Noisy Sorting Capacity

Ziao Wang[1] Nadim Ghaddar[2] Lele Wang[1]

[1]University of British Columbia
[2]University of California, San Diego

Conference on Information Sciences and Systems (CISS)
Princeton University

March 11, 2022

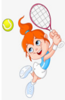
Motivation

- Sorting in the presence of noise

Motivation

- Sorting in the presence of noise

  E.g. Ranking tennis players according to their strengths

A     *B*

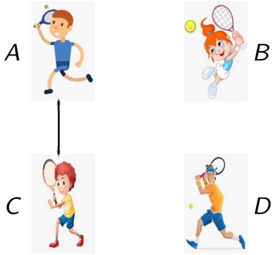
C     *D*

- With some probability, a weaker player can still win over a stronger player.

Motivation

- Sorting in the presence of noise

  E.g. Ranking tennis players according to their strengths

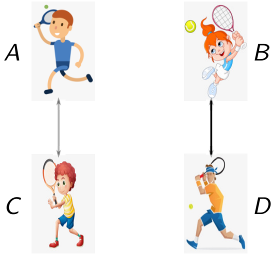


- With some probability, a weaker player can still win over a stronger player.

Motivation

- Sorting in the presence of noise

  E.g. Ranking tennis players according to their strengths

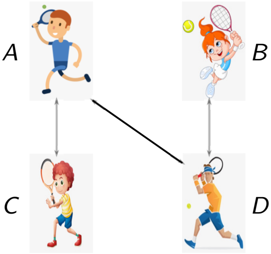


- With some probability, a weaker player can still win over a stronger player.

Motivation

- Sorting in the presence of noise

  E.g. Ranking tennis players according to their strengths

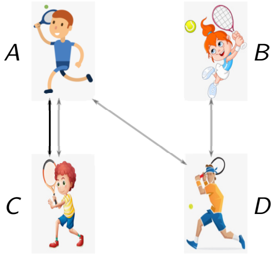


- With some probability, a weaker player can still win over a stronger player.

Motivation

- Sorting in the presence of noise

  E.g. Ranking tennis players according to their strengths

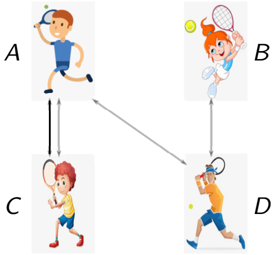


- With some probability, a weaker player can still win over a stronger player.

Motivation

- Sorting in the presence of noise

  E.g. Ranking tennis players according to their strengths



- With some probability, a weaker player can still win over a stronger player.

- Applications
  - Ranking teams in a sports tournament
  - Recommender systems
  - Peer grading
  - · · ·

Noisy Sorting Problem

- Let $\theta_1, \ldots, \theta_n \in \mathbb{R}$.
- **Goal:** Find π s.t. $\theta_{\pi(1)} < \cdots < \theta_{\pi(n)}$ using pairwise comparisons.
  - At kth time step, submit query $(U_k, V_k) \triangleq (\theta_i, \theta_j)$ for $i \neq j$.
  - Receive noisy response
    $$Y_k = \mathbb{1}_{\{U_k < V_k\}} \oplus Z_k,$$
    where $Z_k \sim \mathrm{Bern}(p)$, for some fixed and known p.
  - After m queries, compute estimate $\hat{\pi}$ of π.
  - Exact recovery of π is desired.

Noisy Sorting Problem

- Let $\theta_1, \ldots, \theta_n \in \mathbb{R}$.
- **Goal:** Find π s.t. $\theta_{\pi(1)} < \cdots < \theta_{\pi(n)}$ using pairwise comparisons.
  - At kth time step, submit query $(U_k, V_k) \triangleq (\theta_i, \theta_j)$ for $i \neq j$.
  - Receive noisy response
  $$Y_k = \mathbb{1}_{\{U_k < V_k\}} \oplus Z_k,$$
  where $Z_k \sim \mathrm{Bern}(p)$, for some fixed and known p.
  - After m queries, compute estimate $\hat{\pi}$ of π.
  - Exact recovery of π is desired.
- Noiseless case ($p = 0$):
  - $m = \Theta(n \log n)$ is both sufficient and necessary.
  - E.g., merge sort algorithm attains lower bound.

Noisy Sorting Problem

- Let $\theta_1, \ldots, \theta_n \in \mathbb{R}$.
- **Goal:** Find π s.t. $\theta_{\pi(1)} < \cdots < \theta_{\pi(n)}$ using pairwise comparisons.
  - At kth time step, submit query $(U_k, V_k) \triangleq (\theta_i, \theta_j)$ for $i \neq j$.
  - Receive noisy response
    $$Y_k = \mathbb{1}_{\{U_k < V_k\}} \oplus Z_k,$$
    where $Z_k \sim \mathrm{Bern}(p)$, for some fixed and known p.
  - After m queries, compute estimate $\hat{\pi}$ of π.
  - Exact recovery of π is desired.
- Noiseless case ($p = 0$):
  - $m = \Theta(n \log n)$ is both sufficient and necessary.
  - E.g., merge sort algorithm attains lower bound.
- Noisy case ($p > 0$):
  - $m = \Theta(n \log n)$ is both sufficient and necessary.
  - E.g., algorithm in [1] attains lower bound.

[1] U. Feige, P. Raghavan, D. Peleg, and E. Upfal. "Computing with Noisy Information". In: SIAM J. Comput., 1994.

Related Work

- Active ranking
  - $p = p_{i,j}$ is query-dependent and unknown [2]–[6].

---

[2] R. Heckel, N. B. Shah, K. Ramchandran, and M. J. Wainwright, "Active ranking from pairwise comparisons and when parametric assumptions do not help," The Annals of Statistics, 2019.

[3] N. B. Shah and M. J. Wainwright, "Simple, robust and optimal ranking from pairwise comparisons," Journal of Machine Learning Research, 2018.

[4] M. Falahatgar, A. Orlitsky, V. Pichapati, and A. T. Suresh, "Maximum selection and ranking under noisy comparisons," in Proceedings of ICML, 2017.

[5] S. Mohajer, C. Suh, and A. Elmahdy, "Active learning for top-k rank aggregation from noisy comparisons," in Proceedings of ICML, 2017.

[6] A. Agarwal, S. Agarwal, S. Assadi, and S. Khanna, "Learning with limited rounds of adaptivity: Coin tossing, multi-armed bandits, and ranking from pairwise comparisons," in Proceedings of COLT, 2017.

Related Work

- Active ranking
  - $p = p_{i,j}$ is query-dependent and unknown [2]–[6].
- Bradley–Terry–Luce Model
  - $p_{i,j}$ is a function of the *scores* of elements i and j [7]–[8]

[2] R. Heckel, N. B. Shah, K. Ramchandran, and M. J. Wainwright, "Active ranking from pairwise comparisons and when parametric assumptions do not help," The Annals of Statistics, 2019.

[3] N. B. Shah and M. J. Wainwright, "Simple, robust and optimal ranking from pairwise comparisons," Journal of Machine Learning Research, 2018.

[4] M. Falahatgar, A. Orlitsky, V. Pichapati, and A. T. Suresh, "Maximum selection and ranking under noisy comparisons," in Proceedings of ICML, 2017.

[5] S. Mohajer, C. Suh, and A. Elmahdy, "Active learning for top-k rank aggregation from noisy comparisons," in Proceedings of ICML, 2017.

[6] A. Agarwal, S. Agarwal, S. Assadi, and S. Khanna, "Learning with limited rounds of adaptivity: Coin tossing, multi-armed bandits, and ranking from pairwise comparisons," in Proceedings of COLT, 2017.

[7] W. Ren, J. Liu, and N. B. Shroff, "Pac ranking from pairwise and listwise queries: Lower bounds and upper bounds," 2018.

[8] M. Ajtai, V. Feldman, A. Hassidim, and J. Nelson, "Sorting and selection with imprecise comparisons," in ACM Transactions on Algorithms, 2009.

- Active ranking
  - $p = p_{i,j}$ is query-dependent and unknown [2]–[6].
- Bradley–Terry–Luce Model
  - $p_{i,j}$ is a function of the *scores* of elements i and j [7]–[8]
- Approximate permutation recovery
  - Distance metric between permutations is to be minimized [9]–[10]

[2] R. Heckel, N. B. Shah, K. Ramchandran, and M. J. Wainwright, "Active ranking from pairwise comparisons and when parametric assumptions do not help," The Annals of Statistics, 2019.

[3] N. B. Shah and M. J. Wainwright, "Simple, robust and optimal ranking from pairwise comparisons," Journal of Machine Learning Research, 2018.

[4] M. Falahatgar, A. Orlitsky, V. Pichapati, and A. T. Suresh, "Maximum selection and ranking under noisy comparisons," in Proceedings of ICML, 2017.

[5] S. Mohajer, C. Suh, and A. Elmahdy, "Active learning for top-k rank aggregation from noisy comparisons," in Proceedings of ICML, 2017.

[6] A. Agarwal, S. Agarwal, S. Assadi, and S. Khanna, "Learning with limited rounds of adaptivity: Coin tossing, multi-armed bandits, and ranking from pairwise comparisons," in Proceedings of COLT, 2017.

[7] W. Ren, J. Liu, and N. B. Shroff, "Pac ranking from pairwise and listwise queries: Lower bounds and upper bounds," 2018.

[8] M. Ajtai, V. Feldman, A. Hassidim, and J. Nelson, "Sorting and selection with imprecise comparisons," in ACM Transactions on Algorithms, 2009.

[9] M. Braverman and E. Mossel, "Sorting from noisy information," 2009.

[10] N. Ailon, M. Charikar, and A. Newman, "Aggregating inconsistent information: Ranking and clustering," Journal of the ACM, 2008.

Related Work

- Active ranking
  - $p = p_{i,j}$ is query-dependent and unknown [2]–[6].
- Bradley–Terry–Luce Model
  - $p_{i,j}$ is a function of the *scores* of elements i and j [7]–[8]
- Approximate permutation recovery
  - Distance metric between permutations is to be minimized [9]–[10]

Only the order of the query complexity is considered in prior work.

[2] R. Heckel, N. B. Shah, K. Ramchandran, and M. J. Wainwright, "Active ranking from pairwise comparisons and when parametric assumptions do not help," The Annals of Statistics, 2019.

[3] N. B. Shah and M. J. Wainwright, "Simple, robust and optimal ranking from pairwise comparisons," Journal of Machine Learning Research, 2018.

[4] M. Falahatgar, A. Orlitsky, V. Pichapati, and A. T. Suresh, "Maximum selection and ranking under noisy comparisons," in Proceedings of ICML, 2017.

[5] S. Mohajer, C. Suh, and A. Elmahdy, "Active learning for top-k rank aggregation from noisy comparisons," in Proceedings of ICML, 2017.

[6] A. Agarwal, S. Agarwal, S. Assadi, and S. Khanna, "Learning with limited rounds of adaptivity: Coin tossing, multi-armed bandits, and ranking from pairwise comparisons," in Proceedings of COLT, 2017.

[7] W. Ren, J. Liu, and N. B. Shroff, "Pac ranking from pairwise and listwise queries: Lower bounds and upper bounds," 2018.

[8] M. Ajtai, V. Feldman, A. Hassidim, and J. Nelson, "Sorting and selection with imprecise comparisons," in ACM Transactions on Algorithms, 2009.

[9] M. Braverman and E. Mossel, "Sorting from noisy information," 2009.

[10] N. Ailon, M. Charikar, and A. Newman, "Aggregating inconsistent information: Ranking and clustering," Journal of the ACM, 2008.

Question: What is the maximum ratio $\frac{n \log n}{m}$ such that $\mathrm{P}\{\hat{\pi} = \pi\} = 1$ asymptotically?

Proposed Framework

Question: What is the maximum ratio $\frac{n \log n}{m}$ such that $\mathrm{P}\{\hat{\pi} = \pi\} = 1$ asymptotically?

- Sorting rate: $R = \frac{n \log n}{m}$.
- (R, n) noisy sorting code consists of
  - A causal protocol $\{f_k\}_{k=1}^m$ for determining the queries
    $$(U_k, V_k) = f_k(Y^{k-1}, U^{k-1}, V^{k-1}).$$
  - An estimator $\hat{\pi} = \hat{\pi}(Y^m, U^m, V^m)$

Proposed Framework

Question: What is the maximum ratio $\frac{n \log n}{m}$ such that $\mathrm{P}\{\hat{\pi} = \pi\} = 1$ asymptotically?

- Sorting rate: $R = \frac{n \log n}{m}$.
- (R, n) noisy sorting code consists of
  - A causal protocol $\{f_k\}_{k=1}^{m}$ for determining the queries
  $$(U_k, V_k) = f_k(Y^{k-1}, U^{k-1}, V^{k-1}).$$
  - An estimator $\hat{\pi} = \hat{\pi}(Y^m, U^m, V^m)$
- R is achievable if $\exists$ a sequence of (R, n) noisy sorting codes s.t.
$$\lim_{n \to \infty} \max_{\pi} \mathrm{P}\{\hat{\pi} \neq \pi\} = 0.$$

Proposed Framework

Question: What is the maximum ratio $\frac{n \log n}{m}$ such that $\mathrm{P}\{\hat{\pi} = \pi\} = 1$ asymptotically?

- Sorting rate: $R = \frac{n \log n}{m}$.
- (R, n) noisy sorting code consists of
  - A causal protocol $\{f_k\}_{k=1}^m$ for determining the queries
  $$(U_k, V_k) = f_k(Y^{k-1}, U^{k-1}, V^{k-1}).$$
  - An estimator $\hat{\pi} = \hat{\pi}(Y^m, U^m, V^m)$
- R is achievable if $\exists$ a sequence of (R, n) noisy sorting codes s.t.
$$\lim_{n \to \infty} \max_{\pi} \mathrm{P}\{\hat{\pi} \neq \pi\} = 0.$$
- Noisy sorting capacity $C(p)$: supremum of all achievable sorting rates R.

Proposed Framework

Question: What is the maximum ratio $\frac{n \log n}{m}$ such that $\mathrm{P}\{\hat{\pi} = \pi\} = 1$ asymptotically?

- Sorting rate: $R = \frac{n \log n}{m}$.
- (R, n) noisy sorting code consists of
  - A causal protocol $\{f_k\}_{k=1}^m$ for determining the queries
  $$(U_k, V_k) = f_k(Y^{k-1}, U^{k-1}, V^{k-1}).$$
  - An estimator $\hat{\pi} = \hat{\pi}(Y^m, U^m, V^m)$
- R is achievable if $\exists$ a sequence of (R, n) noisy sorting codes s.t.
$$\lim_{n \to \infty} \max_{\pi} \mathrm{P}\{\hat{\pi} \neq \pi\} = 0.$$
- Noisy sorting capacity $C(p)$: supremum of all achievable sorting rates R.

In this work, we give upper and lower bounds on $C(p)$.

Main Results

Theorem 1 (Converse)

Any sequence of (R, n) *noisy sorting codes with* $\lim_{n\to\infty} \max_{\pi} P\{\hat{\pi} \neq \pi\} = 0$ *satisfies that*

$$R < 1 - H(p),$$

where $H(\cdot)$ *denotes the binary entropy function. That is,* $C(p) \leq 1 - H(p)$.

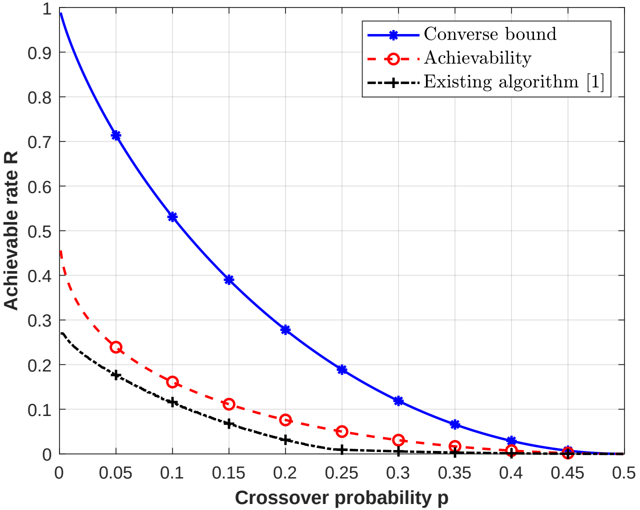
Theorem 2 (Achievability)

Any sorting rate

$$R < \frac{1}{2} - \frac{1}{2}\log\left(1 + 2\sqrt{p(1-p)}\right)$$

is achievable for the noisy sorting problem. That is, $C(p) \geq \frac{1}{2} - \frac{1}{2}\log\left(1 + 2\sqrt{p(1-p)}\right)$.

Main Results



[1] U. Feige, P. Raghavan, D. Peleg, and E. Upfal. "Computing with Noisy Information". In: SIAM J. Comput., 1994.

Main Results

Theorem 1 (Converse)

Any sequence of (R, n) *noisy sorting codes with* $\lim_{n\to\infty} \max_\pi P\{\hat{\pi} \neq \pi\} = 0$ *satisfies that*

$$R < 1 - H(p),$$

where $H(\cdot)$ *denotes the binary entropy function. That is,* $C(p) \leq 1 - H(p)$.

Theorem 2 (Achievability)

Any sorting rate

$$R < \frac{1}{2} - \frac{1}{2}\log\left(1 + 2\sqrt{p(1-p)}\right)$$

is achievable for the noisy sorting problem. That is, $C(p) \geq \frac{1}{2} - \frac{1}{2}\log\left(1 + 2\sqrt{p(1-p)}\right)$.

Main Results

Theorem 1 (Converse)

Any sequence of (R, n) *noisy sorting codes with* $\lim_{n\to\infty} \max_\pi P\{\hat{\pi} \neq \pi\} = 0$ *satisfies that*

$$R < 1 - H(p),$$

where $H(\cdot)$ *denotes the binary entropy function. That is,* $C(p) \leq 1 - H(p)$*.*

- Each query gives at most $1 - H(p)$ bits of information.
- Need to have $m\big(1 - H(p)\big) > \log n!$

Theorem 2 (Achievability)

Any sorting rate

$$R < \frac{1}{2} - \frac{1}{2}\log\left(1 + 2\sqrt{p(1-p)}\right)$$

is achievable for the noisy sorting problem. That is, $C(p) \geq \frac{1}{2} - \frac{1}{2}\log\left(1 + 2\sqrt{p(1-p)}\right)$*.*

Main Results

Theorem 1 (Converse)

Any sequence of (R, n) *noisy sorting codes with* $\lim_{n\to\infty} \max_\pi P\{\hat{\pi} \neq \pi\} = 0$ *satisfies that*

$$R < 1 - H(p),$$

where $H(\cdot)$ *denotes the binary entropy function. That is,* $C(p) \leq 1 - H(p)$.

- Each query gives at most $1 - H(p)$ bits of information.
- Need to have $m\big(1 - H(p)\big) > \log n!$

Theorem 2 (Achievability)

Any sorting rate

$$R < \frac{1}{2} - \frac{1}{2}\log\left(1 + 2\sqrt{p(1-p)}\right)$$

is achievable for the noisy sorting problem. That is, $C(p) \geq \frac{1}{2} - \frac{1}{2}\log\left(1 + 2\sqrt{p(1-p)}\right)$.

- Key ingredients:
  - Burnashev-Zigangirov (BZ) algorithm for noisy searching
  - Insertion sort

Detour: Noisy Searching Problem

- Let $(\tilde{\theta}_0, \ldots, \tilde{\theta}_n) \in \bar{\mathbb{R}}^{n+1}$ be sorted with $\tilde{\theta}_0 = -\infty$ and $\tilde{\theta}_n = \infty$.
- **Goal:** Identify position i^* of a given $\tilde{\theta} \in \mathbb{R}$ s.t. $\tilde{\theta}_{i^*} < \tilde{\theta} < \tilde{\theta}_{i^*+1}$.
  - Query $\tilde{U}_k \triangleq \tilde{\theta}_j$ for some j.
  - Receive noisy response $\tilde{Y}_k = \mathbb{1}_{\{\tilde{U}_k < \tilde{\theta}\}} \oplus \tilde{Z}_k$, where $\tilde{Z}_k \sim \mathrm{Bern}(p)$.
  - After $\tilde{m}$ queries, compute estimate $\hat{i}^*$ of i^*.

Detour: Noisy Searching Problem

- Let $(\tilde{\theta}_0, \ldots, \tilde{\theta}_n) \in \bar{\mathbb{R}}^{n+1}$ be sorted with $\tilde{\theta}_0 = -\infty$ and $\tilde{\theta}_n = \infty$.

- **Goal:** Identify position i^* of a given $\tilde{\theta} \in \mathbb{R}$ s.t. $\tilde{\theta}_{i^*} < \tilde{\theta} < \tilde{\theta}_{i^*+1}$.
  - Query $\tilde{U}_k \triangleq \tilde{\theta}_j$ for some j.
  - Receive noisy response $\tilde{Y}_k = \mathbb{1}_{\{\tilde{U}_k < \tilde{\theta}\}} \oplus \tilde{Z}_k$, where $\tilde{Z}_k \sim \mathrm{Bern}(p)$.
  - After $\tilde{m}$ queries, compute estimate $\hat{i}^*$ of i^*.

- Burnashev-Zigangirov algorithm: searching using posteriors of intermediate intervals
  - Let $I_i = (\tilde{\theta}_{i-1}, \tilde{\theta}_i)$ for each $i \in [n]$.
  - Update posterior of each interval I_i

$$q_k(i) \triangleq \mathrm{P}\{\theta \in I_i \mid \tilde{Y}_1, \ldots, \tilde{Y}_k, \tilde{U}_1, \ldots, \tilde{U}_k\},$$

  for $k \in [\tilde{m}]$ and $i \in [n]$, where $q_0(i) = \frac{1}{n}$ for all i.

Detour: Noisy Searching Problem

- Let $(\tilde{\theta}_0, \ldots, \tilde{\theta}_n) \in \bar{\mathbb{R}}^{n+1}$ be sorted with $\tilde{\theta}_0 = -\infty$ and $\tilde{\theta}_n = \infty$.

- **Goal:** Identify position i^* of a given $\tilde{\theta} \in \mathbb{R}$ s.t. $\tilde{\theta}_{i^*} < \tilde{\theta} < \tilde{\theta}_{i^*+1}$.
  - Query $\tilde{U}_k \triangleq \tilde{\theta}_j$ for some j.
  - Receive noisy response $\tilde{Y}_k = \mathbb{1}_{\{\tilde{U}_k < \tilde{\theta}\}} \oplus \tilde{Z}_k$, where $\tilde{Z}_k \sim \mathrm{Bern}(p)$.
  - After $\tilde{m}$ queries, compute estimate $\hat{i}^*$ of i^*.

- Burnashev-Zigangirov algorithm: searching using posteriors of intermediate intervals
  - Let $I_i = (\tilde{\theta}_{i-1}, \tilde{\theta}_i)$ for each $i \in [n]$.
  - Update posterior of each interval I_i

    $$q_k(i) \triangleq \mathsf{P}\{\theta \in I_i \mid \tilde{Y}_1, \ldots, \tilde{Y}_k, \tilde{U}_1, \ldots, \tilde{U}_k\},$$

    for $k \in [\tilde{m}]$ and $i \in [n]$, where $q_0(i) = \frac{1}{n}$ for all i.
  - Let $j(k)$ be index that bisects the posteriors, i.e.,

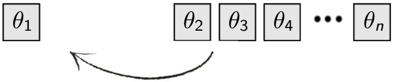
    $$\sum_{i=1}^{j(k)-1} q_k(i) \leq \frac{1}{2} \qquad \text{and} \qquad \sum_{i=1}^{j(k)} q_k(i) > \frac{1}{2}$$

  - Choose U_{k+1} randomly among $\{\tilde{\theta}_{j(k)-1}, \tilde{\theta}_{j(k)}\}$.
  - Repeat for $\tilde{m}$ queries, and output the posterior median index $\hat{i}^* \triangleq j(\tilde{m})$.
  - We have

    $$\mathsf{P}\{\tilde{\theta} \notin I_{\hat{i}^*}\} \leq (n-1)\left(\frac{1}{2} + \sqrt{p(1-p)}\right)^{\tilde{m}}$$

Back to Noisy Sorting Problem

- Can use BZ algorithm with insertion sort



Back to Noisy Sorting Problem

- Can use BZ algorithm with insertion sort

- Insert θ_2 in its estimated position w.r.t. θ_1 using BZ.

Back to Noisy Sorting Problem

- Can use BZ algorithm with insertion sort

- Insert θ_2 in its estimated position w.r.t. θ_1 using BZ.
- Insert θ_3 in its estimated position w.r.t. $\{\theta_1, \theta_2\}$ using BZ.

Back to Noisy Sorting Problem

- Can use BZ algorithm with insertion sort

- Insert θ_2 in its estimated position w.r.t. θ_1 using BZ.
- Insert θ_3 in its estimated position w.r.t. $\{\theta_1, \theta_2\}$ using BZ.
- Insert θ_4 in its estimated position w.r.t. $\{\theta_1, \theta_2, \theta_3\}$ using BZ.
- $\cdots$

Back to Noisy Sorting Problem

- Can use BZ algorithm with insertion sort

- Insert θ_2 in its estimated position w.r.t. θ_1 using BZ.
- Insert θ_3 in its estimated position w.r.t. $\{\theta_1, \theta_2\}$ using BZ.
- Insert θ_4 in its estimated position w.r.t. $\{\theta_1, \theta_2, \theta_3\}$ using BZ.
- $\cdots$
- It can be shown that

$$\mathsf{P}\{\hat{\pi} \neq \pi\} \leq n^2 \left(\frac{1}{2} + \sqrt{p(1-p)}\right)^{\frac{m}{n-1}}$$

$\Rightarrow$ any sorting rate $R < \frac{1}{2} - \frac{1}{2}\log\left(1 + 2\sqrt{p(1-p)}\right)$ is achievable.

Final Remarks

- Extensions:
  - Permutation search algorithms to fully characterize the noisy sorting capacity
  - Unknown p and/or query-dependent p
- Arxiv version: https://arxiv.org/abs/2202.01446
- Any questions?