Dispersion for Data-Driven Algorithm Design, Online Learning, and Private Optimization

Ellen Vitercik

Joint work with Nina Balcan and Travis Dick. Appeared in FOCS 2018.

Data-driven algorithm design

Some problems have optimal, fast algorithms.
Some don't:

- Many methods – which is best for our domain?
- E.g., clustering and subset selection problems

Use ML to automate algorithm design!

This work: **formal guarantees for this approach**

Learning setup

- Fix large family of parameterized algorithms
  - E.g., knapsack: until knapsack full, add items in decreasing order of $\frac{(\text{value})}{(\text{size})^{\rho}}$

- Learner sees stream of T problem instances
- At timestep t, choose parameters ρ_t

$\rho_1 = 1$ $\quad$ $\rho_2 = \frac{1}{4}$ $\quad$ $\rho_3 = \frac{3}{4}$

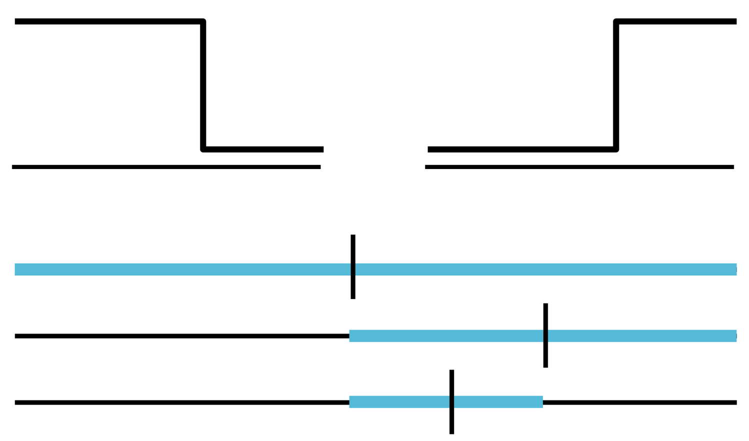
- Want to minimize **regret**:
  - Difference between cumulative performance of those parameters and optimal parameters in hindsight:

$$\max_{\rho} \sum_{t=1}^{T} u_t(\rho) - \sum_{t=1}^{T} u_t(\rho_t),$$

where $u_t(\rho)$ measures performance of algorithm parameterized by ρ on t^{th} problem

Main challenge

- Algorithm's performance on an instance as function of parameters is often piecewise Lipschitz
- In general, optimizing piecewise Lipschitz functions is impossible!



Approach

Exponentially-weighted forecaster (EWF): On round t, choose parameters ρ w.p. $\propto \exp(\lambda \sum_{s=1}^{t-1} u_s(\rho))$

Dispersion

$\{u_1, \ldots, u_T: \mathbb{R}^d \to [0,1]\}$ is **(w, k)-dispersed at ρ** if ℓ_2-ball $B(\rho, w)$ contains discontinuities for $\leq k$ functions

Lemma: If $u_1, \ldots, u_T$ are piecewise L-Lipschitz and (w, k)-dispersed at a maximizer ρ^*, for every $\rho \in B(\rho^*, w)$,

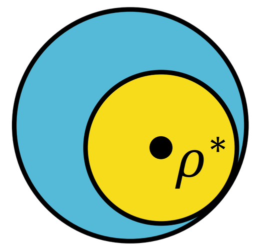
$$\sum_{t=1}^{T} u_t(\rho) \geq OPT - TLw - k.$$

Proof: $u_1, \ldots, u_T$

Is u_t L-Lipschitz on $B(\rho^*, w)$?

No ($\leq k$ functions) → $|u_t(\rho) - u_t(\rho^*)| \leq 1$

Yes ($\leq T$ functions) → $|u_t(\rho) - u_t(\rho^*)| \leq Lw$



Our guarantees

Upper bound: If $u_1, \ldots, u_T$ are piecewise L-Lipschitz and (w, k)-dispersed at ρ^*, EWF has regret

$$O\left(\sqrt{Td \log \frac{1}{w}} + TLw + k\right).$$

When is this a good bound? For $w = \frac{1}{L\sqrt{T}}$ and $k = \tilde{O}(\sqrt{T})$ regret is $\tilde{O}(\sqrt{Td})$.

Matching lower bound: For any algorithm A, there are piecewise constant functions $u_1, \ldots, u_T$ so that A has expected regret

$$\Omega\left(\inf_{(w,k)} \sqrt{Td \log \frac{1}{w}} + k\right).$$

Infimum is over all (w, k)-dispersion parameters satisfied by $\{u_1, \ldots, u_T\}$ at ρ^*.

Applications

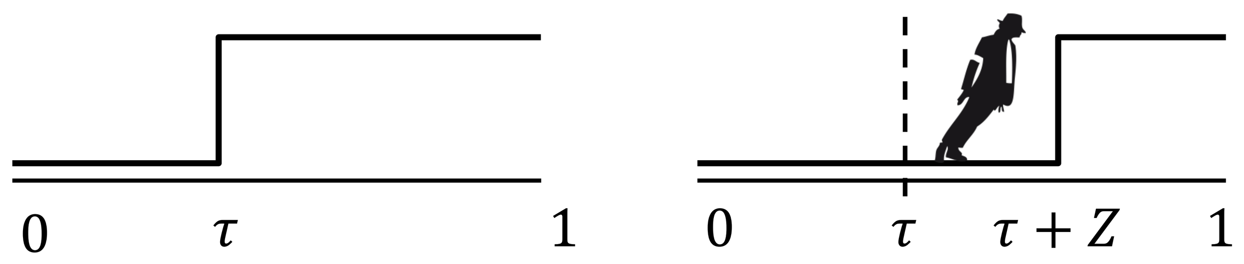
Prove dispersion for:

- Pricing and auction design
- Algorithm configuration for subset selection problems (e.g., knapsack and maximum weight independent set) and integer quadratic programming

Dispersion also implies **differentially private optimization** guarantees (tight lower bounds!)

Smooth adversaries

Adversary chooses thresholds $u_t: [0,1] \to \{0,1\}$



Discontinuity τ "smoothed" by adding $Z \sim N(0, \sigma^2)$.

Lemma: W.h.p., for any $w > 0$, $\{u_1, \ldots, u_T\}$ is (w, k)-dispersed for $k = \tilde{O}\left(\frac{Tw}{\sigma} + \sqrt{T}\right)$.

Proof: Density of $\tau + Z$ is $O\left(\frac{1}{\sigma}\right)$.

- For any width-w interval, expected number discontinuities $= O\left(\frac{Tw}{\sigma}\right)$.
- Intervals have VC-dim 2 → W.h.p., every interval contains $\tilde{O}\left(\frac{Tw}{\sigma} + \sqrt{T}\right)$ discontinuities.

$\boxed{w = \frac{\sigma}{\sqrt{T}} \to \text{Regret} = O\left(\sqrt{T \log \frac{T}{\sigma}}\right)}$

My related work

- Data-driven algorithm design

1. Alabi, Kalai, Ligett, Musco, Tzamos, and V. Speeding-up repeated computations via pruning. 2018.
2. Balcan, Dick, Sandholm, and V. Learning to branch. ICML 2018.
3. Balcan, Nagarajan, V., and White. Learning-theoretic foundations of algorithm configuration for combinatorial partitioning problems. COLT 2017.

- ML and economics

1. Balcan, Sandholm, and V. Sample complexity of automated mechanism design. NIPS 2016.
2. Balcan, Sandholm, and V. A general theory of sample complexity for multi-item profit maximization. EC 2018.
3. Balcan, V, and White. Learning combinatorial functions from pairwise comparisons. COLT 2016.