Top-K Ranking with a Monotone Adversary

Cong Ma
Department of Statistics, UChicago

School of Statistics Seminar, University of Minnesota, Feb. 2024

Yuepeng Yang
UChicago Stat

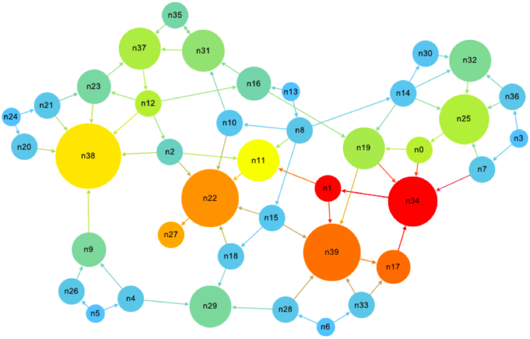
Antares Chen
UChicago CS

Lorenzo Orecchia
UChicago CS

Ranking

A fundamental problem in a wide range of contexts

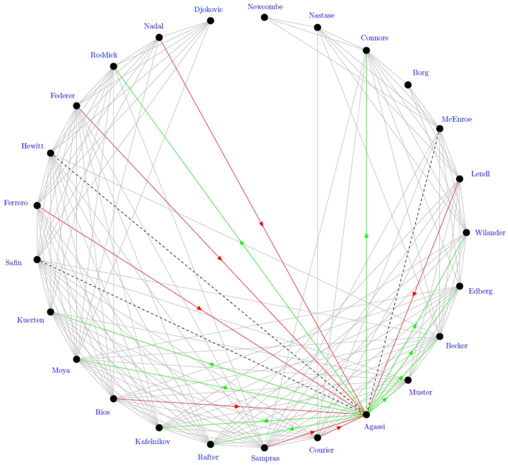
- voting, web search, recommendation systems, admissions, sports competitions, ...



PageRank

figure credit: Dzenan Hamzic

Rank aggregation from pairwise comparisons

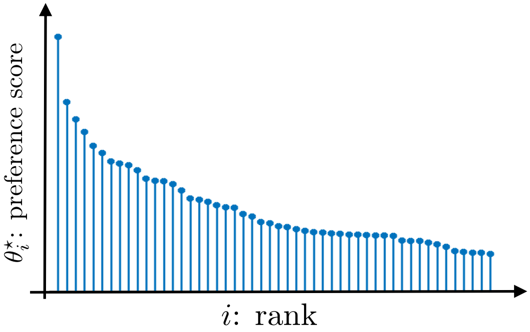


pairwise comparisons for ranking top tennis players

figure credit: Bozóki, Csató, Temesi

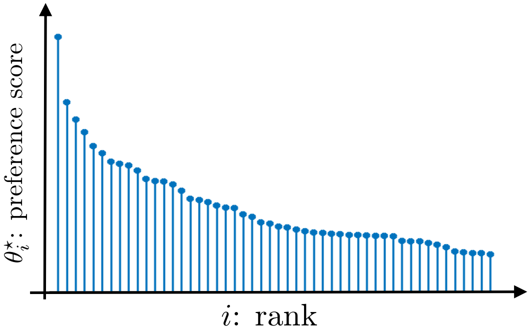
Bradley-Terry-Luce model

Assign latent score to each of n items $\boldsymbol{\theta}^\star = [\theta_1^\star, \cdots, \theta_n^\star]$



Bradley-Terry-Luce model

Assign latent score to each of n items $\boldsymbol{\theta}^\star = [\theta_1^\star, \cdots, \theta_n^\star]$



- Bradley-Terry-Luce (logistic) model assumes

$$\mathbb{P}\{\text{item } j \text{ beats item } i\} = \frac{e^{\theta_i^\star}}{e^{\theta_i^\star} + e^{\theta_j^\star}}$$

WLOG, assume $\mathbf{1}_n^\top \boldsymbol{\theta}^\star = 0$

Typical ranking procedures

Estimate latent scores

$\longrightarrow$ rank items based on score estimates

Top-K ranking

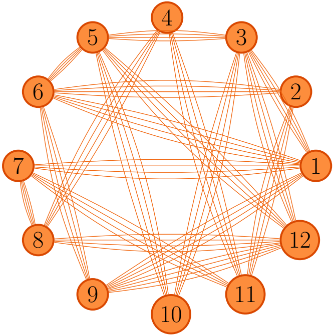
Estimate latent scores

$\longrightarrow$ rank items based on score estimates

Goal: identify the set of top-K items under minimal sample size

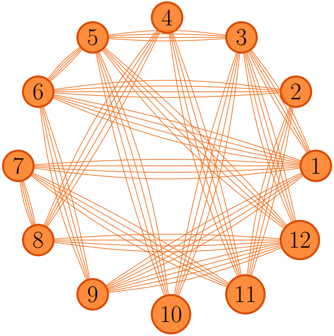
Sampling model

Sampling on comparison graph $\mathcal{G} = ([n], \mathcal{E})$: i, j are compared iff $(i, j) \in \mathcal{E}$



Sampling model

Sampling on comparison graph $\mathcal{G} = ([n], \mathcal{E})$: i, j are compared iff $(i, j) \in \mathcal{E}$



- For each $(i, j) \in \mathcal{E}$, obtain L paired comparisons

$$y_{i,j}^{(l)} \overset{\text{ind.}}{=} \begin{cases} 1, & \text{with prob. } \frac{e^{\theta_j^\star}}{e^{\theta_i^\star} + e^{\theta_j^\star}} \\ 0, & \text{else} \end{cases} \qquad 1 \le l \le L$$

Maximum likelihood estimator

Define $y_{i,j} := \frac{1}{L}\sum_{l=1}^{L} y_{i,j}^{(l)}$. Negative log-likelihood is given by

$$\mathcal{L}(\boldsymbol{\theta}) := -\frac{1}{L}\sum_{(i,j)\in\mathcal{E}}\sum_{l=1}^{L}\log\left(y_{ji}^{(l)}\frac{e^{\theta_i}}{e^{\theta_i}+e^{\theta_j}} + (1-y_{ji}^{(l)})\frac{e^{\theta_j}}{e^{\theta_i}+e^{\theta_j}}\right)$$

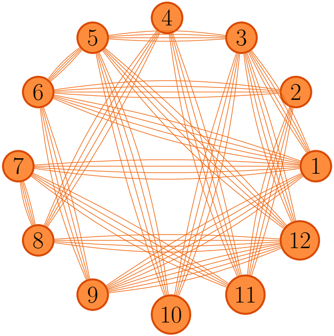
$$= \sum_{(i,j)\in\mathcal{E}}\left(-y_{ji}(\theta_i-\theta_j) + \log(1+e^{\theta_i-\theta_j})\right)$$

Maximum likelihood estimator (MLE)

$$\widehat{\boldsymbol{\theta}}_{\mathrm{MLE}} := \arg\min_{\boldsymbol{\theta}:\mathbf{1}_n^\top\boldsymbol{\theta}=0} \quad \mathcal{L}(\boldsymbol{\theta})$$

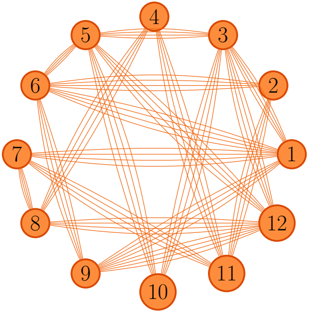
Prior art: Uniform sampling

- Comparison graph: Erdős–Rényi graph $\mathcal{G}_{\mathrm{ER}} \sim \mathcal{G}(n, p)$



MLE is optimal

comparison graph $\mathcal{G}(n,p)$; sample size $\asymp n^2 pL$

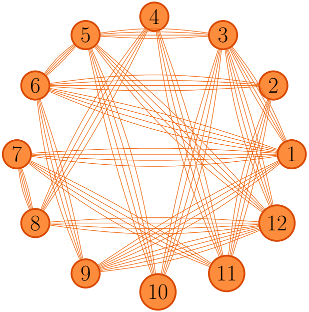


Theorem 1 (Chen, Fan, Ma, Wang, AoS 2019)

When $p \gtrsim \frac{\log n}{n}$, regularized MLE achieves optimal sample complexity for top-K ranking

MLE is optimal

comparison graph $\mathcal{G}(n, p)$; sample size $\asymp n^2 pL$



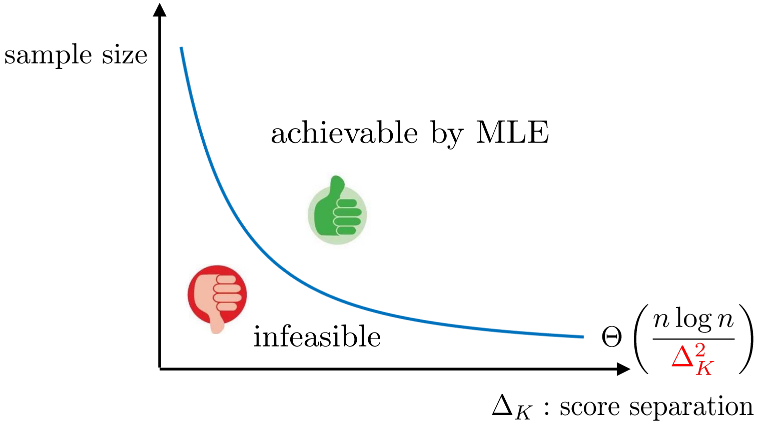
Theorem 1 (Chen, Fan, Ma, Wang, AoS 2019)

When $p \gtrsim \frac{\log n}{n}$, regularized MLE achieves optimal sample complexity for top-K ranking

vanilla MLE works; see Chen, Gao, Zhang, AoS 2022

Optimal sample complexity of MLE

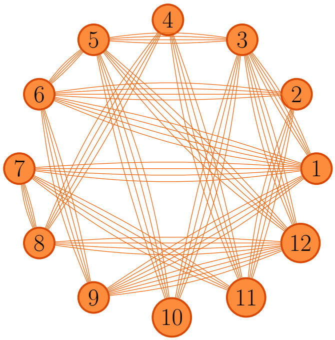
comparison graph $\mathcal{G}(n,p)$; sample size $\asymp n^2 pL$



- $\Delta_K := \theta^\star_K - \theta^\star_{K+1}$: score separation (assuming items are ordered)

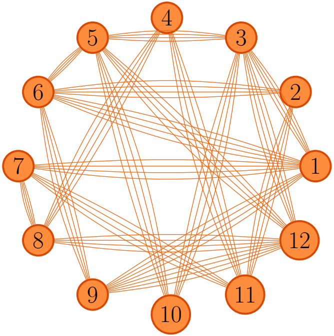
Prior art: General sampling

- General comparison graph $\mathcal{G} = ([n], \mathcal{E})$



Prior art: General sampling

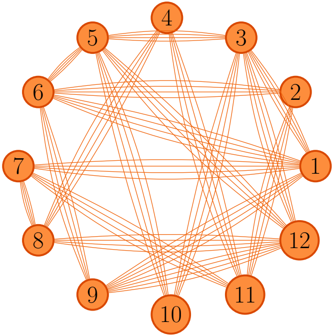
- General comparison graph $\mathcal{G} = ([n], \mathcal{E})$



- General performance guarantees for MLE in ranking

Prior art: General sampling

- General comparison graph $\mathcal{G} = ([n], \mathcal{E})$



- General performance guarantees for MLE in ranking
- Far from satisfactory: gaurantees are loose even when $\mathcal{G} = \mathcal{G}_{\mathrm{ER}}$

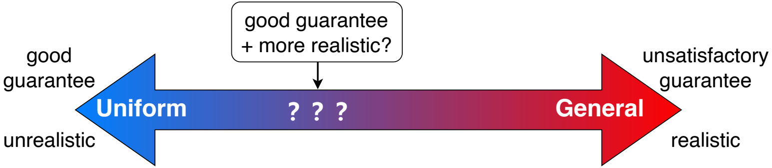
Loose guarantees for MLE

Theorem 2 (Li, Shrotriya, Rinaldo, ICML 2022)

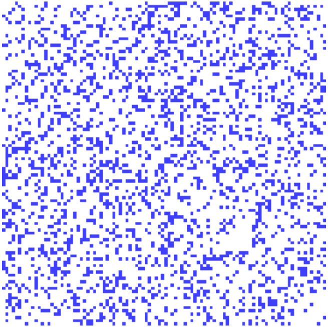
For uniform sampling, when $p \gtrsim \frac{\log n}{n}$*, MLE achieves exact recovery when* $n^2 pL \geq \frac{1}{p} \cdot \frac{n \log n}{\Delta_K^2}$

- Exceeds optimal sample complexity by factor $\frac{1}{p}$
- Extremely large when comparison graph is sparse, i.e., $p \asymp \frac{\log n}{n}$

A middle ground?



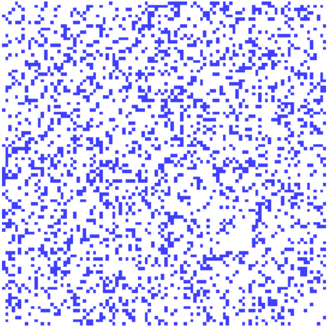
Top-K ranking with a monotone adversary



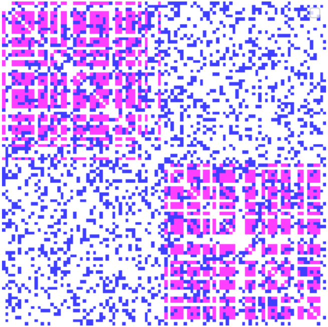
$\mathcal{G}_{\mathrm{ER}} = ([n], \mathcal{E}_{\mathrm{ER}})$

Top-K ranking with a monotone adversary

—aka semi-random adversary



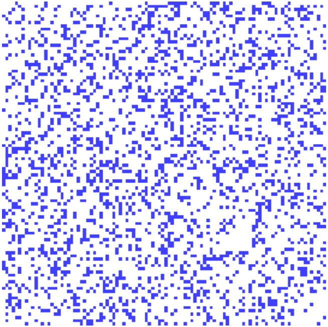
$\mathcal{G}_{\mathrm{ER}} = ([n], \mathcal{E}_{\mathrm{ER}})$



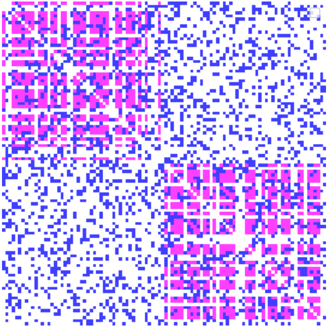
$\mathcal{G}_{\mathrm{SR}} = ([n], \mathcal{E}_{\mathrm{SR}})$ with added edges

Top-K ranking with a monotone adversary

—aka semi-random adversary



$\mathcal{G}_{\mathrm{ER}} = ([n], \mathcal{E}_{\mathrm{ER}})$



$\mathcal{G}_{\mathrm{SR}} = ([n], \mathcal{E}_{\mathrm{SR}})$ with added edges

Can we identify top-K items under monotone adversary?

A detour: semi-random models

- Blum and Spencer 1995 introduced it as intermediary between average-case and worst-case analysis

A detour: semi-random models

- Blum and Spencer 1995 introduced it as intermediary between average-case and worst-case analysis
- Since then, it has been popular for many statistical problems
    - Community detection
    - Clustering via Gaussian mixture models
    - Compressed sensing
    - Matrix completion
    - Dueling optimization
    - ...

A detour: semi-random models

- Blum and Spencer 1995 introduced it as intermediary between average-case and worst-case analysis
- Since then, it has been popular for many statistical problems
  - Community detection
  - Clustering via Gaussian mixture models
  - Compressed sensing
  - Matrix completion
  - Dueling optimization
  - ...
- Poses serious algorithmic and analytical challenges

How to tackle monotone adversary in ranking?

Intuition: mimicking oracle

If we have oracle knowledge of $\mathcal{E}_{\mathrm{ER}}$

Intuition: mimicking oracle

If we have oracle knowledge of $\mathcal{E}_{\mathrm{ER}}$

$\Downarrow$

We would run MLE using edges in $\mathcal{E}_{\mathrm{ER}}$

Intuition: mimicking oracle

If we have oracle knowledge of $\mathcal{E}_{\mathrm{ER}}$

$\Downarrow$

We would run MLE using edges in $\mathcal{E}_{\mathrm{ER}}$

$\Downarrow$

Equivalent to weighted MLE with unit weight on $\mathcal{E}_{\mathrm{ER}}$

Intuition: mimicking oracle

If we have oracle knowledge of $\mathcal{E}_{\mathrm{ER}}$

$$\Downarrow$$

We would run MLE using edges in $\mathcal{E}_{\mathrm{ER}}$

$$\Downarrow$$

Equivalent to weighted MLE with unit weight on $\mathcal{E}_{\mathrm{ER}}$

$$\Downarrow$$

We don't know $\mathcal{E}_{\mathrm{ER}}$

Intuition: mimicking oracle

If we have oracle knowledge of $\mathcal{E}_{\mathrm{ER}}$

$$\Downarrow$$

We would run MLE using edges in $\mathcal{E}_{\mathrm{ER}}$

$$\Downarrow$$

Equivalent to weighted MLE with unit weight on $\mathcal{E}_{\mathrm{ER}}$

$$\Downarrow$$

We don't know $\mathcal{E}_{\mathrm{ER}}$

$$\Downarrow$$

Can we find weights that mimic the above?

Our approach: Weighted MLE

Given weights $\{w_{ij}\}$ supported on $\mathcal{E}_{\mathrm{SR}}$, weighted negative log-likelihood is

$$\mathcal{L}_w(\boldsymbol{\theta}) := \sum_{(i,j):i>j} w_{ij} \left(-y_{ji}(\theta_i - \theta_j) + \log(1 + e^{\theta_i - \theta_j}) \right)$$

Weighted MLE:

$$\widehat{\boldsymbol{\theta}}_w := \arg \min_{\boldsymbol{\theta}:\mathbf{1}_n^\top \boldsymbol{\theta}=0} \quad \mathcal{L}_w(\boldsymbol{\theta})$$

Optimization-based reweighting

Given weights $\{w_{ij}\}$, weighted graph Laplacian is

$$\boldsymbol{L}_w := \sum_{(i,j):i>j} w_{ij}(\boldsymbol{e}_i - \boldsymbol{e}_j)(\boldsymbol{e}_i - \boldsymbol{e}_j)^\top$$

Weight finding via optimization:

$$\begin{aligned} \max_{\boldsymbol{w}} \quad & \lambda_{n-1}(\boldsymbol{L}_w) \\ \text{s.t.} \quad & \sum_i w_{ij} \leq 2np \quad \text{for all } j \\ & 0 \leq w_{ij} \leq 1 \quad \text{for all } i, j \end{aligned}$$

Optimization-based reweighting

Given weights $\{w_{ij}\}$, weighted graph Laplacian is

$$\boldsymbol{L}_w := \sum_{(i,j):i>j} w_{ij}(\boldsymbol{e}_i - \boldsymbol{e}_j)(\boldsymbol{e}_i - \boldsymbol{e}_j)^\top$$

Weight finding via optimization:

$$\begin{aligned} \max_{\boldsymbol{w}} \quad & \lambda_{n-1}(\boldsymbol{L}_w) \\ \text{s.t.} \quad & \sum_i w_{ij} \leq 2np \quad \text{for all } j \\ & 0 \leq w_{ij} \leq 1 \quad \text{for all } i, j \end{aligned}$$

— come back to this later...

Optimal control of entrywise error

Theorem 3 (Yang, Chen, Oreccia, Ma, 2024)

When $p \gtrsim \frac{\log(n)}{n}$ *and* $npL \gtrsim \log^3(n)$, *weighted MLE* $\widehat{\boldsymbol{\theta}}_w$ *obeys*

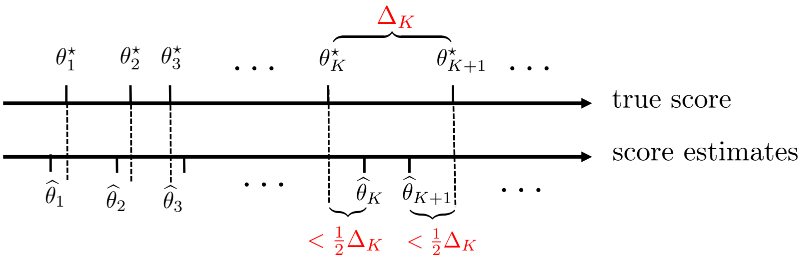
$$\|\widehat{\boldsymbol{\theta}}_w - \boldsymbol{\theta}^\star\|_\infty \lesssim \sqrt{\frac{\log(n)}{npL}}$$

Optimal control of entrywise error

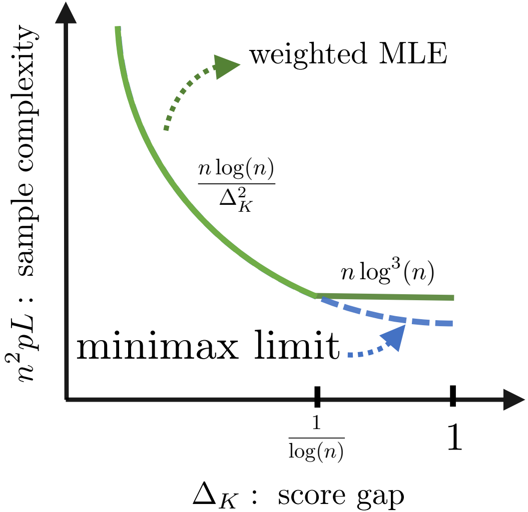
Theorem 3 (Yang, Chen, Oreccia, Ma, 2024)

When $p \gtrsim \frac{\log(n)}{n}$ *and* $npL \gtrsim \log^3(n)$, *weighted MLE* $\widehat{\boldsymbol{\theta}}_w$ *obeys*

$$\|\widehat{\boldsymbol{\theta}}_w - \boldsymbol{\theta}^\star\|_\infty \lesssim \sqrt{\frac{\log(n)}{npL}}$$

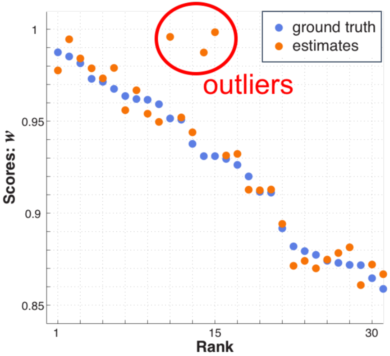


Near-optimal sample complexity



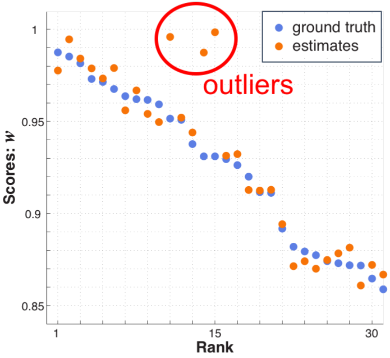
A little analysis

Challenge: Small ℓ_2 loss $\neq$ high ranking accuracy

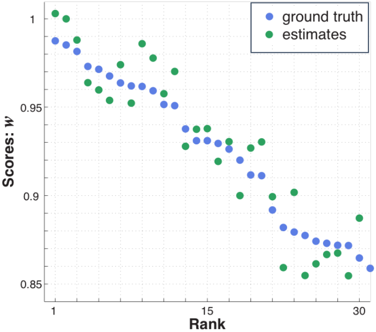


$$\text{Top 3}: \{15, 11, 2\}$$

Challenge: Small ℓ_2 loss $\neq$ high ranking accuracy

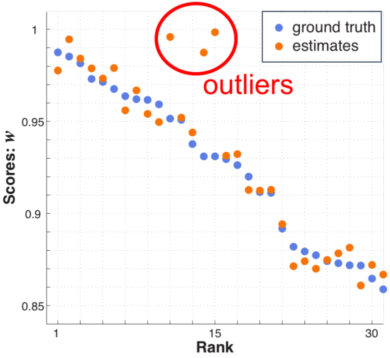


Top 3 : {15, 11, 2}

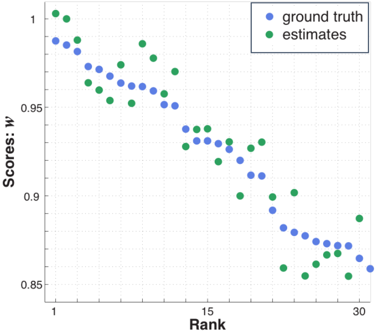


Top 3: {1, 2, 3}

Challenge: Small ℓ_2 loss $\neq$ high ranking accuracy



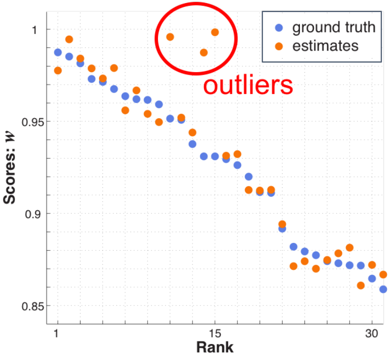
Top 3 : $\{15, 11, 2\}$



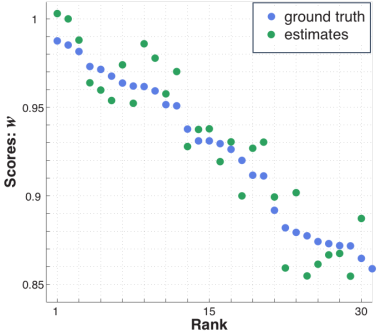
Top 3: $\{1, 2, 3\}$

These two estimates have same ℓ_2 loss, but output different rankings

Challenge: Small ℓ_2 loss $\neq$ high ranking accuracy



Top 3 : $\{15, 11, 2\}$



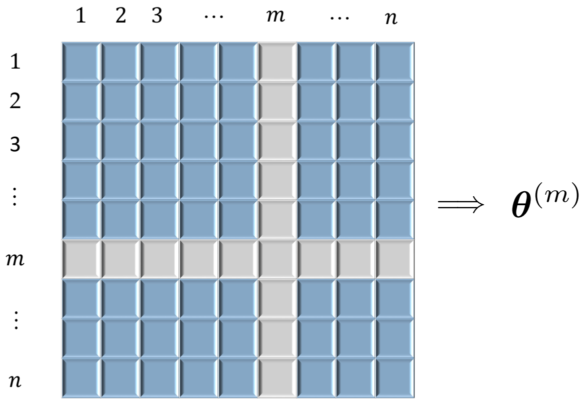
Top 3: $\{1, 2, 3\}$

These two estimates have same ℓ_2 loss, but output different rankings

Need to control entrywise error

Prior art: Leave-one-out analysis

For each $1 \leq m \leq n$, introduce leave-one-out estimate $\boldsymbol{\theta}^{(m)}$



$\boldsymbol{y} = [y_{i,j}]_{1 \leq i,j \leq n}$

Prior art: Leave-one-out analysis

Simple triangle inequality tells us that

$$|\theta_m - \theta_m^\star| \le \underbrace{\left|\theta_m^{(m)} - \theta_m^\star\right|}_{\text{Leave-one-out estimation error}} + \underbrace{\left\|\theta_m^{(m)} - \theta_m\right\|_2}_{\text{Leave-one-out perturbation}}$$

Leave-one-out estimation error ⟶ statistical independence

Leave-one-out perturbation ⟶ stability

Leave-one-out analysis is loose for general sampling

ℓ_∞-Bounds of the MLE in the BTL Model under General Comparison Graphs

In this case our derived ℓ_∞-bound cannot achieve the rate established in Chen et al. (2019), Chen et al. (2020), though our ℓ_2-bound exhibits the optimal rate proved in Negahban et al. (2017). The reason why our bound does not imply the optimal ℓ_∞-rate under a Erdös-Rényi comparison graph is that our bound is a sample-wise bound and thus cannot leverage some regular property of Erdös-Rényi graph beyond algebraic connectivity and degree homogeneity that is exhibited with high probability.

Leave-one-out analysis is loose for general sampling

ℓ_∞-Bounds of the MLE in the BTL Model under General Comparison Graphs

In this case our derived ℓ_∞-bound cannot achieve the rate established in Chen et al. (2019), Chen et al. (2020), though our ℓ_2-bound exhibits the optimal rate proved in Negahban et al. (2017). The reason why our bound does not imply the optimal ℓ_∞-rate under a Erdös-Rényi comparison graph is that our bound is a sample-wise bound and thus cannot leverage some regular property of Erdös-Rényi graph beyond algebraic connectivity and degree homogeneity that is exhibited with high probability.

Calls for new analysis beyond LOO

Trajectory-based analysis

Recall our goal is to analyze

$$\widehat{\boldsymbol{\theta}}_w := \arg \min_{\boldsymbol{\theta}: \mathbf{1}_n^\top \boldsymbol{\theta} = 0} \quad \mathcal{L}_w(\boldsymbol{\theta})$$

Trajectory-based analysis

Recall our goal is to analyze

$$\widehat{\boldsymbol{\theta}}_w := \arg \min_{\boldsymbol{\theta}: \mathbf{1}_n^\top \boldsymbol{\theta} = 0} \quad \mathcal{L}_w(\boldsymbol{\theta})$$

Instead of directly analyzing minimizer, we analyze sequence of iterates given by *preconditioned* gradient descent

—inspired by recent work of Chen 2023

Setting $\theta^0 = \boldsymbol{\theta}^\star$, we run

$$\boldsymbol{\theta}^{t+1} = \boldsymbol{\theta}^t - \eta \nabla^2 \mathcal{L}_w(\boldsymbol{\theta}^\star)^\dagger \nabla \mathcal{L}_w(\boldsymbol{\theta}^t),$$

Preconditioning decouples coordinates

Define error vector $\boldsymbol{\delta}^t := \boldsymbol{\theta}^t - \boldsymbol{\theta}^\star$. *Preconditioned* gradient descent yields recursive relation

$$\boldsymbol{\delta}^{t+1} = (1-\eta)\,\boldsymbol{\delta}^t - \frac{\eta}{L}\left(\nabla^2 \mathcal{L}_w(\boldsymbol{\theta}^\star)^\dagger \boldsymbol{B}\widehat{\boldsymbol{\epsilon}} - L \cdot \nabla^2 \mathcal{L}_w(\boldsymbol{\theta}^\star)^\dagger \boldsymbol{r}^t\right)$$

Preconditioning decouples coordinates

Define error vector $\boldsymbol{\delta}^t := \boldsymbol{\theta}^t - \boldsymbol{\theta}^\star$. *Preconditioned* gradient descent yields recursive relation

$$\boldsymbol{\delta}^{t+1} = (1-\eta)\,\boldsymbol{\delta}^t - \frac{\eta}{L}\left(\nabla^2\mathcal{L}_w(\boldsymbol{\theta}^\star)^\dagger \boldsymbol{B}\widehat{\boldsymbol{\epsilon}} - L\cdot\nabla^2\mathcal{L}_w(\boldsymbol{\theta}^\star)^\dagger \boldsymbol{r}^t\right)$$

- $(1-\eta)\,\boldsymbol{\delta}^t$: approximate contraction in each coordinate

Preconditioning decouples coordinates

Define error vector $\boldsymbol{\delta}^t := \boldsymbol{\theta}^t - \boldsymbol{\theta}^\star$. *Preconditioned* gradient descent yields recursive relation

$$\boldsymbol{\delta}^{t+1} = (1-\eta)\,\boldsymbol{\delta}^t - \frac{\eta}{L}\left(\nabla^2\mathcal{L}_w(\boldsymbol{\theta}^\star)^\dagger \boldsymbol{B}\widehat{\boldsymbol{\epsilon}} - L\cdot\nabla^2\mathcal{L}_w(\boldsymbol{\theta}^\star)^\dagger \boldsymbol{r}^t\right)$$

- $(1-\eta)\,\boldsymbol{\delta}^t$: approximate contraction in each coordinate
- $\nabla^2\mathcal{L}_w(\boldsymbol{\theta}^\star)^\dagger \boldsymbol{B}\widehat{\boldsymbol{\epsilon}}$: sampling error due to noisy comparisons

Preconditioning decouples coordinates

Define error vector $\boldsymbol{\delta}^t := \boldsymbol{\theta}^t - \boldsymbol{\theta}^\star$. *Preconditioned* gradient descent yields recursive relation

$$\boldsymbol{\delta}^{t+1} = (1-\eta)\,\boldsymbol{\delta}^t - \frac{\eta}{L}\left(\nabla^2 \mathcal{L}_w(\boldsymbol{\theta}^\star)^\dagger \boldsymbol{B}\widehat{\boldsymbol{\epsilon}} - L \cdot \nabla^2 \mathcal{L}_w(\boldsymbol{\theta}^\star)^\dagger \boldsymbol{r}^t\right)$$

- $(1-\eta)\,\boldsymbol{\delta}^t$: approximate contraction in each coordinate
- $\nabla^2 \mathcal{L}_w(\boldsymbol{\theta}^\star)^\dagger \boldsymbol{B}\widehat{\boldsymbol{\epsilon}}$: sampling error due to noisy comparisons
- $L \cdot \nabla^2 \mathcal{L}_w(\boldsymbol{\theta}^\star)^\dagger \boldsymbol{r}^t$: Taylor expansion error

Preconditioning decouples coordinates

Define error vector $\boldsymbol{\delta}^t := \boldsymbol{\theta}^t - \boldsymbol{\theta}^\star$. *Preconditioned* gradient descent yields recursive relation

$$\boldsymbol{\delta}^{t+1} = (1-\eta)\,\boldsymbol{\delta}^t - \frac{\eta}{L}\Big(\nabla^2\mathcal{L}_w(\boldsymbol{\theta}^\star)^\dagger \boldsymbol{B}\widehat{\boldsymbol{\epsilon}} - L\cdot\nabla^2\mathcal{L}_w(\boldsymbol{\theta}^\star)^\dagger \boldsymbol{r}^t\Big)$$

- $(1-\eta)\,\boldsymbol{\delta}^t$: approximate contraction in each coordinate
- $\nabla^2\mathcal{L}_w(\boldsymbol{\theta}^\star)^\dagger \boldsymbol{B}\widehat{\boldsymbol{\epsilon}}$: sampling error due to noisy comparisons
- $L\cdot\nabla^2\mathcal{L}_w(\boldsymbol{\theta}^\star)^\dagger \boldsymbol{r}^t$: Taylor expansion error

Key contribution:

relate the latter two to spectral properties of weighted graph

Master theorem for weighted MLE

Notation:

- $w_{\max} := \max_{i,j} w_{ij}$ be the maximum weight
- $d_{\max} := \max_{i\in[n]} \sum_{j:j\neq i} w_{ij}$ be the maximum (weighted) degree
- Weighted graph Laplacian

$$\boldsymbol{L}_w := \sum_{(i,j):i>j} w_{ij}(\boldsymbol{e}_i - \boldsymbol{e}_j)(\boldsymbol{e}_i - \boldsymbol{e}_j)^\top$$

Master theorem for weighted MLE

Theorem 4 (Yang, Chen, Oreccia, Ma, 2024)

When graph is connected, as long as

$$L \gg \frac{w_{\max}(d_{\max})^4 \log^3(n)}{(\lambda_{n-1}(\boldsymbol{L}_w))^5},$$

with high probability, we have

$$\|\widehat{\boldsymbol{\theta}}_w - \boldsymbol{\theta}^\star\|_\infty \lesssim \sqrt{\frac{w_{\max}\log(n)}{\lambda_{n-1}(\boldsymbol{L}_w)L}}$$

- LOO-free analysis
- Depends explicitly only on graph properties
- A by-product: optimality of MLE under uniform sampling

Optimality of MLE under uniform sampling

Corollary 5

When $p \gtrsim \log(n)/n$, *and* $npL \gtrsim \log^3(n)$, *vanilla MLE achieves*

$$\|\widehat{\boldsymbol{\theta}}_{\mathrm{MLE}} - \boldsymbol{\theta}^\star\|_\infty \lesssim \sqrt{\frac{\log(n)}{npL}}$$

Optimality of MLE under uniform sampling

Corollary 5

When $p \gtrsim \log(n)/n$*, and* $npL \gtrsim \log^3(n)$*, vanilla MLE achieves*

$$\|\widehat{\boldsymbol{\theta}}_{\mathrm{MLE}} - \boldsymbol{\theta}^\star\|_\infty \lesssim \sqrt{\frac{\log(n)}{npL}}$$

A three-line proof:

- vanilla MLE = weighted MLE with weight 1
- Compute graph properties

$$\begin{aligned} w_{\max} &\leq 1 \\ d_{\max} &\leq 2np \\ \lambda_{n-1}(\boldsymbol{L}_w) &\geq np/2 \end{aligned}$$

- Apply master theorem

Optimization-based reweighting

Master theorem motivates us to consider following optimization problem

$$\begin{aligned} \max_{\boldsymbol{w}} \quad & \lambda_{n-1}(\boldsymbol{L}_w) \\ \text{s.t.} \quad & \sum_i w_{ij} \leq 2np \quad \text{for all } j \\ & 0 \leq w_{ij} \leq 1 \quad \text{for all } i, j \end{aligned}$$

Since unit weights on $\mathcal{E}_{\mathrm{ER}}$ is feasible, we know the maximizer is at least as good as that for Erdős–Rényi graph

Optimization-based reweighting

Master theorem motivates us to consider following optimization problem

$$\begin{aligned} \max_{\boldsymbol{w}} \quad & \lambda_{n-1}(\boldsymbol{L}_w) \\ \text{s.t.} \quad & \sum_i w_{ij} \leq 2np \quad \text{for all } j \\ & 0 \leq w_{ij} \leq 1 \quad \text{for all } i, j \end{aligned}$$

Since unit weights on $\mathcal{E}_{\mathrm{ER}}$ is feasible, we know the maximizer is at least as good as that for Erdős–Rényi graph

Is this computationally friendly?

Efficient computation

In view of $\lambda_{n-1}(\boldsymbol{L}_w) = \min_{\boldsymbol{X}\in\Delta}\langle \boldsymbol{L_w}, \boldsymbol{X}\rangle$ with

$$\Delta \coloneqq \{\boldsymbol{X} \in \mathbb{R}^{n\times n} \mid \boldsymbol{X} \succeq 0 \ \wedge \ \langle \Pi_{\perp\mathbf{1}}, \boldsymbol{X}\rangle = 1\},$$

Efficient computation

In view of $\lambda_{n-1}(\boldsymbol{L}_w) = \min_{\boldsymbol{X}\in\Delta}\langle \boldsymbol{L_w}, \boldsymbol{X}\rangle$ with

$$\Delta \coloneqq \{\boldsymbol{X} \in \mathbb{R}^{n\times n} \mid \boldsymbol{X} \succeq 0 \ \wedge \ \langle \Pi_{\perp\mathbf{1}}, \boldsymbol{X}\rangle = 1\},$$

reweighting is equivalent to saddle-point semi-definite program (SDP)

$$\max_{\boldsymbol{w}\in\mathcal{F}} \min_{\boldsymbol{X}\in\Delta} \quad \langle \boldsymbol{L_w}, \boldsymbol{X}\rangle,$$

where $\mathcal{F}$ is feasible set

$$\mathcal{F} \coloneqq \{w_{ij} \mid \forall i, \sum_{j:(i,j)\in\mathcal{E}_{\mathrm{SR}}} w_{ij} \le 2np \ \wedge \ \forall (i,j) \in \mathcal{E}_{\mathrm{SR}}, w_{ij} \le 1\}$$

Efficient computation

In view of $\lambda_{n-1}(\boldsymbol{L}_w) = \min_{\boldsymbol{X}\in\Delta}\langle \boldsymbol{L}_{\boldsymbol{w}}, \boldsymbol{X}\rangle$ with

$$\Delta := \{\boldsymbol{X} \in \mathbb{R}^{n\times n} \mid \boldsymbol{X} \succeq 0 \ \wedge \ \langle \Pi_{\perp \mathbf{1}}, \boldsymbol{X}\rangle = 1\},$$

reweighting is equivalent to saddle-point semi-definite program (SDP)

$$\max_{\boldsymbol{w}\in\mathcal{F}} \min_{\boldsymbol{X}\in\Delta} \quad \langle \boldsymbol{L}_{\boldsymbol{w}}, \boldsymbol{X}\rangle,$$

where $\mathcal{F}$ is feasible set

$$\mathcal{F} := \{w_{ij} \mid \forall i, \sum_{j:(i,j)\in\mathcal{E}_{\mathrm{SR}}} w_{ij} \leq 2np \ \wedge \ \forall(i,j) \in \mathcal{E}_{\mathrm{SR}}, w_{ij} \leq 1\}$$

Key observation: it is a zero-sum game between $\boldsymbol{w}$ and $\boldsymbol{X}$

Matrix multiplicative weight update

—Arora and Kale, JACM 2016

Initialization $\boldsymbol{w}^{(0)} = \boldsymbol{0}$;
For $t = 1, 2, \ldots, T$ do

- Update $\boldsymbol{X}^{(t)}$:

$$\boldsymbol{Z}^{(t)} = \exp\left\{-\eta \sum_{s=0}^{t-1} \boldsymbol{L}_{\boldsymbol{w}^{(s)}}\right\}, \quad \text{and} \quad \boldsymbol{X}^{(t)} = \frac{\boldsymbol{Z}^{(t)}}{\langle \Pi_{\perp \boldsymbol{1}}, \boldsymbol{Z}^{(t)} \rangle}$$

- Update $\boldsymbol{w}^{(t)}$:

$$\boldsymbol{w}^{(t)} := \arg\max_{\boldsymbol{w} \in \mathcal{F}} \langle \boldsymbol{L}_{\boldsymbol{w}}, \boldsymbol{X}^{(t)} \rangle$$

Matrix multiplicative weight update

—Arora and Kale, JACM 2016

Initialization $\boldsymbol{w}^{(0)} = \boldsymbol{0}$;
For $t = 1, 2, \ldots, T$ do

- Update $\boldsymbol{X}^{(t)}$:

$$\boldsymbol{Z}^{(t)} = \exp\left\{-\eta \sum_{s=0}^{t-1} \boldsymbol{L}_{\boldsymbol{w}^{(s)}}\right\}, \quad \text{and} \quad \boldsymbol{X}^{(t)} = \frac{\boldsymbol{Z}^{(t)}}{\langle \Pi_{\perp \boldsymbol{1}}, \boldsymbol{Z}^{(t)} \rangle}$$

- Update $\boldsymbol{w}^{(t)}$:

$$\boldsymbol{w}^{(t)} := \arg\max_{\boldsymbol{w} \in \mathcal{F}} \langle \boldsymbol{L}_{\boldsymbol{w}}, \boldsymbol{X}^{(t)} \rangle$$

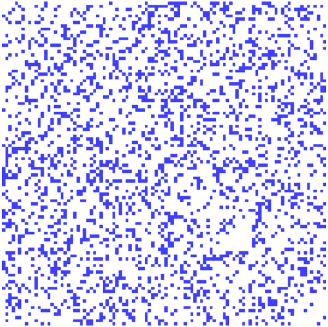
Converge even if updates are approximately computed
$\Longrightarrow$ near-linear time computation

Summary on computational guarantee

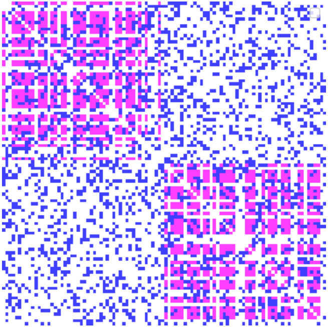
- Reweighting is saddle-point SDP
- We leverage matrix multiplicative weight update framework developed by Arora and Kale 2016
- It suffices to approximately compute updates
- These lead to near-linear-time computational complexity

Numerical experiment

- Comparison graph



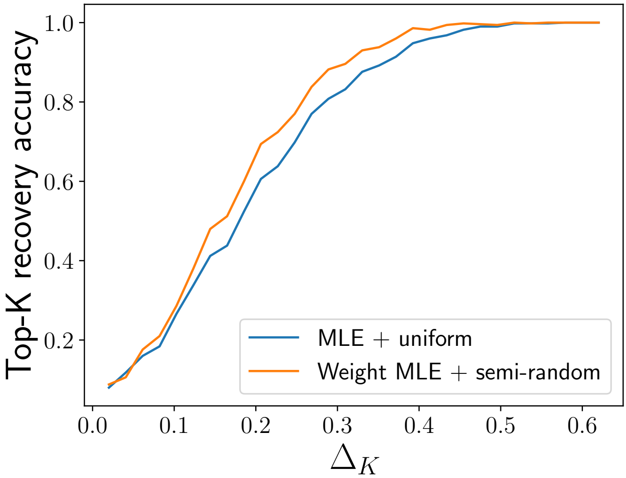
$\mathcal{G}_{\mathrm{ER}} = ([n], \mathcal{E}_{\mathrm{ER}})$



$\mathcal{G}_{\mathrm{SR}} = ([n], \mathcal{E}_{\mathrm{SR}})$ with added edges

- Score vector $\theta^{\star}_{1:K} = \Delta_K$, and $\theta^{\star}_{K+1:n} = 0$

Numerical experiment



Concluding remarks

Weighted MLE is statistically and computationally efficient for top-K ranking with monotone adversary

- A novel analysis of weighted MLE with general weights
- An efficient algorithm to approximately solve SDP-based reweighting

Concluding remarks

Weighted MLE is statistically and computationally efficient for top-K ranking with monotone adversary

- A novel analysis of weighted MLE with general weights
- An efficient algorithm to approximately solve SDP-based reweighting

Future directions:

- Is weighted MLE necessary?
- Stronger adversary?

Concluding remarks

Weighted MLE is statistically and computationally efficient for top-K ranking with monotone adversary

- A novel analysis of weighted MLE with general weights
- An efficient algorithm to approximately solve SDP-based reweighting

Future directions:

- Is weighted MLE necessary?
- Stronger adversary?

Papers:

- Y. Yang, A. Chen, L. Orecchia, C. Ma, "Top-K ranking with a monotone adversary," arXiv:2402.07445, 2024
- Y. Chen, J. Fan, C. Ma, K. Wang, "Spectral method and regularized MLE are both optimal for top-K ranking," Annals of Statistics, 2019