use last slide + Chernoff and union bound

Solving “super-heavy” in $O(1)$ words of memory

Solving "super-heavy" in $O(1)$ words of memory

Will make use of ...

Core lemma: If $0 = y^{(0)}, \ldots, y^{(T)}$ is the evolution of a vector updated in an insertion-only stream and $\sigma \in \{-1, 1\}^n$ has 4-wise independent entries, then

$$\mathop{\mathbb{E}}_{\sigma} \sup_{t \in [T]} |\left\langle \sigma, y^{(t)} \right\rangle| \lesssim \|y^{(T)}\|_2.$$

($y^{(t)}$ is frequency vector after first t updates in stream)

Basic idea to make use core lemma

- Wishful thinking: assume we know $\|x^{(m)}\|_2$ exactly.
  (m is the length of the stream)

Basic idea to make use core lemma

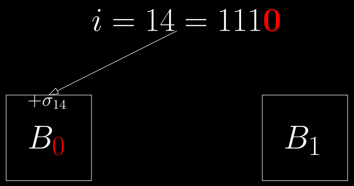
- Wishful thinking: assume we know $\|x^{(m)}\|_2$ exactly.
  (m is the length of the stream)
- $H \in \{1, \ldots, n\}$ is the ID of the super-heavy item
  we will try to learn $H = H[0]H[1] \cdots H[\log n]$ bit by bit
  (will learn these bits iteratively, from $j = 0$ to $\log n$)

Basic idea to make use core lemma

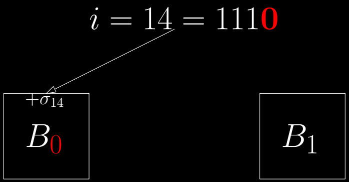
- Wishful thinking: assume we know $\|x^{(m)}\|_2$ exactly.
  (m is the length of the stream)
- $H \in \{1, \ldots, n\}$ is the ID of the super-heavy item
  we will try to learn $H = H[0]H[1] \cdots H[\log n]$ bit by bit
  (will learn these bits iteratively, from $j = 0$ to $\log n$)
- Learning H_0:
  - $\sigma \in \{-1, 1\}^n$ from 4-wise independent family
  - instantiate two counters B_0, B_1

Basic idea to make use core lemma

- Wishful thinking: assume we know $\|x^{(m)}\|_2$ exactly.
  (m is the length of the stream)
- $H \in \{1, \ldots, n\}$ is the ID of the super-heavy item
  we will try to learn $H = H[0]H[1] \cdots H[\log n]$ bit by bit
  (will learn these bits iteratively, from $j = 0$ to $\log n$)
- Learning H_0:
  - $\sigma \in \{-1, 1\}^n$ from 4-wise independent family
  - instantiate two counters B_0, B_1
  - when we see $i \in [n]$ in stream, increment $B_{i[0]}$ by σ_i



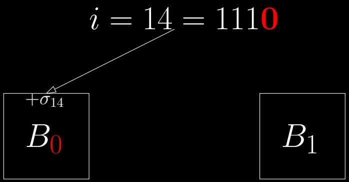
Basic idea to make use core lemma



- For the sake of illustration, let's say $H[0] = 1$
- $\implies B_1 = \pm x_H + \sum_{i \neq H, H[0]=1} \sigma_i x_i$

  $B_0 = \sum_{H[0]=0} \sigma_i x_i$

Basic idea to make use core lemma

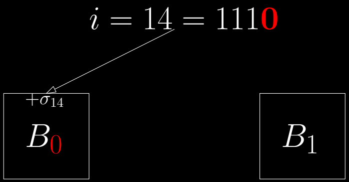


- For the sake of illustration, let's say $H[0] = 1$
- $\implies B_1 = \pm x_H + \sum_{i \neq H, H[0]=1} \sigma_i x_i$

  $B_0 = \sum_{H[0]=0} \sigma_i x_i$
- Super-heaviness:

  $x_H^2 > 1000 \sum_{i \neq H} x_i^2 \implies \frac{x_H^{(m)}}{\|x^{(m)}\|_2} > \sqrt{\frac{1000}{1001}} > .999$

Basic idea to make use core lemma



- For the sake of illustration, let's say $H[0] = 1$
- $\implies B_1 = \pm x_H + \sum_{i \neq H, H[0]=1} \sigma_i x_i$

  $B_0 = \sum_{H[0]=0} \sigma_i x_i$
- Super-heaviness:

  $x_H^2 > 1000 \sum_{i \neq H} x_i^2 \implies \frac{x_H^{(m)}}{\|x^{(m)}\|_2} > \sqrt{\frac{1000}{1001}} > .999$
- Remember *we know* $\|x^{(m)}\|_2$. Wait until some $|B_j| > .1\|x^{(m)}\|_2$, then we learn $H[0] = j$.

  "Core Lemma" applied twice (once to each bucket) implies two $\sum$'s above probably never exceed $.01\|x^{(m)}\|_2$

Basic idea: making it work

- We know how to learn $H[0]$. How about future bits? Iterate?

Basic idea: making it work

- We know how to learn $H[0]$. How about future bits? Iterate?
- Problem: learned $H[0]$ after $\approx 10\%$ of H's occurrences. Can only do that 10 times, but need to learn $\log n$ bits of H.

Basic idea: making it work

- We know how to learn $H[0]$. How about future bits? Iterate?
- Problem: learned $H[0]$ after $\approx 10\%$ of H's occurrences. Can only do that 10 times, but need to learn $\log n$ bits of H.
- Pseudofix: when learning $H[j]$, ignore any stream index whose bits $1, \ldots, j-1$ don't match what we already learned

  (**idea:** filtering cuts out $\approx \frac{1}{2^j}$ fraction of noise, so can afford to say we've learned $H[j]$ after some $|B_r| > (\frac{9}{10})^j \cdot .1\|x^{(m)}\|_2$)

Basic idea: making it work

- We know how to learn $H[0]$. How about future bits? Iterate?
- Problem: learned $H[0]$ after $\approx 10\%$ of H's occurrences. Can only do that 10 times, but need to learn $\log n$ bits of H.
- Pseudofix: when learning $H[j]$, ignore any stream index whose bits $1, \ldots, j-1$ don't match what we already learned

  (**idea:** filtering cuts out $\approx \frac{1}{2^j}$ fraction of noise, so can afford to say we've learned $H[j]$ after some $|B_r| > (\frac{9}{10})^j \cdot .1\|x^{(m)}\|_2$)
- Learn all bits after $.1 \cdot \sum_j (\frac{9}{10})^j < 1$ fraction of H's occurrences

Basic idea: making it work

- We know how to learn $H[0]$. How about future bits? Iterate?
- Problem: learned $H[0]$ after $\approx 10\%$ of H's occurrences. Can only do that 10 times, but need to learn $\log n$ bits of H.
- Pseudofix: when learning $H[j]$, ignore any stream index whose bits $1, \ldots, j-1$ don't match what we already learned (**idea:** filtering cuts out $\approx \frac{1}{2^j}$ fraction of noise, so can afford to say we've learned $H[j]$ after some $|B_r| > (\frac{9}{10})^j \cdot .1\|x^{(m)}\|_2$)
- Learn all bits after $.1 \cdot \sum_j (\frac{9}{10})^j < 1$ fraction of H's occurrences
- Problem: Might not be that only $\frac{1}{2^j}$ fraction of mass matches H's length-j binary prefix. i.e. mass isn't randomly distributed.

Basic idea: making it work

- We know how to learn $H[0]$. How about future bits? Iterate?
- Problem: learned $H[0]$ after $\approx 10\%$ of H's occurrences. Can only do that 10 times, but need to learn $\log n$ bits of H.
- Pseudofix: when learning $H[j]$, ignore any stream index whose bits $1, \ldots, j-1$ don't match what we already learned
  (**idea:** filtering cuts out $\approx \frac{1}{2^j}$ fraction of noise, so can afford to say we've learned $H[j]$ after some $|B_r| > (\frac{9}{10})^j \cdot .1 \|x^{(m)}\|_2$)
- Learn all bits after $.1 \cdot \sum_j (\frac{9}{10})^j < 1$ fraction of H's occurrences
- Problem: Might not be that only $\frac{1}{2^j}$ fraction of mass matches H's length-j binary prefix. i.e. mass isn't randomly distributed.
- Final fix: Pick 2-wise permutation $\pi : [n^3] \to [n^3]$ and for each stream update i, feed $\pi(i)$ to algorithm. Then indices **are** random, and we can learn $H' = \pi(H)$. Then return $\pi^{-1}(H')$.

Final algorithm: removing the remaining assumption

- "Super-heavy" algorithm assumed we knew $\|x^{(m)}\|_2$ exactly (it's actually enough to know it up to a factor of 2)

Final algorithm: removing the remaining assumption

- "Super-heavy" algorithm assumed we knew $\|x^{(m)}\|_2$ exactly (it's actually enough to know it up to a factor of 2)
- Fix: guess $\|x^{(m)}\|_2 \approx 1, 2, \ldots, 2^{R-1}$ in parallel ($R = 10$, say)

Final algorithm: removing the remaining assumption

- "Super-heavy" algorithm assumed we knew $\|x^{(m)}\|_2$ exactly (it's actually enough to know it up to a factor of 2)
- Fix: guess $\|x^{(m)}\|_2 \approx 1, 2, \ldots, 2^{R-1}$ in parallel ($R = 10$, say)
  - For each of R guesses in parallel, instantiate "super-heavy" alg

Final algorithm: removing the remaining assumption

- "Super-heavy" algorithm assumed we knew $\|x^{(m)}\|_2$ exactly (it's actually enough to know it up to a factor of 2)
- Fix: guess $\|x^{(m)}\|_2 \approx 1, 2, \ldots, 2^{R-1}$ in parallel ($R = 10$, say)
  - For each of R guesses in parallel, instantiate "super-heavy" alg
  - Run tracker in parallel; continously outputs $a_t \approx \|x^{(t)}\|_2$

Final algorithm: removing the remaining assumption

- "Super-heavy" algorithm assumed we knew $\|x^{(m)}\|_2$ exactly (it's actually enough to know it up to a factor of 2)
- Fix: guess $\|x^{(m)}\|_2 \approx 1, 2, \ldots, 2^{R-1}$ in parallel ($R = 10$, say)
  - For each of R guesses in parallel, instantiate "super-heavy" alg
  - Run tracker in parallel; continously outputs $a_t \approx \|x^{(t)}\|_2$
  - Say currently running parallel algs for $2^j, 2^{j+1}, \ldots, 2^{R+j-1}$ when $a_t > 2^j$, kill 2^j process and start new process for 2^{R+j}

Final algorithm: removing the remaining assumption

- "Super-heavy" algorithm assumed we knew $\|x^{(m)}\|_2$ exactly (it's actually enough to know it up to a factor of 2)
- Fix: guess $\|x^{(m)}\|_2 \approx 1, 2, \ldots, 2^{R-1}$ in parallel ($R = 10$, say)
  - For each of R guesses in parallel, instantiate "super-heavy" alg
  - Run tracker in parallel; continously outputs $a_t \approx \|x^{(t)}\|_2$
  - Say currently running parallel algs for $2^j, 2^{j+1}, \ldots, 2^{R+j-1}$ when $a_t > 2^j$, kill 2^j process and start new process for 2^{R+j}
  - The newly booted process missed out on some prefix of the stream, but if $\|x^{(m)}\|_2$ actually ends up $\approx 2^{R+j}$, we only missed out on mass leading up to $\|x\|_2 \approx 2^j$, so only missed $\approx 2^{-R}$ fraction of the final x_H occurrences. **QED**.

How can we prove the core lemma?

How can we prove the core lemma?

Core lemma: If $0 = y^{(0)}, \ldots, y^{(T)}$ is the evolution of a vector updated in an insertion-only stream and $\sigma \in \{-1, 1\}^n$ has 4-wise independent entries, then

$$\mathop{\mathbb{E}}_{\sigma} \sup_{t \in [T]} \left| \left\langle \sigma, y^{(t)} \right\rangle \right| \lesssim \|y^{(T)}\|_2.$$

($y^{(t)}$ is frequency vector after first t updates in stream)

Warmup

Simple random walk on a line.

$$y^{(t)} = (\overbrace{1, \ldots, 1}^{t}, \overbrace{0, 0, 0, 0, \ldots, 0}^{n-t})$$

- $\sigma \in \{-1, 1\}^n$, row of Π, has 4-wise independent entries

Warmup

Simple random walk on a line.

$$y^{(t)} = (\overbrace{1, \ldots, 1}^{t}, \overbrace{0, 0, 0, 0, \ldots, 0}^{n-t})$$

- $\sigma \in \{-1, 1\}^n$, row of Π, has 4-wise independent entries
- $\langle \sigma, y^{(t)} \rangle$: the location of a random walk on $\mathbb{Z}$ after t steps, starting at 0, each step goes left/right with equal probability

Warmup

Simple random walk on a line.

$$y^{(t)} = (\overbrace{1, \dots, 1}^{t}, \overbrace{0, 0, 0, 0, \dots, 0}^{n-t})$$

- $\sigma \in \{-1, 1\}^n$, row of Π, has 4-wise independent entries
- $\langle \sigma, y^{(t)} \rangle$: the location of a random walk on $\mathbb{Z}$ after t steps, starting at 0, each step goes left/right with equal probability
- Kolmogorov/Lévy maximal inequalities:
  $\mathbb{E}_\sigma \sup_{t \in [T]} |\langle \sigma, y^{(t)} \rangle| \lesssim \sqrt{T}$
  (if σ has independent entries)

Warmup

Simple random walk on a line.

$$y^{(t)} = (\overbrace{1, \dots, 1}^{t}, \overbrace{0, 0, 0, 0, \dots, 0}^{n-t})$$

- $\sigma \in \{-1, 1\}^n$, row of Π, has 4-wise independent entries
- $\langle \sigma, y^{(t)} \rangle$: the location of a random walk on $\mathbb{Z}$ after t steps, starting at 0, each step goes left/right with equal probability
- Kolmogorov/Lévy maximal inequalities:
  $\mathbb{E}_\sigma \sup_{t \in [T]} | \langle \sigma, y^{(t)} \rangle | \lesssim \sqrt{T}$
  (if σ has independent entries)
- Will now show a proof (outline) of above standard result that can be adapted to handle 4-wise independent σ_i

Suprema of stochastic processes

Suprema of stochastic processes

We have $V \subset B_{\ell_2^n}$ and want to upper bound

$$\alpha(V) := \mathbb{E} \sup_{v \in V} |\langle \sigma, v \rangle|$$

(in our case $V = \{\frac{y^{(t)}}{\sqrt{T}}\}_{t=0}^{T}$ and want to show $\alpha(V) \lesssim 1$)

Suprema of stochastic processes

We have $V \subset B_{\ell_2^n}$ and want to upper bound

$$\alpha(V) := \mathbb{E} \sup_{v \in V} |\langle \sigma, v \rangle|$$

Method 1 (union bound):

Suprema of stochastic processes

We have $V \subset B_{\ell_2^n}$ and want to upper bound

$$\alpha(V) := \mathbb{E} \sup_{v \in V} |\langle \sigma, v \rangle|$$

Method 1 (union bound): Khintchine inequality says $\mathbb{P}_\sigma(|\langle \sigma, v \rangle| > \lambda) \leq 2e^{-\lambda^2/(2\|v\|_2^2)}$.

Suprema of stochastic processes

We have $V \subset B_{\ell_2^n}$ and want to upper bound

$$\alpha(V) := \mathbb{E} \sup_{v \in V} | \langle \sigma, v \rangle |$$

Method 1 (union bound): Khintchine inequality says $\mathbb{P}_\sigma(| \langle \sigma, v \rangle | > \lambda) \le 2e^{-\lambda^2/(2\|v\|_2^2)}$.

$$\begin{aligned}
\alpha(V) &= \int_0^\infty \mathbb{P}(\sup_{v \in V} | \langle \sigma, v \rangle | > \lambda) d\lambda \\
&= \int_0^\tau \overbrace{\mathbb{P}(\sup_{v \in V} | \langle \sigma, v \rangle | > \lambda)}^{\le 1} d\lambda + \int_\tau^\infty \overbrace{\mathbb{P}(\sup_{v \in V} | \langle \sigma, v \rangle | > \lambda)}^{\le \sum_{v \in V} \mathbb{P}(|\langle \sigma, v \rangle| > \lambda)} d\lambda \\
&\le \tau + |V| \cdot 2e^{-\tau^2/2} \\
&\lesssim \sqrt{\lg |V|} \ (\text{set } \tau = C\sqrt{\lg |V|})
\end{aligned}$$

Suprema of stochastic processes

We have $V \subset B_{\ell_2^n}$ and want to upper bound

$$\alpha(V) := \mathbb{E} \sup_{v \in V} |\langle \sigma, v \rangle|$$

Method 2 (ε-net):

Suprema of stochastic processes

We have $V \subset B_{\ell_2^n}$ and want to upper bound

$$\alpha(V) := \mathbb{E} \sup_{v \in V} |\langle \sigma, v \rangle|$$

Method 2 (ε-net): V' is an ε-net of V in ℓ_2 if $\forall v \in V \; \exists v' \in V'$ such that $\|v - v'\|_2 \leq \varepsilon$

Suprema of stochastic processes

We have $V \subset B_{\ell_2^n}$ and want to upper bound

$$\alpha(V) := \mathbb{E} \sup_{v \in V} |\langle \sigma, v \rangle|$$

Method 2 (ε-net): V' is an ε-net of V in ℓ_2 if $\forall v \in V\ \exists v' \in V'$ such that $\|v - v'\|_2 \le \varepsilon$

$$\begin{aligned}
\mathbb{E} \sup_{v \in V} |\langle \sigma, v \rangle| &= \mathbb{E} \sup_{v \in V} \left|\left\langle \sigma, v' + (v - v') \right\rangle\right| \\
&\le \mathbb{E} \sup_{v' \in V'} \left|\left\langle \sigma, v' \right\rangle\right| + \mathbb{E} \sup_{v \in V} \underbrace{\left|\left\langle \sigma, v - v' \right\rangle\right|}_{\le \varepsilon\sqrt{n}} \\
&\lesssim \sqrt{\lg |V'|} + \varepsilon\sqrt{n} \\
&:= \lg^{1/2} \mathcal{N}(V, \ell_2, \varepsilon) + \varepsilon\sqrt{n}
\end{aligned}$$

Suprema of stochastic processes

We have $V \subset B_{\ell_2^n}$ and want to upper bound

$$\alpha(V) := \mathbb{E} \sup_{v \in V} |\langle \sigma, v \rangle|$$

Method 2 (ε-net): V' is an ε-net of V in ℓ_2 if $\forall v \in V\ \exists v' \in V'$ such that $\|v - v'\|_2 \le \varepsilon$

$$\begin{aligned}
\mathbb{E} \sup_{v \in V} |\langle \sigma, v \rangle| &= \mathbb{E} \sup_{v \in V} \left|\left\langle \sigma, v' + (v - v') \right\rangle\right| \\
&\le \mathbb{E} \sup_{v' \in V'} \left|\left\langle \sigma, v' \right\rangle\right| + \mathbb{E} \sup_{v \in V} \underbrace{\left|\left\langle \sigma, v - v' \right\rangle\right|}_{\le \varepsilon\sqrt{n}} \\
&\lesssim \sqrt{\lg |V'|} + \varepsilon\sqrt{n} \\
&:= \lg^{1/2} \mathcal{N}(V, \ell_2, \varepsilon) + \varepsilon\sqrt{n}
\end{aligned}$$

$$\implies \alpha(V) \lesssim \inf_{\varepsilon > 0} \left\{ \lg^{1/2} \mathcal{N}(V, \ell_2, \varepsilon) + \varepsilon\sqrt{n} \right\}$$

Suprema of stochastic processes

We have $V \subset B_{\ell_2^n}$ and want to upper bound

$$\alpha(V) := \mathbb{E} \sup_{v \in V} |\langle \sigma, v \rangle|$$

Method 2 (ε-net): V' is an ε-net of V in ℓ_2 if $\forall v \in V\ \exists v' \in V'$ such that $\|v - v'\|_2 \le \varepsilon$

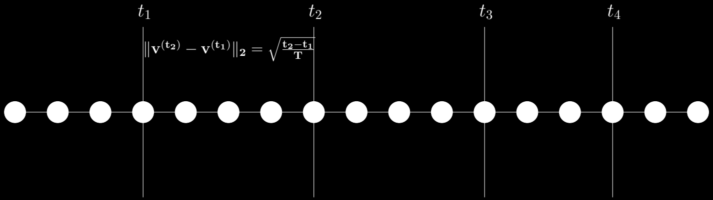
$$\begin{aligned}
\mathbb{E} \sup_{v \in V} |\langle \sigma, v \rangle| &= \mathbb{E} \sup_{v \in V} \left|\left\langle \sigma, v' + (v - v') \right\rangle\right| \\
&\le \mathbb{E} \sup_{v' \in V'} \left|\left\langle \sigma, v' \right\rangle\right| + \mathbb{E} \sup_{v \in V} \underbrace{\left|\left\langle \sigma, v - v' \right\rangle\right|}_{\le \varepsilon\sqrt{n}} \\
&\lesssim \sqrt{\lg |V'|} + \varepsilon\sqrt{n} \\
&:= \lg^{1/2} \mathcal{N}(V, \ell_2, \varepsilon) + \varepsilon\sqrt{n}
\end{aligned}$$

$$\Longrightarrow \alpha(V) \lesssim \inf_{\varepsilon > 0} \left\{ \lg^{1/2} \mathcal{N}(V, \ell_2, \varepsilon) + \varepsilon\sqrt{n} \right\}$$

For us: will show $\mathcal{N}(V, \ell_2, \varepsilon) \simeq 1/\varepsilon^2$, so $\lg^{1/2}(1/\varepsilon) + \varepsilon\sqrt{n}$

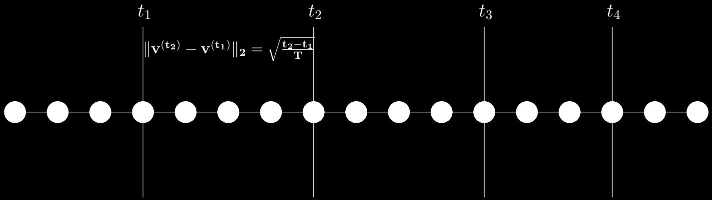
Net size for random walk on line

Recall for us: $V = \{\frac{y^{(t)}}{\sqrt{T}}\}_{t=0}^{T}$, $v^{(t)} = \frac{1}{\sqrt{T}} \cdot y^{(t)}$.



Net size for random walk on line

Recall for us: $V = \{\frac{y^{(t)}}{\sqrt{T}}\}_{t=0}^{T}$, $v^{(t)} = \frac{1}{\sqrt{T}} \cdot y^{(t)}$.



optimal ε-net is: $\{v^{(s\varepsilon^2 T)}\}$ for $s = 1, 2, \ldots, 1/\varepsilon^2$,
so $\mathcal{N}(V, \ell_2, \varepsilon) = 1/\varepsilon^2$

Suprema of stochastic processes

We have $V \subset B_{\ell_2^n}$ and want to upper bound

$$\alpha(V) := \mathbb{E} \sup_{v \in V} |\langle \sigma, v \rangle|$$

Method 3 (Dudley chaining):

Suprema of stochastic processes

We have $V \subset B_{\ell_2^n}$ and want to upper bound

$$\alpha(V) := \mathbb{E} \sup_{v \in V} |\langle \sigma, v \rangle|$$

Method 3 (Dudley chaining): Net argument: $v = v' + (v - v')$

Suprema of stochastic processes

We have $V \subset B_{\ell_2^n}$ and want to upper bound

$$\alpha(V) := \mathbb{E} \sup_{v \in V} |\langle \sigma, v \rangle|$$

Method 3 (Dudley chaining): Net argument: $v = v' + (v - v')$

This time: V_k is a 2^{-k}-net of V and $v(k)$ closest to v in V_k

Suprema of stochastic processes

We have $V \subset B_{\ell_2^n}$ and want to upper bound

$$\alpha(V) := \mathbb{E} \sup_{v \in V} |\langle \sigma, v \rangle|$$

Method 3 (Dudley chaining): Net argument: $v = v' + (v - v')$

This time: V_k is a 2^{-k}-net of V and $v(k)$ closest to v in V_k

$$v = v(0) + \sum_{k=1}^{\infty} (v(k) - v(k-1))$$

Suprema of stochastic processes

We have $V \subset B_{\ell_2^n}$ and want to upper bound

$$\alpha(V) := \mathbb{E} \sup_{v \in V} |\langle \sigma, v \rangle|$$

Method 3 (Dudley chaining): Net argument: $v = v' + (v - v')$
This time: V_k is a 2^{-k}-net of V and $v(k)$ closest to v in V_k

$$v = v(0) + \sum_{k=1}^{\infty} (v(k) - v(k-1))$$

$$\begin{aligned}
\mathbb{E} \sup_{v \in V} |\langle \sigma, v \rangle| &\le \sum_{k=1}^{\infty} \mathbb{E} \sup_{v \in V} |\langle \sigma, v(k) - v(k-1) \rangle| \\
&\lesssim \sum_{k=1}^{\infty} \sup_{v} \|v(k) - v(k-1)\|_2 \\
&\qquad \times \lg^{1/2}(\mathcal{N}(V, \ell_2, \frac{1}{2^k}) \cdot \mathcal{N}(V, \ell_2, \frac{1}{2^{k-1}})) \\
&\lesssim \sum_{k=1}^{\infty} \frac{1}{2^k} \cdot \lg^{1/2} \mathcal{N}(V, \ell_2, \frac{1}{2^k})
\end{aligned}$$

Suprema of stochastic processes

We have $V \subset B_{\ell_2^n}$ and want to upper bound

$$\alpha(V) := \mathbb{E} \sup_{v \in V} |\langle \sigma, v \rangle|$$

Method 3 (Dudley chaining): Net argument: $v = v' + (v - v')$
This time: V_k is a 2^{-k}-net of V and $v(k)$ closest to v in V_k

$$v = v(0) + \sum_{k=1}^{\infty} (v(k) - v(k-1))$$

$$\begin{aligned}
\mathbb{E} \sup_{v \in V} |\langle \sigma, v \rangle| &\le \sum_{k=1}^{\infty} \mathbb{E} \sup_{v \in V} |\langle \sigma, v(k) - v(k-1) \rangle| \\
&\lesssim \sum_{k=1}^{\infty} \sup_{v} \|v(k) - v(k-1)\|_2 \\
&\qquad \times \lg^{1/2}\left(\mathcal{N}\left(V, \ell_2, \frac{1}{2^k}\right) \cdot \mathcal{N}\left(V, \ell_2, \frac{1}{2^{k-1}}\right)\right) \\
&\lesssim \sum_{k=1}^{\infty} \frac{1}{2^k} \cdot \lg^{1/2} \mathcal{N}\left(V, \ell_2, \frac{1}{2^k}\right) \quad \left(\le \sum_k \frac{\sqrt{k}}{2^k} = O(1)\right)
\end{aligned}$$

What about the 4-wise independence?

Dudley chaining with p-wise independence

Where it all started: Khintchine inequality says $\mathbb{P}_\sigma(|\langle \sigma, v\rangle| > \lambda) \le 2e^{-\lambda^2/(2\|v\|_2^2)}$.

Dudley chaining with *p*-wise independence

Where it all started: ~~Khintchine inequality says $\mathbb{P}_\sigma(|\langle\sigma, v\rangle| > \lambda) \le 2e^{-\lambda^2/(2\|v\|_2^2)}$.~~

Khintchine says $\mathbb{E}|\langle\sigma, v\rangle|^p \le (\sqrt{p}\cdot\|v\|_2)^p$ for all $p \ge 1$

so by Markov, $\mathbb{P}(|\langle\sigma, v\rangle| > \lambda) \le (\frac{\sqrt{p}\cdot\|v\|_2}{\lambda})^p$

Dudley chaining with p-wise independence

Where it all started: ~~Khintchine inequality says $\mathbb{P}_\sigma(|\langle\sigma, v\rangle| > \lambda) \le 2e^{-\lambda^2/(2\|v\|_2^2)}$.~~

Khintchine says $\mathbb{E}|\langle\sigma, v\rangle|^p \le (\sqrt{p}\cdot\|v\|_2)^p$ for all $p \ge 1$

so by Markov, $\mathbb{P}(|\langle\sigma, v\rangle| > \lambda) \le (\frac{\sqrt{p}\cdot\|v\|_2}{\lambda})^p$

If use above new tail bound in Method 1 and push through the Dudley argument, and note $|\{v(k) - v(k-1) : v \in V\}| \le 2|V_k|$, obtain a new "Dudley-esque" bound for our V:

Dudley chaining with p-wise independence

Where it all started: ~~Khintchine inequality says $\mathbb{P}_\sigma(|\langle\sigma, v\rangle| > \lambda) \le 2e^{-\lambda^2/(2\|v\|_2^2)}$.~~

Khintchine says $\mathbb{E}|\langle\sigma, v\rangle|^p \le (\sqrt{p}\cdot\|v\|_2)^p$ for all $p \ge 1$

so by Markov, $\mathbb{P}(|\langle\sigma, v\rangle| > \lambda) \le (\frac{\sqrt{p}\cdot\|v\|_2}{\lambda})^p$

If use above new tail bound in Method 1 and push through the Dudley argument, and note $|\{v(k) - v(k-1) : v \in V\}| \le 2|V_k|$, obtain a new "Dudley-esque" bound for our V:

$$\begin{aligned}\alpha(V) &\lesssim \sum_{k=1}^{\infty} \frac{1}{2^k}\cdot\sqrt{p}\cdot(\mathcal{N}(V, \ell_2, \frac{1}{2^k}))^{1/p}\\ &\le \sum_{k=1}^{\infty}\sqrt{p}\cdot\frac{2^{2k/p}}{2^k}\\ &\lesssim 1 \text{ (for } p \ge 3)\end{aligned}$$

Yay – done with the warmup!

Recap: what we showed (and what's left)

Core lemma: If $0 = y^{(0)}, \dots, y^{(T)}$ is the evolution of a vector updated in an insertion-only stream and $\sigma \in \{-1, 1\}^n$ has 4-wise independent entries, then

$$\mathop{\mathbb{E}}_{\sigma} \sup_{t \in [T]} \left| \left\langle \sigma, v^{(t)} \right\rangle \right| \lesssim \|v^{(T)}\|_2 \quad (\text{where } v^{(t)} := \frac{y^{(t)}}{\|y^{(T)}\|_2})$$

Recap: what we showed (and what's left)

Core lemma: If $0 = y^{(0)}, \ldots, y^{(T)}$ is the evolution of a vector updated in an insertion-only stream and $\sigma \in \{-1, 1\}^n$ has 4-wise independent entries, then

$$\mathop{\mathbb{E}}_{\sigma} \sup_{t \in [T]} \left| \left\langle \sigma, v^{(t)} \right\rangle \right| \lesssim \|v^{(T)}\|_2 \quad \text{(where } v^{(t)} := \frac{y^{(t)}}{\|y^{(T)}\|_2}\text{)}$$

We showed: we proved core lemma in special case

$$v^{(t)} = \frac{1}{\sqrt{T}} \cdot (\overbrace{1, \ldots, 1}^{t}, \overbrace{0, 0, 0, 0, 0, 0, \ldots, 0}^{n-t})$$

Recap: what we showed (and what's left)

Core lemma: If $0 = y^{(0)}, \ldots, y^{(T)}$ is the evolution of a vector updated in an insertion-only stream and $\sigma \in \{-1,1\}^n$ has 4-wise independent entries, then

$$\mathop{\mathbb{E}}_{\sigma} \sup_{t \in [T]} \left| \left\langle \sigma, v^{(t)} \right\rangle \right| \lesssim \|v^{(T)}\|_2 \quad \left(\text{where } v^{(t)} := \frac{y^{(t)}}{\|y^{(T)}\|_2}\right)$$

We showed: we proved core lemma in special case

$$v^{(t)} = \frac{1}{\sqrt{T}} \cdot (\overbrace{1, \ldots, 1}^{t}, \overbrace{0, 0, 0, 0, 0, 0, \ldots, 0}^{n-t})$$

Missing to show general case? Need to bound $\mathcal{N}(V, \ell_2, \varepsilon)$ (and show $|\{v(k) - v(k-1) : v \in V\}| \leq 2|V_k|$)

Recap: what we showed (and what's left)

Core lemma: If $0 = y^{(0)}, \dots, y^{(T)}$ is the evolution of a vector updated in an insertion-only stream and $\sigma \in \{-1,1\}^n$ has 4-wise independent entries, then

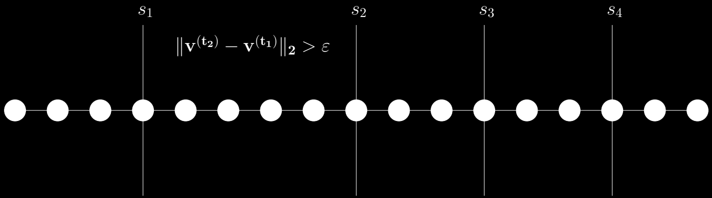
$$\mathop{\mathbb{E}}_{\sigma} \sup_{t \in [T]} \left| \left\langle \sigma, v^{(t)} \right\rangle \right| \lesssim \|v^{(T)}\|_2 \quad (\text{where } v^{(t)} := \frac{y^{(t)}}{\|y^{(T)}\|_2})$$

We showed: we proved core lemma in special case

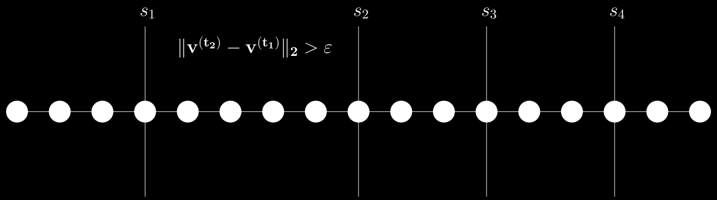
$$v^{(t)} = \frac{1}{\sqrt{T}} \cdot (\overbrace{1, \dots, 1}^{t}, \overbrace{0, 0, 0, 0, 0, 0, \dots, 0}^{n-t})$$

Missing to show general case? Need to bound $\mathcal{N}(V, \ell_2, \varepsilon)$ (and show $|\{v(k) - v(k-1) : v \in V\}| \leq 2|V_k|$)

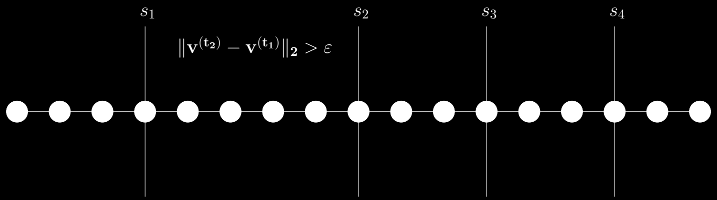
Same proof works!



- Our ε-net will be $V' = \{v^{(0)} := v^{(t_0)}, v^{(t_1)}, \ldots, v^{(t_R)}\}$
- t_j is smallest $t > t_{j-1}$ s.t. $\|v^{(t_j)} - v^{(t_{j-1})}\|_2 > \varepsilon$
- Note, again $|\{v(k) - v(k-1) : v \in V\}| \leq 2|V_k|$

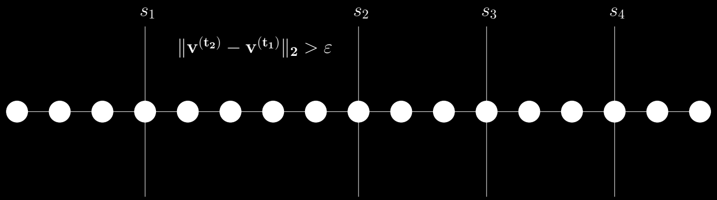


- Our ε-net will be $V' = \{v^{(0)} := v^{(t_0)}, v^{(t_1)}, \ldots, v^{(t_R)}\}$
- t_j is smallest $t > t_{j-1}$ s.t. $\|v^{(t_j)} - v^{(t_{j-1})}\|_2 > \varepsilon$
- Note, again $|\{v(k) - v(k-1) : v \in V\}| \leq 2|V_k|$
- V' is an ε-net by construction, but how big is $R := |V'| - 1$?



- Our ε-net will be $V' = \{v^{(0)} := v^{(t_0)}, v^{(t_1)}, \ldots, v^{(t_R)}\}$
- t_j is smallest $t > t_{j-1}$ s.t. $\|v^{(t_j)} - v^{(t_{j-1})}\|_2 > \varepsilon$
- Note, again $|\{v(k) - v(k-1) : v \in V\}| \leq 2|V_k|$
- V' is an ε-net by construction, but how big is $R := |V'| - 1$?

$$
\begin{aligned}
1 &\geq \|v^{(t_R)}\|_2^2 \\
&= \|\sum_{j=1}^{R} (\underbrace{v^{(t_j)} - v^{(t_{j-1})}}_{w_j})\|_2^2 \\
&\geq \sum_{j=1}^{R} \|v^{(t_j)} - v^{(t_{j-1})}\|_2^2 \quad (\text{since } \langle w_j, w_{j'} \rangle \geq 0) \\
&> R \cdot \varepsilon^2
\end{aligned}
$$



- Our ε-net will be $V' = \{v^{(0)} := v^{(t_0)}, v^{(t_1)}, \ldots, v^{(t_R)}\}$
- t_j is smallest $t > t_{j-1}$ s.t. $\|v^{(t_j)} - v^{(t_{j-1})}\|_2 > \varepsilon$
- Note, again $|\{v(k) - v(k-1) : v \in V\}| \le 2|V_k|$
- V' is an ε-net by construction, but how big is $R := |V'| - 1$?

$$
\begin{aligned}
1 &\ge \|v^{(t_R)}\|_2^2 \\
&= \|\sum_{j=1}^{R} (\underbrace{v^{(t_j)} - v^{(t_{j-1})}}_{w_j})\|_2^2 \\
&\ge \sum_{j=1}^{R} \|v^{(t_j)} - v^{(t_{j-1})}\|_2^2 \quad (\text{since } \langle w_j, w_{j'} \rangle \ge 0) \\
&> R \cdot \varepsilon^2 \quad (\Longrightarrow R < 1/\varepsilon^2)
\end{aligned}
$$

Open Problems

Open Problems

- $O(k)$ words of memory for insertion-only ℓ_2 heavy hitters?
- Does core lemma hold with 2-wise independence?