Oblivious Online Vector Balancing

Mehtaab Sawhney (MIT)
Joint with Ryan Alweiss (Princeton), Yang P. Liu (Stanford)

August 3, 2020

Six Standard Deviations Suffice

Theorem (Spencer, 1985)

Fix vectors $v_1, v_2, \ldots, v_n \in \{0,1\}^n$ *for all* $i \in [n]$. *Then exist signs* $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n \in \{-1, 1\}$ *such that* $\|\varepsilon_1 v_1 + \varepsilon_2 v_2 + \cdots + \varepsilon_t v_t\|_\infty \leq 6\sqrt{n}$.

Six Standard Deviations Suffice

Theorem (Spencer, 1985)

Fix vectors $v_1, v_2, \ldots, v_n \in \{0,1\}^n$ *for all* $i \in [n]$. *Then exist signs* $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n \in \{-1, 1\}$ *such that* $\|\varepsilon_1 v_1 + \varepsilon_2 v_2 + \cdots + \varepsilon_t v_t\|_\infty \leq 6\sqrt{n}$.

- Equivalent given n subsets of $[n]$, one can color the elements red and blue so that that ever set has discrepancy $6\sqrt{n}$.
- The original proof of Spencer was not algorithmic, but this has been made algorithmic in seminal works of Bansal (2010) and Lovett, Meka (2012).
- For further work in this direction see the particularly elegant Rothvoss (2016).

Komlós Conjecture

Conjecture

For vectors $v_1, v_2, \ldots, v_t \in \mathbb{R}^n$ with $\|v_i\|_2 \leq 1$ for all $i \in [t]$, there exist signs $\varepsilon_i \in \{-1, 1\}$ such that $\|\varepsilon_1 v_1 + \varepsilon_2 v_2 + \cdots + \varepsilon_t v_t\|_\infty \leq O(1)$.

Komlós Conjecture

Conjecture

For vectors $v_1, v_2, \ldots, v_t \in \mathbb{R}^n$ with $\|v_i\|_2 \leq 1$ for all $i \in [t]$, there exist signs $\varepsilon_i \in \{-1, 1\}$ such that $\|\varepsilon_1 v_1 + \varepsilon_2 v_2 + \cdots + \varepsilon_t v_t\|_\infty \leq O(1)$.

- The current best known bound of $O(\sqrt{\log n})$ was due to Banaszczyk (1998) – not algorithmic.
- This has been made algorithmic (polynomial time) by Bansal, Dadush, Garg (2016) and Bansal, Dadush, Garg, Lovett (2016).

Online Vector Balancing

Online Vector Balancing (Spencer 1977)

Assigns signs $\varepsilon_1, \ldots, \varepsilon_t$ to vectors $v_1, v_2, \ldots, v_t$ which arrive one at a time. The goal is to keep $\|\varepsilon_1 v_1 + \cdots + \varepsilon_i v_i\|_\infty$ as small as possible for all i. We call the quantity $\max_{1 \le i \le t} \|\varepsilon_1 v_1 + \cdots + \varepsilon_i v_i\|_\infty$ the *discrepancy*.

Examples of Online Vector Balancing

	Vector	Sign	Partial Sum
v_1			
v_2			
v_3			
v_4			
v_5			

Example Online Vector Balancing

	Vector	Sign	Partial Sum
v_1	(-1, 1, 1, -1)		
v_2			
v_3			
v_4			
v_5			

Example Online Vector Balancing

	Vector	Sign	Partial Sum
v_1	(-1, 1, 1, -1)	1	(-1, 1, 1, -1)
v_2			
v_3			
v_4			
v_5			

Example Online Vector Balancing

	Vector	Sign	Partial Sum
v_1	(-1, 1, 1, -1)	1	(-1, 1, 1, -1)
v_2	(-1, -1, -1, 1)		
v_3			
v_4			
v_5			

Example Online Vector Balancing

	Vector	Sign	Partial Sum
v_1	(-1, 1, 1, -1)	1	(-1, 1, 1, -1)
v_2	(-1, -1, -1, 1)	1	(-2, 0, 0, 0)
v_3			
v_4			
v_5			

Example Online Vector Balancing

	Vector	Sign	Partial Sum
v_1	(-1, 1, 1, -1)	1	(-1, 1, 1, -1)
v_2	(-1, -1, -1, 1)	1	(-2, 0, 0, 0)
v_3	(-1, 1, 1, 1)		
v_4			
v_5			

Example Online Vector Balancing

	Vector	Sign	Partial Sum
v_1	(-1, 1, 1, -1)	1	(-1, 1, 1, -1)
v_2	(-1, -1, -1, 1)	1	(-2, 0, 0, 0)
v_3	(-1, 1, 1, 1)	-1	(-1, -1, -1, -1)
v_4			
v_5			

Example Online Vector Balancing

	Vector	Sign	Partial Sum
v_1	(-1, 1, 1, -1)	1	(-1, 1, 1, -1)
v_2	(-1, -1, -1, 1)	1	(-2, 0, 0, 0)
v_3	(-1, 1, 1, 1)	-1	(-1, -1, -1, -1)
v_4	(-1, 1, -1, -1)		
v_5			

Example Online Vector Balancing

	Vector	Sign	Partial Sum
v_1	(-1, 1, 1, -1)	1	(-1, 1, 1, -1)
v_2	(-1, -1, -1, 1)	1	(-2, 0, 0, 0)
v_3	(-1, 1, 1, 1)	-1	(-1, -1, -1, -1)
v_4	(-1, 1, -1, -1)	-1	(0, -2, 0, 0)
v_5			

Example Online Vector Balancing

	Vector	Sign	Partial Sum
v_1	(-1, 1, 1, -1)	1	(-1, 1, 1, -1)
v_2	(-1, -1, -1, 1)	1	(-2, 0, 0, 0)
v_3	(-1, 1, 1, 1)	-1	(-1, -1, -1, -1)
v_4	(-1, 1, -1, -1)	-1	(0, -2, 0, 0)
v_5	(-1, -1, 1, 1)		

Example Online Vector Balancing

	Vector	Sign	Partial Sum
v_1	(-1, 1, 1, -1)	1	(-1, 1, 1, -1)
v_2	(-1, -1, -1, 1)	1	(-2, 0, 0, 0)
v_3	(-1, 1, 1, 1)	-1	(-1, -1, -1, -1)
v_4	(-1, 1, -1, -1)	-1	(0, -2, 0, 0)
v_5	(-1, -1, 1, 1)	-1	(1, -1, -1, -1)

Applications of Vector Balancing

- The Gram-Schmidt walk (Bansal, Dadush, Garg, Lovett 2016) has been used in studying the design of randomized controlled trials (Harshaw, Sävje, Spielman, Zhang 2019).

Applications of Vector Balancing

- The Gram-Schmidt walk (Bansal, Dadush, Garg, Lovett 2016) has been used in studying the design of randomized controlled trials (Harshaw, Sävje, Spielman, Zhang 2019).
- Can reduce several problems in online geometric discrepancy (interval discrepancy, Tusnády's problem) to online vector balancing (Jiang, Kulkarni, Singla 2019 and Bansal, Jiang, Singla, Sinha 2020).

Applications of Vector Balancing

- The Gram-Schmidt walk (Bansal, Dadush, Garg, Lovett 2016) has been used in studying the design of randomized controlled trials (Harshaw, Sävje, Spielman, Zhang 2019).
- Can reduce several problems in online geometric discrepancy (interval discrepancy, Tusnády's problem) to online vector balancing (Jiang, Kulkarni, Singla 2019 and Bansal, Jiang, Singla, Sinha 2020).
- Recent results in online discrepancy theory give algorithms for online envy minimization algorithms (Jiang, Kulkarni, Singla 2019 and Bansal, Jiang, Singla, Sinha 2020).

Applications of Vector Balancing

- The Gram-Schmidt walk (Bansal, Dadush, Garg, Lovett 2016) has been used in studying the design of randomized controlled trials (Harshaw, Sävje, Spielman, Zhang 2019).
- Can reduce several problems in online geometric discrepancy (interval discrepancy, Tusnády's problem) to online vector balancing (Jiang, Kulkarni, Singla 2019 and Bansal, Jiang, Singla, Sinha 2020).
- Recent results in online discrepancy theory give algorithms for online envy minimization algorithms (Jiang, Kulkarni, Singla 2019 and Bansal, Jiang, Singla, Sinha 2020).
- Graph balancing and the carpooling problem (Ajtai, Aspnes, Naor, Rabani, Schulman, Waarts 1998 and Gupta, Krishnaswamy, Kumar, Singla 2020).

Models of Online Vector Balancing

Models for Online Vector Balancing

- (Stochastic Model) Vectors v_i come from a fixed distribution $\mathfrak{p}$ known to the algorithm.
- (Adaptive Model) Adversary chooses vectors depending on the algorithm run, turns out to be simple as adversary can pick orthogonal vector.
- (Oblivious Model) Vectors decided beforehand, do not change based on randomness used in the algorithm.
- (Other Variations) Other models that interpolate between these such as the prophet model, where the distribution $\mathfrak{p}$ can depend on i.

Relationship Between Models

Input vectors $v_1, \cdots, v_t \in \mathbb{R}^n$ with $\|v_i\|_2 \leq 1$.

	Model	Lower Bound	Upper Bound
Most general	Adaptive	$\Omega(\sqrt{t})$	$O(\sqrt{t})$
	Oblivious	$\tilde{\Omega}(\sqrt{\log t})$	$O(\sqrt{t})$
	Prophet	$\tilde{\Omega}(\sqrt{\log t})$	$O(\sqrt{t})$
	Stochastic	$\tilde{\Omega}(\sqrt{\log t})$	$O(n^{3/2} \log t)$
Least general	Uniform	$\tilde{\Omega}(\sqrt{\log t})$	$O(\log t)$

Relationship Between Models

Input vectors $v_1, \cdots, v_t \in \mathbb{R}^n$ with $\|v_i\|_2 \leq 1$.

	Model	Lower Bound	Upper Bound
Most general	Adaptive	$\Omega(\sqrt{t})$	$O(\sqrt{t})$
	Oblivious	$\tilde{\Omega}(\sqrt{\log t})$	$O(\sqrt{t})$
	Prophet	$\tilde{\Omega}(\sqrt{\log t})$	$O(\sqrt{t})$
	Stochastic	$\tilde{\Omega}(\sqrt{\log t})$	$O(n^{3/2} \log t)$
Least general	Uniform	$\tilde{\Omega}(\sqrt{\log t})$	$O(\log t)$

- Results due to Bansal, Spencer 2019, Jiang, Kulkarni, Singla 2019, and Bansal, Jiang, Singla, Sinha 2020.

Relationship Between Models

Input vectors $v_1, \cdots, v_t \in \mathbb{R}^n$ with $\|v_i\|_2 \leq 1$.

	Model	Lower Bound	Upper Bound
Most general	Adaptive	$\Omega(\sqrt{t})$	$O(\sqrt{t})$
	Oblivious	$\tilde{\Omega}(\sqrt{\log t})$	$O(\sqrt{t})$
	Prophet	$\tilde{\Omega}(\sqrt{\log t})$	$O(\sqrt{t})$
	Stochastic	$\tilde{\Omega}(\sqrt{\log t})$	$O(n^{3/2} \log t)$
Least general	Uniform	$\tilde{\Omega}(\sqrt{\log t})$	$O(\log t)$

- Results due to Bansal, Spencer 2019, Jiang, Kulkarni, Singla 2019, and Bansal, Jiang, Singla, Sinha 2020.
- Oblivious model generalizes the standard offline vector balancing problem (e.g. the setup for the Komlós conjecture).

Relationship Between Models

Input vectors $v_1, \cdots, v_t \in \mathbb{R}^n$ with $\|v_i\|_2 \leq 1$.

	Model	Lower Bound	Upper Bound
Most general	Adaptive	$\Omega(\sqrt{t})$	$O(\sqrt{t})$
	Oblivious	$\tilde{\Omega}(\sqrt{\log t})$	$O(\log nt)$
	Prophet	$\tilde{\Omega}(\sqrt{\log t})$	$O(\log nt)$
	Stochastic	$\tilde{\Omega}(\sqrt{\log t})$	$O(\log nt)$
Least general	Uniform	$\tilde{\Omega}(\sqrt{\log t})$	$O(\log t)$

- Results due to Bansal, Spencer 2019, Jiang, Kulkarni, Singla 2019, and Bansal, Jiang, Singla, Sinha 2020.
- Oblivious model generalizes the standard offline vector balancing problem (e.g. the setup for the Komlós conjecture).
- $O(\log(nt)^4)$ results for the stochastic model were achieved (concurrently) by Bansal, Jiang, Meka, Singla, Sinha 2020.

Stochastic Vector Balancing Results

- Vectors v_i come from a fixed distribution $\mathfrak{p}$ known to the algorithm.

Stochastic Vector Balancing Results

- Vectors v_i come from a fixed distribution $\mathfrak{p}$ known to the algorithm.
- Trivial $O(\sqrt{t \log t})$ bound is achievable for even the adaptive setting by randomly assigning signs.

Stochastic Vector Balancing Results

- Vectors v_i come from a fixed distribution $\mathfrak{p}$ known to the algorithm.
- Trivial $O(\sqrt{t \log t})$ bound is achievable for even the adaptive setting by randomly assigning signs.
- For v_i uniform in $[-1, 1]^n$, can guarantee $\|\varepsilon_1 v_1 + \cdots + \varepsilon_i v_i\|_\infty = O(\sqrt{n} \log t)$ for all $i \in [t]$ (Bansal, Spencer 2019).

Stochastic Vector Balancing Results

- Vectors v_i come from a fixed distribution $\mathfrak{p}$ known to the algorithm.
- Trivial $O(\sqrt{t \log t})$ bound is achievable for even the adaptive setting by randomly assigning signs.
- For v_i uniform in $[-1,1]^n$, can guarantee $\|\varepsilon_1 v_1 + \cdots + \varepsilon_i v_i\|_\infty = O(\sqrt{n} \log t)$ for all $i \in [t]$ (Bansal, Spencer 2019).
- For arbitrary distribution $\mathfrak{p}$ on $[-1,1]^n$ and v_i iid from $\mathfrak{p}$, can can guarantee $\|\varepsilon_1 v_1 + \cdots + \varepsilon_i v_i\|_\infty = O(n^2 \log t)$ for all $i \in [t]$ (Bansal, Jiang, Singla, Sinha 2020). This was recently improved (in concurrent work) to $O(\sqrt{n} \log(nt)^4)$ by Bansal, Jiang, Meka, Singla, Sinha 2020.
- One can also achieve a bound of $O_n(\sqrt{\log t})$, however the n dependence is at least exponential (Aru, Narayanan, Scott, Venkatesen 2018).

Discussion of Methods

- Previous stochastic vector balancing algorithms pick a sign ε_i to minimize a potential function such as $\Phi(w) := \sum_{i=1}^{n} \cosh(\lambda w_i)$.

Discussion of Methods

- Previous stochastic vector balancing algorithms pick a sign ε_i to minimize a potential function such as $\Phi(w) := \sum_{i=1}^{n} \cosh(\lambda w_i)$.
- Does not work for oblivious setting where v_i are arbitrary vectors with $\|v_i\|_2 \leq 1$, as potential minimizing algorithms are deterministic.

Discussion of Methods

- Previous stochastic vector balancing algorithms pick a sign ε_i to minimize a potential function such as $\Phi(w) := \sum_{i=1}^{n} \cosh(\lambda w_i)$.
- Does not work for oblivious setting where v_i are arbitrary vectors with $\|v_i\|_2 \leq 1$, as potential minimizing algorithms are deterministic.
- Any deterministic algorithm must have discrepancy $\Omega(\sqrt{t})$ as the adversary can make the next vector orthogonal to current position.

Discussion of Methods

- Previous stochastic vector balancing algorithms pick a sign ε_i to minimize a potential function such as $\Phi(w) := \sum_{i=1}^{n} \cosh(\lambda w_i)$.
- Does not work for oblivious setting where v_i are arbitrary vectors with $\|v_i\|_2 \leq 1$, as potential minimizing algorithms are deterministic.
- Any deterministic algorithm must have discrepancy $\Omega(\sqrt{t})$ as the adversary can make the next vector orthogonal to current position.
- Best known oblivious bound is $O(\sqrt{t})$. (Simply assign signs in a greedy manner.)

Intuition for Algorithm

Desired Properties

- Algorithm must be randomized to avoid the $\Omega(\sqrt{t})$ lower bound.

Intuition for Algorithm

Desired Properties

- Algorithm must be randomized to avoid the $\Omega(\sqrt{t})$ lower bound.
- Previous analyses of Komlós are based on subgaussianity, so the algorithm should be rotation invariant.

Intuition for Algorithm

Desired Properties

- Algorithm must be randomized to avoid the $\Omega(\sqrt{t})$ lower bound.
- Previous analyses of Komlós are based on subgaussianity, so the algorithm should be rotation invariant.
- Algorithm depends on current partial sum and next input vector v only.

Intuition for Algorithm

Desired Properties

- Algorithm must be randomized to avoid the $\Omega(\sqrt{t})$ lower bound.
- Previous analyses of Komlós are based on subgaussianity, so the algorithm should be rotation invariant.
- Algorithm depends on current partial sum and next input vector v only.

If w is the current partial sum, and v is next vector, the probability that the sign of v will be -1 or $+1$ respectively will depend only on the inner product $\langle v, w \rangle$.

Self-Balancing Walk

Algorithm: $\textsc{Balance}(v_1, \cdots, v_t, \delta)$

$w_0 \leftarrow 0$.
$c \leftarrow 30 \log(nt/\delta)$.
for $1 \leq i \leq t$ **do**
  if $|\langle w_{i-1}, v_i \rangle| > c$ *or* $\|w_{i-1}\|_\infty > c$ **then**
    Fail. Algorithm terminates with failure.
  $p_i \leftarrow \frac{1}{2} - \frac{\langle w_{i-1}, v_i \rangle}{2c}$.
  $\varepsilon_i \leftarrow 1$ with probability p_i, and $\varepsilon_i \leftarrow -1$ with probability $1 - p_i$.
  $w_i \leftarrow w_{i-1} + \varepsilon_i v_i$.

Self-Balancing Walk

Algorithm: $\textsc{Balance}(v_1, \cdots, v_t, \delta)$

$w_0 \leftarrow 0$.
$c \leftarrow 30 \log(nt/\delta)$.
for $1 \le i \le t$ **do**
  if $|\langle w_{i-1}, v_i \rangle| > c$ *or* $\|w_{i-1}\|_\infty > c$ **then**
    Fail. Algorithm terminates with failure.
  $p_i \leftarrow \frac{1}{2} - \frac{\langle w_{i-1}, v_i \rangle}{2c}$.
  $\varepsilon_i \leftarrow 1$ with probability p_i, and $\varepsilon_i \leftarrow -1$ with probability $1 - p_i$.
  $w_i \leftarrow w_{i-1} + \varepsilon_i v_i$.

- Maintain the current partial sum w_i.

Self-Balancing Walk

Algorithm: $\textsc{Balance}(v_1, \cdots, v_t, \delta)$

$w_0 \leftarrow 0$.
$c \leftarrow 30\log(nt/\delta)$.
for $1 \le i \le t$ **do**
  if $|\langle w_{i-1}, v_i\rangle| > c$ *or* $\|w_{i-1}\|_\infty > c$ **then**
    Fail. Algorithm terminates with failure.
  $p_i \leftarrow \frac{1}{2} - \frac{\langle w_{i-1}, v_i\rangle}{2c}$.
  $\varepsilon_i \leftarrow 1$ with probability p_i, and $\varepsilon_i \leftarrow -1$ with probability $1 - p_i$.
  $w_i \leftarrow w_{i-1} + \varepsilon_i v_i$.

- Maintain the current partial sum w_i.
- The sign ε_i is determined by a random Bernoulli with probability $\frac{1}{2} - \frac{\langle w_{i-1}, v_i\rangle}{2c}$.

Self-Balancing Walk

Algorithm: $\textsc{Balance}(v_1, \cdots, v_t, \delta)$

$w_0 \leftarrow 0$.
$c \leftarrow 30 \log(nt/\delta)$.
for $1 \le i \le t$ **do**
  if $|\langle w_{i-1}, v_i \rangle| > c$ *or* $\|w_{i-1}\|_\infty > c$ **then**
    Fail. Algorithm terminates with failure.
  $p_i \leftarrow \frac{1}{2} - \frac{\langle w_{i-1}, v_i \rangle}{2c}$.
  $\varepsilon_i \leftarrow 1$ with probability p_i, and $\varepsilon_i \leftarrow -1$ with probability $1 - p_i$.
  $w_i \leftarrow w_{i-1} + \varepsilon_i v_i$.

- Maintain the current partial sum w_i.
- The sign ε_i is determined by a random Bernoulli with probability $\frac{1}{2} - \frac{\langle w_{i-1}, v_i \rangle}{2c}$.
- Algorithm must fail if $|\langle w_{i-1}, v_i \rangle| > c$, or else probability isn't in $[0, 1]$!

Self-Balancing Walk Theorem

Algorithm: $\textsc{Balance}(v_1, \cdots, v_t, \delta)$

$w_0 \leftarrow 0$.
$c \leftarrow 30\log(nt/\delta)$.
for $1 \le i \le t$ **do**
  if $|\langle w_{i-1}, v_i\rangle| > c$ *or* $\|w_{i-1}\|_\infty > c$ **then**
    Fail. Algorithm terminates with failure.
  $p_i \leftarrow \frac{1}{2} - \frac{\langle w_{i-1}, v_i\rangle}{2c}$.
  $\varepsilon_i \leftarrow 1$ with probability p_i, and $\varepsilon_i \leftarrow -1$ with probability $1 - p_i$.
  $w_i \leftarrow w_{i-1} + \varepsilon_i v_i$.

Self-Balancing Walk Theorem

Algorithm: $\textsc{Balance}(v_1, \cdots, v_t, \delta)$

$w_0 \leftarrow 0$.
$c \leftarrow 30 \log(nt/\delta)$.
for $1 \leq i \leq t$ **do**
  if $|\langle w_{i-1}, v_i \rangle| > c$ *or* $\|w_{i-1}\|_\infty > c$ **then**
    Fail. Algorithm terminates with failure.
  $p_i \leftarrow \frac{1}{2} - \frac{\langle w_{i-1}, v_i \rangle}{2c}$.
  $\varepsilon_i \leftarrow 1$ with probability p_i, and $\varepsilon_i \leftarrow -1$ with probability $1 - p_i$.
  $w_i \leftarrow w_{i-1} + \varepsilon_i v_i$.

Theorem (Alweiss, Liu, S. 2020)

For any vectors $v_1, v_2, \cdots, v_t \in \mathbb{R}^n$ with $\|v_i\|_2 \leq 1$ for all $i \in [t]$, algorithm $\textsc{Balance}(v_1, \cdots, v_t, \delta)$ maintains $\|w_i\|_\infty = O(\log(nt/\delta))$ for all $i \in [t]$ with probability $1 - \delta$.

Corollaries of Main Theorem

Corollary

Given a matrix $A \in \mathbb{R}^{n\times t}$ with columns with ℓ_2-norm at most 1, we can find with high probability in $O(\mathrm{nnz}(A))$ time a vector $x \in \{-1,1\}^t$ such that $\|Ax\|_\infty = O(\sqrt{\log t \cdot \log n})$.

Makes progress towards giving input sparsity / linear time algorithms for discrepancy problems.

Corollaries of Main Theorem

Corollary

Given a matrix $A \in \mathbb{R}^{n\times t}$ with columns with ℓ_2-norm at most 1, we can find with high probability in $O(\mathrm{nnz}(A))$ time a vector $x \in \{-1,1\}^t$ such that $\|Ax\|_\infty = O(\sqrt{\log t \cdot \log n})$.

Makes progress towards giving input sparsity / linear time algorithms for discrepancy problems.

- Obtain improvements to several online geometric discrepancy problems (online Tusnády's problem, online interval discrepancy).
- "Nearly" match best known *offline* bounds for Tusnády's problem.

Steps in the Analysis

- Let $w_i = \varepsilon_1 v_1 + \ldots + \varepsilon_i v_i$.
- We consider the distribution of w_i and how it evolves at every step.

Steps in the Analysis

- Let $w_i = \varepsilon_1 v_1 + \ldots + \varepsilon_i v_i$.
- We consider the distribution of w_i and how it evolves at every step.
- Decompose each step into "a shift in expectation" and "variance between w_i and w_{i+1}"

Steps in the Analysis

- Let $w_i = \varepsilon_1 v_1 + \ldots + \varepsilon_i v_i$.
- We consider the distribution of w_i and how it evolves at every step.
- Decompose each step into "a shift in expectation" and "variance between w_i and w_{i+1}"
- Using this decomposition achieve the necessary concentration.

Difficulties in Achieving Concentration

- We wish to achieve high probability bounds on the position of w_i.

Difficulties in Achieving Concentration

- We wish to achieve high probability bounds on the position of w_i.
- Does not seem doable with classical martingale concentration inequalities.
- In particular, v_i may be orthogonal to w_i, which prevents us from arguing that some potential is decreasing pointwise, which is necessary for Azuma's inequality, etc.

Difficulties in Achieving Concentration

- We wish to achieve high probability bounds on the position of w_i.
- Does not seem doable with classical martingale concentration inequalities.
- In particular, v_i may be orthogonal to w_i, which prevents us from arguing that some potential is decreasing pointwise, which is necessary for Azuma's inequality, etc.
- Therefore, we require more global control over the distribution of w_i.

Spreading

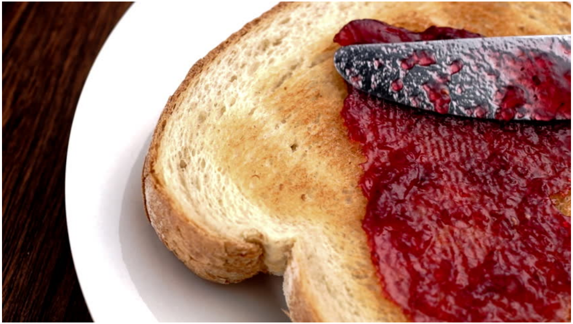

Figure 1: Spreading jam on a piece of bread

(Actual) Spreading

The key idea in our analysis is the following definition.

Definition

We say that random variables Y on $\mathbb{R}^n$ is a spread of random variable X on $\mathbb{R}^n$ if there exists a coupling of X and Y such that $\mathbb{E}[Y|X] = X$.

(Actual) Spreading

The key idea in our analysis is the following definition.

Definition

We say that random variables Y on $\mathbb{R}^n$ is a spread of random variable X on $\mathbb{R}^n$ if there exists a coupling of X and Y such that $\mathbb{E}[Y|X] = X$.

- A more intuitive definition is that Y can be sampled by first sampling X, and then adding mean 0 noise (conditional on X).
- The univariate notion of the definition above appears in mathematical economics literature under the names "mean-preserving spread" and is closely related to "second-order stochastic dominace".

Properties of Spreading

Spreading satisfies several useful and intuitive properties.

Properties of Spreading

Spreading satisfies several useful and intuitive properties.

- (Linearity) If Y is a spread of X, then for any linear transformation M on $\mathbb{R}^n$ we have that MY is a spread of MX.

Properties of Spreading

Spreading satisfies several useful and intuitive properties.

- (Linearity) If Y is a spread of X, then for any linear transformation M on $\mathbb{R}^n$ we have that MY is a spread of MX.
- (Transitivity) If Z is a spread of Y and Y is a spread of X, then Z is a spread of X.

Properties of Spreading

Spreading satisfies several useful and intuitive properties.

- (Linearity) If Y is a spread of X, then for any linear transformation M on $\mathbb{R}^n$ we have that MY is a spread of MX.
- (Transitivity) If Z is a spread of Y and Y is a spread of X, then Z is a spread of X.
- (Convexity) Let distribution Y be a spread of X. For any convex function $\Phi : \mathbb{R}^n \to \mathbb{R}$, we have that $\mathbb{E}_{x \sim X}\Phi(x) \leq \mathbb{E}_{y \sim Y}\Phi(y)$.

Properties of Spreading

Spreading satisfies several useful and intuitive properties.

- (Linearity) If Y is a spread of X, then for any linear transformation M on $\mathbb{R}^n$ we have that MY is a spread of MX.
- (Transitivity) If Z is a spread of Y and Y is a spread of X, then Z is a spread of X.
- (Convexity) Let distribution Y be a spread of X. For any convex function $\Phi : \mathbb{R}^n \to \mathbb{R}$, we have that $\mathbb{E}_{x \sim X}\Phi(x) \leq \mathbb{E}_{y \sim Y}\Phi(y)$.
- (Spreading real variables by Gaussians) Let X be a real-valued random variable with $\mathbb{E}[X] = 0$ and $|X| \leq C$. Then $G = \mathcal{N}(0, \pi C^2/2)$ is a spread of X.
- If PSD matrices A, B satisfy $A \preceq B$ then $\mathcal{N}(0, A)$ is spread by $\mathcal{N}(0, B)$.

Example of Spreading

Spreading of Gaussians

If PSD matrices A, B satisfy $A \preceq B$ then $\mathcal{N}(0, A)$ is spread by $\mathcal{N}(0, B)$.

Example of Spreading

Spreading of Gaussians

If PSD matrices A, B satisfy $A \preceq B$ then $\mathcal{N}(0, A)$ is spread by $\mathcal{N}(0, B)$.

Proof.

Note that $\mathcal{N}(0, B) = \mathcal{N}(0, A) + \mathcal{N}(0, B - A)$ and $B - A \succeq 0$. $\square$

Example of Spreading

Spreading of Gaussians

If PSD matrices A, B satisfy $A \preceq B$ then $\mathcal{N}(0, A)$ is spread by $\mathcal{N}(0, B)$.

Proof.

Note that $\mathcal{N}(0, B) = \mathcal{N}(0, A) + \mathcal{N}(0, B - A)$ and $B - A \succeq 0$. $\square$

- In particular, $\mathbb{E}_{x \sim \mathcal{N}(0,B)}$ can be sampled by first sampling $\mathbb{E}_{x \sim \mathcal{N}(0,A)}$, and then adding the mean-zero random variable $\mathcal{N}(0, B - A)$.

Example of Spreading

Spreading of Gaussians

If PSD matrices A, B satisfy $A \preceq B$ then $\mathcal{N}(0, A)$ is spread by $\mathcal{N}(0, B)$.

Proof.

Note that $\mathcal{N}(0, B) = \mathcal{N}(0, A) + \mathcal{N}(0, B - A)$ and $B - A \succeq 0$. $\square$

- In particular, $\mathbb{E}_{x\sim\mathcal{N}(0,B)}$ can be sampled by first sampling $\mathbb{E}_{x\sim\mathcal{N}(0,A)}$, and then adding the mean-zero random variable $\mathcal{N}(0, B - A)$.
- For general spreading, the mean-zero random variable can differ between samples.

Distributions under Consideration

- Let w_i be the position of vector at time i. Note that this is not precisely defined when the algorithm has failed, we will remedy this momentarily.
- If the algorithm has not failed at step i, one can verify that

$$\begin{aligned}\mathbb{E}[w_{i+1}|w_i] &= w_i + (2p_{i+1} - 1)v_{i+1}\\ &= w_i - c^{-1}v_{i+1}v_{i+1}^T w_i\\ &= (I - c^{-1}v_{i+1}v_{i+1}^T)w_i.\end{aligned}$$

- Given this we can write

$$\begin{aligned}w_{i+1} &= \mathbb{E}[w_{i+1}|w_i] + R(v_{i+1}, w_i)v_{i+1}\\ &= (I - c^{-1}v_{i+1}v_{i+1}^T)w_i + R(v_{i+1}, w_i)v_{i+1}\end{aligned}$$

Increment Step

- We see by direct computation that

$$R(w_i, v_{i+1}) = \begin{cases} 0 & \text{if algorithm has failed} \\ 1 + c^{-1}\langle w_i, v_{i+1}\rangle & \text{with prob. } p_{i+1} \text{ otherwise} \\ -1 + c^{-1}\langle w_i, v_{i+1}\rangle & \text{with prob. } 1 - p_{i+1} \text{ otherwise} \end{cases}$$

- Thus we now have a well defined notion for the position of vector w_i even if we the algorithm has failed before this step.
- The key thing to note is that if the algorithm has not failed then w_i is by construction the position of the current signed sum.

Properties of Increment Steps

- Recall

$$R(w_i, v_{i+1}) = \begin{cases} 0 & \text{if algorithm has failed} \\ 1 + c^{-1}\langle w_i, v_{i+1}\rangle & \text{with prob. } p_{i+1} \text{ otherwise} \\ -1 + c^{-1}\langle w_i, v_{i+1}\rangle & \text{with prob. } 1 - p_{i+1} \text{ otherwise} \end{cases}$$

- Note that by construction

$$\mathbb{E}[R(w_i, v_{i+1})|w_i] = 0.$$

- Furthermore note that

$$|R(w_i, v_{i+1})| \leq 2$$

by construction.

Deduction Given Spreading

Recall $c = O(\log nt/\delta)$.

Lemma

$\mathcal{N}(0, 2\pi cI)$ is a spread of the distribution of w_i for all times $i \in [t]$.

Deduction Given Spreading

Recall $c = O(\log nt/\delta)$.

Lemma

$\mathcal{N}(0, 2\pi c I)$ is a spread of the distribution of w_i for all times $i \in [t]$.

Convexity

Let distribution Y be a spread of X. For any convex function $\Phi : \mathbb{R}^n \to \mathbb{R}$, we have that $\mathbb{E}_{x\sim X}\Phi(x) \leq \mathbb{E}_{y\sim Y}\Phi(y)$.

Deduction Given Spreading

Recall $c = O(\log nt/\delta)$.

Lemma

$\mathcal{N}(0, 2\pi c I)$ is a spread of the distribution of w_i for all times $i \in [t]$.

Convexity

Let distribution Y be a spread of X. For any convex function $\Phi : \mathbb{R}^n \to \mathbb{R}$, we have that $\mathbb{E}_{x\sim X}\Phi(x) \leq \mathbb{E}_{y\sim Y}\Phi(y)$.

Proof.

We need to verify that the algorithm does not fail whp; in particular we need to verify $|\langle w_i, v_{i+1}\rangle| \leq c$ and that $\|w_i\|_\infty \leq c$. The lemma above implies that w_i is $O(c)$-subgaussian, so by the convexity property with $\Phi(x) := \exp(\langle x, u\rangle^2/8\pi c)$ we get that at most $O(\delta/t)$-fraction fraction of samples fail at each stage. Summing over the t-steps gives the desired result. $\square$

Proof of Main Lemma

Lemma

$\mathcal{N}(0, 2\pi c I)$ is a spread of the distribution of w_i for all times $i \in [t]$.

Proof of Main Lemma

Lemma

$\mathcal{N}(0, 2\pi c I)$ is a spread of the distribution of w_i for all times $i \in [t]$.

Proof.

Recall the random variables

$$R(w_i, v_{i+1}) = \begin{cases} 0 & \text{if algorithm has failed} \\ 1 + c^{-1}\langle w_i, v_{i+1}\rangle & \text{with prob. } p_{i+1} \text{ otherwise} \\ -1 + c^{-1}\langle w_i, v_{i+1}\rangle & \text{with prob. } 1 - p_{i+1} \text{ otherwise} \end{cases}$$

so that

$$\begin{aligned} w_{i+1} &= \mathbb{E}[w_{i+1} | w_i] + R(w_i, v_{i+1}) v_{i+1} \\ &= (1 - c^{-1} v_{i+1} v_{i+1}^T) w_i + R(w_i, v_{i+1}) v_{i+1}. \end{aligned}$$

Proof of Main Lemma (Continued)

Lemma

$\mathcal{N}(0, 2\pi c I)$ is a spread of the distribution of w_i for all times $i \in [t]$.

Proof of Main Lemma (Continued)

Lemma

$\mathcal{N}(0, 2\pi cI)$ is a spread of the distribution of w_i for all times $i \in [t]$.

Spreading real variables by Gaussians

If $\mathbb{E}[X] = 0$ and $|X| \leq C$, then $G = \mathcal{N}(0, \pi C^2/2)$ is a spread of X.

Proof of Main Lemma (Continued)

Lemma

$\mathcal{N}(0, 2\pi cI)$ is a spread of the distribution of w_i for all times $i \in [t]$.

Spreading real variables by Gaussians

If $\mathbb{E}[X] = 0$ and $|X| \leq C$, then $G = \mathcal{N}(0, \pi C^2/2)$ is a spread of X.

Proof.

Recall that $R(w_i, v_{i+1})$ is mean 0, and always supported on $[-2, 2]$, so it is spread by $\mathcal{N}(0, 2\pi)$ even given w_i. Furthermore by induction and linearity we have that

$$\mathcal{N}(0, (I - c^{-1}v_{i+1}v_{i+1}^T)2\pi cI(I - c^{-1}v_{i+1}v_{i+1}^T))$$

is a spread of $\mathbb{E}[w_{i+1}|w_i] = (I - c^{-1}v_{i+1}v_{i+1}^T)w_i$.

Proof of Main Lemma (Continued)

Lemma

$\mathcal{N}(0, 2\pi c I)$ is a spread of the distribution of w_i for all times $i \in [t]$.

Proof of Main Lemma (Continued)

Lemma

$\mathcal{N}(0, 2\pi cI)$ is a spread of the distribution of w_i for all times $i \in [t]$.

Proof.

Therefore $w_{i+1} = (I - c^{-1} v_i v_i^T) w_i + R(w_i, v_{i+1}) v_{i+1}$ is spread by

$$\mathcal{N}(0, (I - c^{-1} v_{i+1} v_{i+1}^T) 2\pi cI (I - c^{-1} v_{i+1} v_{i+1}^T) + 2\pi v_{i+1} v_{i+1}^T).$$

To finish simply note that

$$(I - c^{-1} v_i v_i^T) 2\pi cI (I - c^{-1} v_i v_i^T) + 2\pi v_i v_i^T \preceq 2\pi cI.$$

□

Conclusion and Open Problems

- Algorithm achieving logarithmic bounds for the online Komlós problem against oblivious adversaries.

Conclusion and Open Problems

- Algorithm achieving logarithmic bounds for the online Komlós problem against oblivious adversaries.
- Based on randomly choosing signs ε_i with probability as a linear function of the inner product of the current partial sum w_i and next input vector v_i.

Conclusion and Open Problems

- Algorithm achieving logarithmic bounds for the online Komlós problem against oblivious adversaries.
- Based on randomly choosing signs ε_i with probability as a linear function of the inner product of the current partial sum w_i and next input vector v_i.
- Analysis is based on the concept of spreading and bounds on the covariance.

Conclusion and Open Problems

- Algorithm achieving logarithmic bounds for the online Komlós problem against oblivious adversaries.
- Based on randomly choosing signs ε_i with probability as a linear function of the inner product of the current partial sum w_i and next input vector v_i.
- Analysis is based on the concept of spreading and bounds on the covariance.

Open Question

Can a similar algorithm achieve a $O(\sqrt{\log nt})$ bound?

The End

- The paper is available at https://arxiv.org/abs/2006.14009 with title "Discrepancy Minimization via a Self-Balancing Walk"

The End

- The paper is available at https://arxiv.org/abs/2006.14009 with title "Discrepancy Minimization via a Self-Balancing Walk"
- Thank you for you attention!

The End

- The paper is available at https://arxiv.org/abs/2006.14009 with title "Discrepancy Minimization via a Self-Balancing Walk"
- Thank you for you attention!
- Questions?