Statistical inference of a ranked community in a directed graph

Tim Kunisky
(joint with Dan Spielman, Alex Wein, and Xifan Yu)

Department of Applied Mathematics & Statistics, Johns Hopkins University

INFORMS APS: Finding Hidden Structure in Networks
June 30, 2025

Motivation: Planted Dense Subgraph

Motivation: Planted Dense Subgraph

Widely studied random graph $G \sim \mathcal{G}(n, p, q, k)$ on $[n]$:

1. Choose $S \subseteq [n]$ with $|S| = k$ uniformly at random.
2. Set $i \sim j$ independently with probability

$$\left\{ \begin{array}{ll} q > p & \text{if } i, j \in S \\ p & \text{otherwise} \end{array} \right\}.$$

Motivation: Planted Dense Subgraph

Widely studied random graph $G \sim \mathcal{G}(n, p, q, k)$ on $[n]$:

1. Choose $S \subseteq [n]$ with $|S| = k$ uniformly at random.
2. Set $i \sim j$ independently with probability

$$\begin{Bmatrix} q > p & \text{if } i, j \in S \\ p & \text{otherwise} \end{Bmatrix}.$$

Natural family of scalings: $q = n^{-\alpha}$, $k = n^{\beta}$, $p = n^{-\gamma}$ for $\alpha, \beta, \gamma \in (0, 1)$ (**"log-density"** setting).

Motivation: Planted Dense Subgraph

Widely studied random graph $G \sim \mathcal{G}(n, p, q, k)$ on $[n]$:

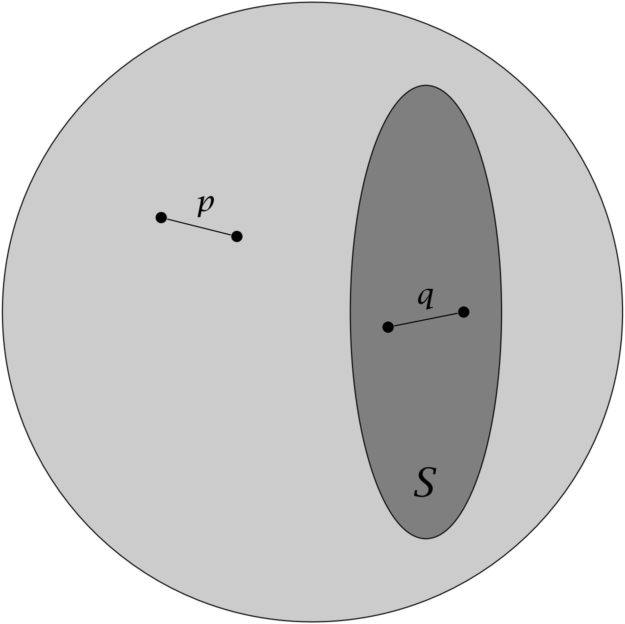
1. Choose $S \subseteq [n]$ with $|S| = k$ uniformly at random.
2. Set $i \sim j$ independently with probability

$$\left\{ \begin{array}{ll} q > p & \text{if } i, j \in S \\ p & \text{otherwise} \end{array} \right\}.$$

Natural family of scalings: $q = n^{-\alpha}$, $k = n^{\beta}$, $p = n^{-\gamma}$ for $\alpha, \beta, \gamma \in (0, 1)$ (**"log-density"** setting).

Statistical questions:

- **Detection:** Test $G \sim \mathcal{G}(n, p, q, 0)$ vs. $G \sim \mathcal{G}(n, p, q, k)$?
- **Recovery:** When $G \sim \mathcal{G}(n, p, q, k)$, recover S?


p
q
S

Formulation of Statistical Questions

Formulation of Statistical Questions

(Strong) detection: find **tests** $t_n : \{\text{graphs}\} \to \{0,1\}$ with

$$\mathop{\mathbb{P}}_{G \sim \mathcal{G}(n,p,q,k)}[t_n(G) = 1] = \mathop{\mathbb{P}}_{G \sim \mathcal{G}(n,p,q,0)}[t_n(G) = 0] = 1 - o(1),$$

or **separating statistics** $f_n : \{\text{graphs}\} \to \mathbb{R}$ with

$$\mathop{\mathbb{E}}_{G \sim \mathcal{G}(n,p,q,k)}[f_n(G)] - \mathop{\mathbb{E}}_{G \sim \mathcal{G}(n,p,q,0)}[f_n(G)] \gg \sqrt{\mathop{\mathrm{Var}}_{G \sim \mathcal{G}(n,p,q,k)}[f_n(G)]} + \sqrt{\mathop{\mathrm{Var}}_{G \sim \mathcal{G}(n,p,q,0)}[f_n(G)]}.$$

Formulation of Statistical Questions

(Strong) detection: find **tests** $t_n : \{\text{graphs}\} \to \{0,1\}$ with

$$\mathop{\mathbb{P}}_{G \sim \mathcal{G}(n,p,q,k)}[t_n(G) = 1] = \mathop{\mathbb{P}}_{G \sim \mathcal{G}(n,p,q,0)}[t_n(G) = 0] = 1 - o(1),$$

or **separating statistics** $f_n : \{\text{graphs}\} \to \mathbb{R}$ with

$$\mathop{\mathbb{E}}_{G \sim \mathcal{G}(n,p,q,k)}[f_n(G)] - \mathop{\mathbb{E}}_{G \sim \mathcal{G}(n,p,q,0)}[f_n(G)] \gg \sqrt{\mathop{\mathsf{Var}}_{G \sim \mathcal{G}(n,p,q,k)}[f_n(G)]} + \sqrt{\mathop{\mathsf{Var}}_{G \sim \mathcal{G}(n,p,q,0)}[f_n(G)]}.$$

Recovery: find **estimators** $\widehat{S} : \{\text{graphs}\} \to \binom{[n]}{k}$ with

$$\mathop{\mathbb{E}}_{(S,G) \sim \mathcal{G}(n,p,q,k)} \frac{|S \triangle \widehat{S}(G)|}{k} = \begin{cases} 1 - \Omega(1) & \text{(weak)} \\ o(1) & \text{(strong)} \end{cases}$$

Main Phenomena

Main Phenomena

- **Statistical-computational gap:** For some parameter choices, detection and/or recovery are **possible** but (conjecturally) **computationally hard**.

  For example, sometimes can detect by **brute-force computation** of likelihood ratio in time $\approx n^k = n^{n^\beta}$.

Main Phenomena

- **Statistical-computational gap:** For some parameter choices, detection and/or recovery are **possible** but (conjecturally) **computationally hard**.

  For example, sometimes can detect by **brute-force computation** of likelihood ratio in time $\approx n^k = n^{n^\beta}$.

- **Detection-recovery gap:** For some parameter choices, detection is easy while recovery is hard or impossible.

  For example, sometimes can detect subgraph merely by **counting total edges**, but this alone does not help to recover S.

[Bhaskara et al ’10; Hajek, Wu, Xu ’15; Verzelen, Arias-Castro ’15; Ma, Wu ’16; Chen, Xu ’16; Brennan, Bresler, Huleihel ’18; Schramm, Wein ’22; Dhawan, Mao, Wein ’23]

Question: What is the analog of this story over **directed graphs**?

Planted Ranked Subgraph

Planted Ranked Subgraph

Generate **oriented directed graph** (without $\begin{tikzcd}\bullet \arrow[r, bend left] & \bullet \arrow[l, bend left]\end{tikzcd}$):

Planted Ranked Subgraph

Generate **oriented directed graph** (without $\bullet \rightleftarrows \bullet$):

1. Sample $S \subseteq [n]$ with $|S| = k$ uniformly at random.
2. Sample π permutation of S uniformly at random.

Planted Ranked Subgraph

Generate **oriented directed graph** (without $\bullet \rightleftarrows \bullet$):

1. Sample $S \subseteq [n]$ with $|S| = k$ uniformly at random.
2. Sample π permutation of S uniformly at random.
3. If $i, j \in S$ and $\pi(i) < \pi(j)$, then set

$$\left\{ \begin{array}{ll} i \longrightarrow j & \text{with probability } \left(\frac{1}{2} + q\right) p \\ i \longleftarrow j & \text{with probability } \left(\frac{1}{2} - q\right) p \\ \text{no edge} & \text{with probability } 1 - p \end{array} \right\};$$

Planted Ranked Subgraph

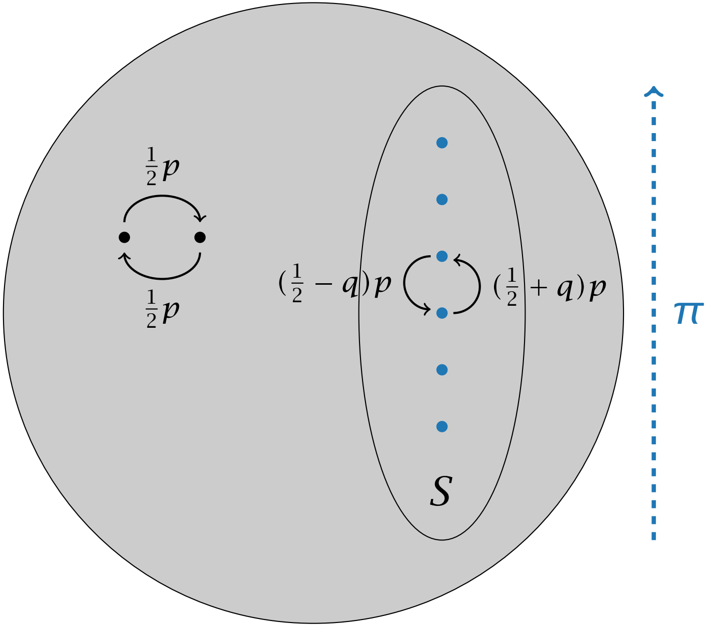
Generate **oriented directed graph** (without $\bullet \rightleftarrows \bullet$):

1. Sample $S \subseteq [n]$ with $|S| = k$ uniformly at random.
2. Sample π permutation of S uniformly at random.
3. If $i, j \in S$ and $\pi(i) < \pi(j)$, then set

$$\left\{ \begin{array}{ll} i \longrightarrow j & \text{with probability } \left(\frac{1}{2} + q\right) p \\ i \longleftarrow j & \text{with probability } \left(\frac{1}{2} - q\right) p \\ \text{no edge} & \text{with probability } 1 - p \end{array} \right\};$$

otherwise, set

$$\left\{ \begin{array}{ll} i \longrightarrow j & \text{with probability } \frac{1}{2} p \\ i \longleftarrow j & \text{with probability } \frac{1}{2} p \\ \text{no edge} & \text{with probability } 1 - p \end{array} \right\}.$$


$\frac{1}{2}p$
$\frac{1}{2}p$
$(\frac{1}{2}-q)p$
$(\frac{1}{2}+q)p$
S
π

Related Models

Some literature considers full **hidden permutations** or **rankings**, i.e., the case $k = n$. Common settings:

Related Models

Some literature considers full **hidden permutations** or **rankings**, i.e., the case $k = n$. Common settings:

- **Bradley-Terry-Luce (BTL):** Observe $i > j$ with probability $\frac{x_i}{x_i+x_j}$ for a hidden "preference vector" $(x_1, \ldots, x_n) \in \mathbb{R}^n_{>0}$. Often studied in terms of query complexity of learning preferences.

Related Models

Some literature considers full **hidden permutations** or **rankings**, i.e., the case $k = n$. Common settings:

- **Bradley-Terry-Luce (BTL):** Observe $i > j$ with probability $\frac{x_i}{x_i + x_j}$ for a hidden "preference vector" $(x_1, \ldots, x_n) \in \mathbb{R}^n_{>0}$. Often studied in terms of query complexity of learning preferences.
- **Noisy sorting:** Closer to our model—simultaneously observe all pairwise comparisons ($k = n$ and $p = 1$) once, each correct with probability $\frac{1}{2} + q$.

Related Models

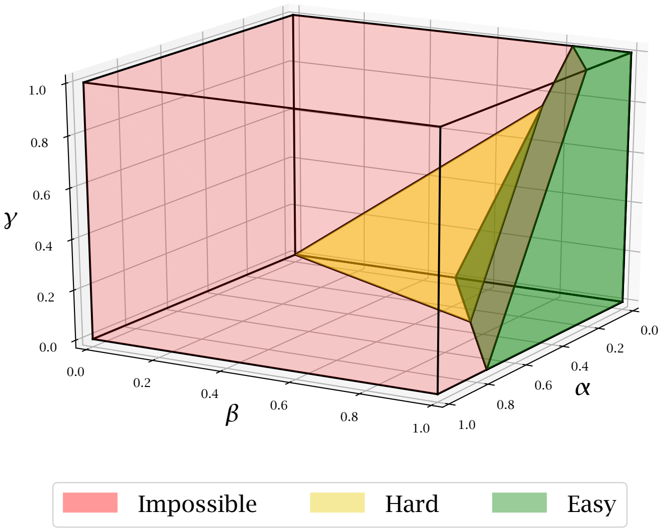
Some literature considers full **hidden permutations** or **rankings**, i.e., the case $k = n$. Common settings:

- **Bradley-Terry-Luce (BTL):** Observe $i > j$ with probability $\frac{x_i}{x_i + x_j}$ for a hidden "preference vector" $(x_1, \ldots, x_n) \in \mathbb{R}^n_{>0}$. Often studied in terms of query complexity of learning preferences.

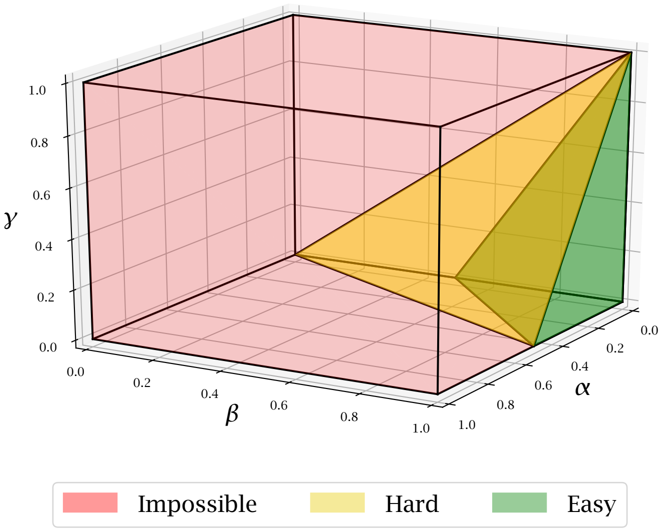
- **Noisy sorting:** Closer to our model—simultaneously observe all pairwise comparisons ($k = n$ and $p = 1$) once, each correct with probability $\frac{1}{2} + q$.

Key difference: possibility of **community structure**, i.e., of having $k < n$.

Main Results: Detection



Main Results: Recovery



The same main phenomena still happen (indeed are a little simpler!) when the community structure is carried entirely by the **edge direction** rather than edge density.

Algebraic Encoding

Standard perspective for graph setting: encode G into $\{0, 1\}$-valued adjacency matrix $\boldsymbol{Y} = \boldsymbol{Y}^\top$,

$$Y_{ij} := \left\{ \begin{array}{rl} +1 & \text{if } i \sim j \\ 0 & \text{if } i \not\sim j \text{ or } i = j \end{array} \right\}.$$

Algebraic Encoding

Standard perspective for graph setting: encode G into $\{0, 1\}$-valued adjacency matrix $\boldsymbol{Y} = \boldsymbol{Y}^\top$,

$$Y_{ij} := \left\{ \begin{array}{rl} +1 & \text{if } i \sim j \\ 0 & \text{if } i \not\sim j \text{ or } i = j \end{array} \right\}.$$

For oriented directed graph G, natural to encode into $\{0, +1, -1\}$-valued $\boldsymbol{Y} = -\boldsymbol{Y}^\top$,

$$Y_{ij} := \left\{ \begin{array}{rl} +1 & \text{if } i \longrightarrow j \\ -1 & \text{if } i \longleftarrow j \\ 0 & \text{if no edge or } i = j \end{array} \right\}.$$

Problems may then be viewed over $\{0, +1, -1\}^{n \times n}_{\mathsf{antisym}}$.

Lower Bound Idea (Briefly)

Form of hardness results: no sequence of polynomials of **low degree** $\lesssim (\log n)^{1+\varepsilon}$ can achieve detection (separation) or recovery (of indicator of support S).

Analysis follows standard toolkit using orthogonal polynomials.

Lower Bound Idea (Briefly)

Form of hardness results: no sequence of polynomials of **low degree** $\lesssim (\log n)^{1+\varepsilon}$ can achieve detection (separation) or recovery (of indicator of support S).

Analysis follows standard toolkit using orthogonal polynomials.

Mild complications: orthogonal polynomials must be taken over null model, "oriented directed Erdős-Rényi graph," whose (adjacency matrix) entries are **sparse Rademacher**:

$$\left\{ \begin{array}{rl} +1 & \text{with probability } \frac{1}{2}p \\ -1 & \text{with probability } \frac{1}{2}p \\ 0 & \text{with probability } 1-p \end{array} \right\}.$$

Algorithmic Idea 1: Directed PCA

For planted dense subgraph and clique, powerful algorithmic idea is to use the top eigenvalue/vector of $\boldsymbol{Y}$.

Algorithmic Idea 1: Directed PCA

For planted dense subgraph and clique, powerful algorithmic idea is to use the top eigenvalue/vector of $\boldsymbol{Y}$.

To deal with antisymmetric $\boldsymbol{Y} = -\boldsymbol{Y}^\top$, consider

$$\widetilde{\boldsymbol{Y}} := i\boldsymbol{Y}.$$

Algorithmic Idea 1: Directed PCA

For planted dense subgraph and clique, powerful algorithmic idea is to use the top eigenvalue/vector of $\boldsymbol{Y}$.

To deal with antisymmetric $\boldsymbol{Y} = -\boldsymbol{Y}^\top$, consider

$$\widetilde{\boldsymbol{Y}} := i\boldsymbol{Y}.$$

$\widetilde{\boldsymbol{Y}}$ is **Hermitian** and behaves like $\mathbb{C}$-valued spiked matrix model. More complicated "signal part," looking like:

$$\mathbb{E}[\widetilde{\boldsymbol{Y}} \mid S, \pi] = \begin{bmatrix} 0 & +i & \cdots & +i \\ -i & 0 & \cdots & +i \\ \vdots & \vdots & \ddots & \vdots \\ -i & -i & \cdots & 0 \end{bmatrix}, \quad \text{rank} = k \sim n^\beta$$

Algorithmic Idea 2: Win Counts

For planted dense subgraph and clique, simple but sometimes optimal idea is to count all edges (1-stars).

Algorithmic Idea 2: Win Counts

For planted dense subgraph and clique, simple but sometimes optimal idea is to count all edges (1-stars).

We have calibrated edge count to not carry information, but can consider **win counts** (degrees) of vertices:

$$W_i := \sum_{j \neq i} Y_{ij} = \#\{\text{wins}\} - \#\{\text{losses}\}.$$

Algorithmic Idea 2: Win Counts

For planted dense subgraph and clique, simple but sometimes optimal idea is to count all edges (1-stars).

We have calibrated edge count to not carry information, but can consider **win counts** (degrees) of vertices:

$$W_i := \sum_{j \neq i} Y_{ij} = \#\{\text{wins}\} - \#\{\text{losses}\}.$$

For detection, thresholding the **sample variance** of $W_1, \dots, W_n$ achieves the optimal threshold. Using a spectral algorithm for detection is sometimes suboptimal!

Algorithmic Idea 2: Win Counts

For planted dense subgraph and clique, simple but sometimes optimal idea is to count all edges (1-stars).

We have calibrated edge count to not carry information, but can consider **win counts** (degrees) of vertices:

$$W_i := \sum_{j \neq i} Y_{ij} = \#\{\text{wins}\} - \#\{\text{losses}\}.$$

For detection, thresholding the **sample variance** of $W_1, \dots, W_n$ achieves the optimal threshold. Using a spectral algorithm for detection is sometimes suboptimal!

For recovery, when $k = n$, then **ranking by wins** (i.e., by the W_i scores) achieves the optimal threshold.

Future directions: What does your favorite graph problem (cut, coloring, clique, independent set) or model (stochastic block model, random geometric graph) look like over directed graphs?

Thank you!