Massachusetts Institute of Technology
Laboratory for Information & Decision Systems
Srikant T. Jagabathula
Devavrat Shah

Under Constrained Sensing
Learning Popular Ranking
Motivation

Three examples:

One. Elections
   - n candidates
   - Each voter has a preference list in mind

Goal:
   - Learn popular preferences/ranking through polls
Motivation

Three examples:

Two. Revenue maximization
   - n products
   - Each customer has a preference list in mind

Goal:
   - Learn popular preferences/ranking through polls
Learn popular rankings through results of games.

Three: Sports league

Examples:

Motivation
Basic questions:

- How to learn?
- Partial information:
  - How to learn?
  - When is it possible to learn?

Examples:

- Through partial data
- Learn popular rankings with sparse support
- Motivation
Outline

1. Types of partial information

2. Recovery results

3. Algorithm, recoverability condition
First order information: $n = 3$

Partial Information Types
First order information: \( n = 3 \)

Partial Information Indexes
First order information: $n = 3$

Partial Information guesses
First order information: \( n = 3 \)

Partial Information Trees
First order information: n = 3

Partial Information Presses
Partial Information Trees

Second order information: $n = 3$

$\rho (c_0 \rightarrow c_i \rightarrow c_j) \equiv \rho = [c_0 \rightarrow c_i \rightarrow c_j]$
Partial Information Types

First order information: $n=3$
Second order information: \( n = 3 \)

Parabolic Information Preserves
Examples:

\[ D_{\lambda} = \frac{!_{\lambda}!}{n!} \]

\( D_{\lambda} \) distinct ways to partition \( 1 \ldots n \) as per \( \lambda \).

Partition \( \lambda \) of \( n \):

\[ \lambda = (n-1, 1) \]

- \( \lambda = (n-2, 1, 1) \) : second order
- \( \lambda = (n-1, 1) \) : first order
- \( \lambda = (1, 1, 1, \ldots, 1) \) : complete information
\[
\{x \in \mathcal{P}(S) \mid f(x) = t\} = \bigcap_{(t)} \mathcal{P}(S)
\]

\[(\text{mapped to}) \]

\[\mathcal{O} = \mathcal{P}(\text{partition } t)\]

\[
\begin{bmatrix}
\mathcal{O} \\
\mathcal{O}^t
\end{bmatrix}
\]

- D×D, matrix \( O = [\mathcal{O} \ \mathcal{O}^t] \)

\(\chi\)-order partial information

\(X\) distinct ways to partition \(S_1 \ldots S_n\) as per \(X\)

- D

\(X = (x_1, x_2)\), \(x_1^2 \leq x_2, x_1 + x_2 = n\)

Partition \(X\) of \(n\)

Partial Information Types
\[ \{s_0 \mid s_0 \in \mathcal{I}(G), t(s_0) = t \} \subseteq \mathcal{I}(G) \]

\[ \mathcal{I}(G) \triangleq \{(\text{mapped } t), \text{(partition } t)\} \]

\[ \mathcal{O}^{t,t} = \mathcal{P}(\text{partition } t) \]

\[ [\mathcal{O}^{t,t}] = \mathcal{O}^{t,t} \]

- Order partial information:

\[ \mathcal{O} = \{ (x_1, \ldots, x_r) \mid x_1 \geq x_2 \geq \ldots \geq x_r, \sum x_i = n \} \]

\[ \text{Partition } \mathcal{O} \text{ of } n \]

Partial Information Types
\[ K = \{ g \in \mathbb{R}_+^n \mid \text{is small (certainty } \approx 0) \} \]

- Likely, when support \( \mathcal{P} \) is sparse, i.e.
  \( \mathbb{P}(g) \), \( g \in \mathbb{S}_n \), possibly with distinct values.

Task: Recover the distribution over permutations, i.e.

\[ D \times D \text{ matrix } \mathbf{Q} = [Q_{i,j}] \]

Given: \( x \) - order partial information

\[ x = (x_1, \ldots, x_r), \ 1 \leq x_1 < x_2 < \cdots < x_r \leq n \]

Partition \( x \) of \( n \):

Question
K = \{ g : \mathbb{P}(g) > 0 \} \text{ is small (certainty \leq \alpha^2) }

- Likely, when support \( \mathbb{P} \) is sparse, i.e.

- \( \mathbb{P}(g), \quad g \in \mathbb{S} \) ; possibly \#_G \text{ distinct values.}

\textbf{Task:} Recover the distribution over permutations, i.e.

- \( D \times D \) matrix \( \mathbb{O} = [\Theta_{ij}] \)

\textbf{Given:} \( x \)-order partial information

\[ x = (x_1, \ldots, x_n) \text{, } x_1 \geq x_2 \geq \ldots \geq x_n \text{, } x_1 + \ldots + x_n = n \]

\textbf{Question:}
\[
\begin{align*}
\text{\text{opt-\text{rmt}} & \equiv \min_{\|Z\|_0 = 1} \{ Z \in \mathbb{Z}^n : Z^c = 0 \text{ over } \mathbb{Z}^n \} \text{ subject to } A Z = y. \\
\end{align*}
\]

Occam's Razor: the sparest consistent solution.

columns correspond to action of permutation

\[ Y = A X \]

recover \( x \in \mathbb{R}^n \) from \( y \in \mathbb{R}^m \)

Question
Recent work on when to-opt provides correct answer.
Then $x$ can be recovered as long as $|x| > \frac{\pi}{2}$.

- For us, $A$ is given.

- Primary focus: design of $A$ with minimal means.

  Recent work on when $G_0$-opt provides correct answer

\[
\begin{align*}
A_2 &= \mathbf{y} \\
\text{subject to } &
\{ \begin{array}{l}
\min \|Z\|_0 = 1 \\
\text{over } Z \in \mathbb{R}_+^{m \times n}
\end{array} \}
\end{align*}
\]
Question

\[ \|z\|_0 - \text{opt} = \left\{ \begin{array}{l}
\min \|z\|_0 = \left| \{ Z_c : Z_c \neq 0 \} \right| \quad \text{over } Z \in \mathbb{R}^n_+ \\
\text{subject to } \quad Az = y
\end{array} \right. \]

Counter-example: lack of k-RIP, k = 4.
or solve \( g_0 \)-opt?

- Is there a simple algorithm to recover,
- How well does it perform?
- When does \( g_0 \)-opt recover distribution?

**Questions**

\[
\begin{align*}
A_2 &= \gamma, \\
\text{subject to } &\{ z_0 \} \text{ over } Z_0 \in \mathbb{R}^n \\
\end{align*}
\]

\( \in \) \( g_0 \)-opt

**Question**
- Set probabilities: \( p_i = \frac{q_i}{\sum q_i}, 1 \leq i \leq K \)
- Choose \( q_i \in [0,1] \) uniformly, \( 1 \leq i \leq K \)
- To each assign probabilities uniformly at random, e.g.
- Choose \( K \) permutations uniformly at random

Random model: is counter-example very special?

Results
With probability 1-\(o(1)\), up to what \(K\) can distribution be recovered.

Revised question: Given \(X\)-order information,

- Set probabilities: \(P_i = \frac{1}{\sqrt{K}}, i \in \mathbb{N}\).
- Choose \(q_i \in [0,1]\) uniformly, \(i \in \mathbb{N}\).
- To each assign probability at random, e.g.,
  - Choose \(K\) permutations uniformly at random.

Random model: is counter-example very special?

Results
Then, with probability $1 - o(1)$, Log^t has a unique solution. Further, a simple algorithm recovers it.

Theorem 1. $x = (n-1, 1)$. Let, $K > (1-\varepsilon)n \log n$ for any $\varepsilon > 0$. Results
Solutions with probability $1-o(1)$. The $L$-opt will admit multiple solutions. Proposition 2. $X = (n^{-1/2})$. The $L$-opt will have multiple solutions. Subject to \[
\begin{align*}
A_2 y &= y \\
\min \|z\|_1 &= z \geq 0 \quad \text{over } Z \in \mathbb{R}^n
\end{align*}
\]

$R$-opt $\approx \frac{1}{\log n}$.

Further, a simple algorithm recovers it. Then, with probability $1-o(1)$, $L$-opt has a unique solution. Theorem 1. $X = (n^{-1/2})$. Let, $K = (1-\varepsilon)n \log n$ for any $\varepsilon > 0$. Results
$- \frac{1}{2} \log_2 n \leq K - \frac{\log n}{\log(1 + \epsilon)} \leq 3 \log n$

Then, no algorithm can recover with probability $1-o(1)$, if $K \leq \frac{\log n}{\log(1 + \epsilon)}$.

Then, with probability $1-o(1)$, C2-opt has a unique solution. Further, a simple algorithm recovers it.

$\frac{\log n}{\log(1 + \epsilon)} \leq K \leq (1-\epsilon) \log n$ for any $\epsilon > 0$.

Theorem 2. $\lambda = (n-1, 1)$. Let, $K > (1-\epsilon) \log n$ for any $\epsilon > 0$. Results
$K \sim D^2 \log D$.

More generally, for $x = (x_1, \ldots, x_r)$ with $x_i = n-o(1)$,

Here $m$ is a constant.

$$K \geq \frac{m}{(1-\epsilon)}$$

for any $\epsilon > 0$. and our algorithm will recover distinguisher $\xi$ long as

Theorem 2. $x = (n-m, n-m, \ldots, n-m)$ with probability $1-o(1)$, $K_{OPT}$

Results
Results

Theorem 3. \( X \approx (A^{1/2} S^{1/2}) \cdot X = n - o(n^{2/3}) \) with prob. \( 1 - o(1) \), so opt.
\[
\left[ \frac{1}{1-x} \right] = H \quad : \quad \frac{H(1) - H(x)}{H(x)} = \log(1-x) = -x
\]

\[
H(x) = \frac{1}{x} \quad : \quad \log x = -x \log x
\]

Definition: \( f(x) \):

\[ x = x_0, \quad 1 \leq x \]

\[ K \geq D \]

\[ f(x) \]

Theorem 4. \( x = x_0 \): general case. With proof. (I-01), Log opt.

Results
That is, $\mathbb{K} \sim \mathbb{D}$.

Thus $1-3: \alpha \leftarrow \mathbb{I}$ can be shown that $\eta(\alpha) \sim \mathbb{I}$.

$$
\left[ \frac{1-x_i}{1} \right] = \mathbb{W} \quad : \quad \frac{H(x)}{(x, W, H, H)} \left( 1-0 \right) = \left( 1-0 \right) \mathbb{W} \mathbb{W}
$$

Thm 4: $x = \frac{1}{\ln \gamma}$, $H(x) = \frac{1}{\ln \gamma} - \frac{\ln \gamma}{\ln \gamma} - \ln \gamma$. Hence $\alpha \leftarrow \mathbb{I}$.

Theorem 4: $x = \frac{1}{\ln \gamma}$, $H(x) = \frac{1}{\ln \gamma} - \frac{\ln \gamma}{\ln \gamma}$.

\[ x \geq D \Rightarrow \exists \gamma \left( x \leftarrow \mathbb{I}(\gamma) \right) \]

Results
In compliance with Birthday Paradox: \( x = (1, \ldots, 1) \).

That is, \( \mathbb{E} \sum_{i} \mathbb{1}_{K \leq i} \sim \sqrt{K} \).

Other extreme: \( \alpha \to 0 \), can be shown that \( \xi(x) \sim \frac{1}{\alpha^2} \).

\[
\left[ 1 - \frac{1}{x} \right] = H_{\text{M}} \left( \frac{x}{M}, H(x) \right) \frac{x}{1 - 0} (1) = (1 - 0) \xi(x)
\]

Theorem 4: \( x = (\alpha, \ldots, \alpha) \) \( K \geq D \).

Results
Algorithm

3. Unique witness: for any $g$ with $P(g) > 0$, there

2. Linear independence: given $K$ permutations

1. With probability $P$, $i \in e K$

2. $c_i p_i \neq 0$ for any $c \in e K$, $i \in e K$
Algorithm: $\lambda = (n-1, 1)$
Algorithm : $x = (n-1, 1)$
Algorithm: $x = (n-1, 1)$
Algorithm: $x = (n-1, 1)$
Algorithm: $x = (n-1, 1)$
Algorithm: $X = (n-1, 1)$

$P_{(\theta^2)} = 0.2$

$P_{(\theta^3)} = 0.4$
Algorithm: \( X = (n-1,1) \)
Algorithm: $x = (n-1, 1)$

\[
\mathbb{P}(x_1) = 0.1 \\
\mathbb{P}(x_2) = 0.2 \\
\mathbb{P}(x_3) = 0.7
\]
Algorithm \( \text{\texttt{A}} = (n - 1, 1) \)
Algorithm: \( \mathcal{A} = (n-1, 1) \)
More work needs to be done (data?). Surprisingly well for learning customer choices.

A natural optimization formulation seems to work robustly.

Robustness type of partial information.

And, recoverability gracefully depends on the when their support is reasonably sparse.

It is possible to recover popular rankings.

Discussion
Comparison plus favorable: $K \sim \log n$

Comparison: $K \sim \log n$

Other types of data: e.g.

Type of partial information. And, recoverability gracefully depends on the
when their support is reasonably sparse.

It is possible to recover popular rankings.

Discussion
Exciting set of questions:

- And, lots of applications.
- Type of partial information.
- And, recoverability gracefully depends on the.
- When their support is reasonably sparse.
- It is possible to recover popular rankings.

Discussion