Unbalanced multisection in the stochastic block model

Amelia Perry (MIT)

Joint work with Alex Wein (MIT)

3 July 2017

Community detection in networks

Goal: find subsets of more tightly interconnected nodes in a graph

Community detection in networks

Goal: find subsets of more tightly interconnected nodes in a graph

- e.g. finding social communities in Facebook

Community detection in networks

Goal: find subsets of more tightly interconnected nodes in a graph

- e.g. finding social communities in Facebook
- e.g. finding biological subsystems in protein–protein interaction nets

Community detection in networks

Goal: find subsets of more tightly interconnected nodes in a graph

- e.g. finding social communities in Facebook
- e.g. finding biological subsystems in protein–protein interaction nets
- In general, the analogue of clustering for network data.

Community detection in networks

Goal: find subsets of more tightly interconnected nodes in a graph

- e.g. finding social communities in Facebook
- e.g. finding biological subsystems in protein–protein interaction nets
- In general, the analogue of clustering for network data.

This talk: a robust, statistically strong, poly-time algorithm to find communities.

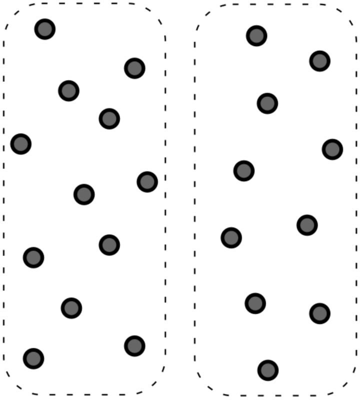
Stochastic block model (Holland et al. 1983)

Generative model for graphs with community structure,
studied in statistics, information theory, computer science, statistical physics. . .

Stochastic block model (Holland et al. 1983)

Generative model for graphs with community structure,
studied in statistics, information theory, computer science, statistical physics. . .

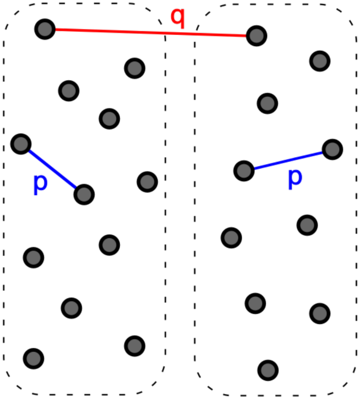
- n vertices partitioned into k 'communities'



Stochastic block model (Holland et al. 1983)

Generative model for graphs with community structure, studied in statistics, information theory, computer science, statistical physics. . .

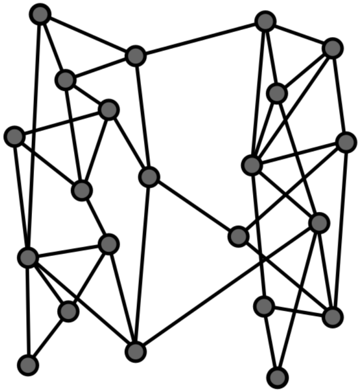
- n vertices partitioned into k 'communities'
- Edges drawn independently with probabilities $p > q$



Stochastic block model (Holland et al. 1983)

Generative model for graphs with community structure, studied in statistics, information theory, computer science, statistical physics. . .

- n vertices partitioned into k 'communities'
- Edges drawn independently with probabilities $p > q$



Goal: recover the planted communities (exactly or approximately)

Many algorithmic approaches

- Girvan–Newman modularity [GN02]

Many algorithmic approaches

- Girvan–Newman modularity [GN02]
- Spectral clustering [Bop87, McS01, RCY11, AS15]

Many algorithmic approaches

- Girvan–Newman modularity [GN02]
- Spectral clustering [Bop87, McS01, RCY11, AS15]
- Belief propagation [DMKZ11, MNS14]

Many algorithmic approaches

- Girvan–Newman modularity [GN02]
- Spectral clustering [Bop87, McS01, RCY11, AS15]
- Belief propagation [DMKZ11, MNS14]
- Semidefinite programming [GW94, ABH14, HWX15, ABKK15]

Sharp thresholds

n — number of vertices
k — number of communities
p — internal edge probability
q — external edge probability

Denser regime

$p = a \log n / n$, $q = b \log n / n$.

Sharp thresholds

n — number of vertices
k — number of communities
p — internal edge probability
q — external edge probability

Denser regime

$p = a \log n/n$, $q = b \log n/n$.
Average degree is order $\log n$.

Sharp thresholds

n — number of vertices
k — number of communities
p — internal edge probability
q — external edge probability

Denser regime

$p = a \log n/n$, $q = b \log n/n$.
Average degree is order $\log n$.

Theorem [ABH14, MNS14, HWX15] With equal-size communities, exact recovery is possible iff

$$\sqrt{a} - \sqrt{b} \geq \sqrt{k}.$$

Sharp thresholds

n — number of vertices
k — number of communities
p — internal edge probability
q — external edge probability

Denser regime

$p = a \log n/n$, $q = b \log n/n$.
Average degree is order $\log n$.

Theorem [ABH14, MNS14, HWX15] With equal-size communities, exact recovery is possible iff

$$\sqrt{a} - \sqrt{b} \geq \sqrt{k}.$$

Sparser regime

Sharp thresholds

n — number of vertices
k — number of communities
p — internal edge probability
q — external edge probability

Denser regime

$p = a \log n/n$, $q = b \log n/n$.
Average degree is order $\log n$.

Theorem [ABH14, MNS14, HWX15] With equal-size communities, exact recovery is possible iff

$$\sqrt{a} - \sqrt{b} \geq \sqrt{k}.$$

Sparser regime

$p = a/n$, $q = b/n$.

Sharp thresholds

n — number of vertices
k — number of communities
p — internal edge probability
q — external edge probability

Denser regime

$p = a \log n/n$, $q = b \log n/n$.
Average degree is order $\log n$.

Theorem [ABH14, MNS14, HWX15] With equal-size communities, exact recovery is possible iff

$$\sqrt{a} - \sqrt{b} \geq \sqrt{k}.$$

Sparser regime

$p = a/n$, $q = b/n$.
Average degree is constant.

Sharp thresholds

n — number of vertices
k — number of communities
p — internal edge probability
q — external edge probability

Denser regime

$p = a \log n/n$, $q = b \log n/n$.
Average degree is order $\log n$.

Theorem [ABH14, MNS14, HWX15] With equal-size communities, exact recovery is possible iff

$$\sqrt{a} - \sqrt{b} \geq \sqrt{k}.$$

Sparser regime

$p = a/n$, $q = b/n$.
Average degree is constant.

Theorem [ABH14, MNS14, HWX15] With **two** equal-size communities, partial recovery is possible iff

$$(a - b)^2 > 2(a + b).$$

Robustness to model misspecification

- Many papers prove that with high probability, some chosen algorithm recovers the communities.

Robustness to model misspecification

- Many papers prove that with high probability, some chosen algorithm recovers the communities.
- But many of these algorithms fail on real-world data, and even on tiny modifications of the SBM!

Robustness to model misspecification

- Many papers prove that with high probability, some chosen algorithm recovers the communities.
- But many of these algorithms fail on real-world data, and even on tiny modifications of the SBM!
- [RJM16]: planting just a few tetrahedra into a large SBM graph causes spectral clustering to fail.

Robustness to model misspecification

- Many papers prove that with high probability, some chosen algorithm recovers the communities.
- But many of these algorithms fail on real-world data, and even on tiny modifications of the SBM!
- [RJM16]: planting just a few tetrahedra into a large SBM graph causes spectral clustering to fail.
- Q: How do we design algorithms that transfer to e.g. power-law graphs and many other models? Algorithms that are robust to model misspecification?

Semirandom models

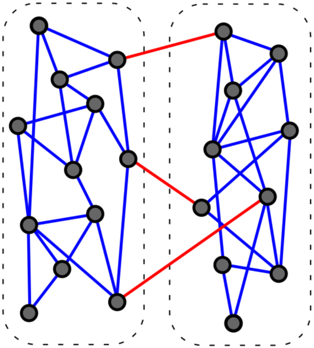
Models between average-case (generative) and worst-case.

Semirandom models

Models between average-case (generative) and worst-case.

For SBM: [FK00]

- Draw a graph from the usual SBM

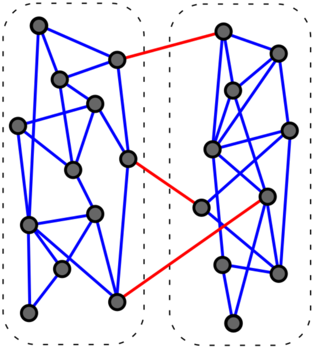


Semirandom models

Models between average-case (generative) and worst-case.

For SBM: [FK00]

- Draw a graph from the usual SBM
- An adversary may perform any number of **monotone** ('helpful') changes:
  - add edges within communities
  - remove edges within communities

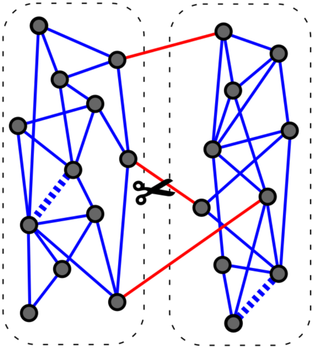


Semirandom models

Models between average-case (generative) and worst-case.

For SBM: [FK00]

- Draw a graph from the usual SBM
- An adversary may perform any number of **monotone** ('helpful') changes:
  - add edges within communities
  - remove edges within communities

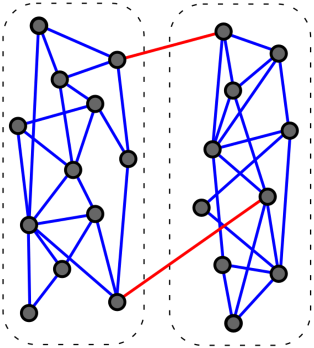


Semirandom models

Models between average-case (generative) and worst-case.

For SBM: [FK00]

- Draw a graph from the usual SBM
- An adversary may perform any number of **monotone** ('helpful') changes:
    - add edges within communities
    - remove edges within communities

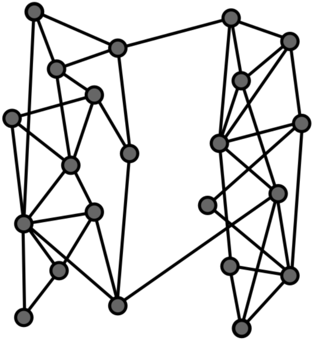


Semirandom models

Models between average-case (generative) and worst-case.

For SBM: [FK00]

- Draw a graph from the usual SBM
- An adversary may perform any number of **monotone** (‘helpful’) changes:
  - add edges within communities
  - remove edges within communities

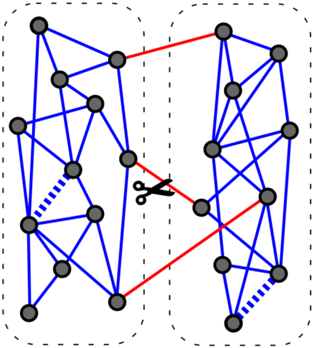


Semirandom models

Models between average-case (generative) and worst-case.

For SBM: [FK00]

- Draw a graph from the usual SBM
- An adversary may perform any number of **monotone** ('helpful') changes:
  - add edges within communities
  - remove edges within communities
- Prevents algorithms from over-tuning to specific model statistics (degree distribution, spectrum, etc.)



SDPs are robust [FK00]

Convex programs such as SDPs tend to be robust to monotone changes: if they succeded before the changes then they still succeed afterward.

SDPs are robust [FK00]

Convex programs such as SDPs tend to be robust to monotone changes: if they succeded before the changes then they still succeed afterward.

Reason: under a monotone change, the true SDP solution increases in value more than any other feasible solution.

SDPs are robust [FK00]

Convex programs such as SDPs tend to be robust to monotone changes: if they succeded before the changes then they still succeed afterward.

Reason: under a monotone change, the true SDP solution increases in value more than any other feasible solution.

If the true solution was optimal before the change, it remains optimal afterward.

Our convex program

$$
\begin{aligned}
\max \quad & \langle A, X \rangle - \omega \langle \mathbf{1}\mathbf{1}^\top, X \rangle \\
\text{s.t.} \quad & \forall u \quad X_{uu} = 1, \\
& \forall u, v \quad X_{uv} \geq \frac{-1}{k-1}, \\
& X \succeq 0.
\end{aligned}
$$

Our convex program

$$\begin{aligned} \max \quad & \langle A, X\rangle - \omega\langle \mathbf{1}\mathbf{1}^\top, X\rangle \\ \text{s.t.} \quad & \forall u \quad X_{uu} = 1, \\ & \forall u,v \quad X_{uv} \geq \frac{-1}{k-1}, \\ & X \succeq 0. \end{aligned}$$

- First suggested for the SBM in [ABKK15], based on classical multicut SDPs [FJ94].

Our convex program

$$\begin{aligned} \max \quad & \langle A, X\rangle - \omega\langle \mathbf{1}\mathbf{1}^\top, X\rangle \\ \text{s.t.} \quad & \forall u \quad X_{uu} = 1, \\ & \forall u, v \quad X_{uv} \geq \frac{-1}{k-1}, \\ & X \succeq 0. \end{aligned}$$

- First suggested for the SBM in [ABKK15], based on classical multicut SDPs [FJ94].
- We analyze this program on the SBM.

Our convex program

$$
\begin{aligned}
\max \quad & \langle A, X \rangle - \omega \langle \mathbf{1}\mathbf{1}^\top, X \rangle \\
\text{s.t.} \quad & \forall u \quad X_{uu} = 1, \\
& \forall u, v \quad X_{uv} \geq \frac{-1}{k-1}, \\
& X \succeq 0.
\end{aligned}
$$

- First suggested for the SBM in [ABKK15], based on classical multicut SDPs [FJ94].
- We analyze this program on the SBM.
- Allows k communities of different sizes.

Our convex program

$$\begin{aligned} \max \quad & \langle A, X\rangle - \omega\langle \mathbf{1}\mathbf{1}^\top, X\rangle \\ \text{s.t.} \quad & \forall u \quad X_{uu} = 1, \\ & \forall u, v \quad X_{uv} \geq \frac{-1}{k-1}, \\ & X \succeq 0. \end{aligned}$$

- First suggested for the SBM in [ABKK15], based on classical multicut SDPs [FJ94].
- We analyze this program on the SBM.
- Allows k communities of different sizes.
- The hyperparameter ω is necessary for a robust algorithm.

Main theorem

Theorem: our SDP achieves exact recovery w.h.p., and is robust to monotone changes, for all a, b, k and all community proportions for which the problem is statistically possible as $n \to \infty$.

Main theorem

Theorem: our SDP achieves exact recovery w.h.p., and is robust to monotone changes, for all a, b, k and all community proportions for which the problem is statistically possible as $n \to \infty$.

Depends on roughly the right choice of $\omega \approx \frac{a-b}{\log a - \log b} \frac{\log n}{n}$.

Conclusion

- When tuning heavily to specific generative models, we should care about robustness to model misspecification.

Conclusion

- When tuning heavily to specific generative models, we should care about robustness to model misspecification.
- Robustness is achievable: convex programming.

Conclusion

- When tuning heavily to specific generative models, we should care about robustness to model misspecification.
- Robustness is achievable: convex programming.
- Open: can we find cheaper algorithms with similar robustness guarantees?

Conclusion

- When tuning heavily to specific generative models, we should care about robustness to model misspecification.
- Robustness is achievable: convex programming.
- Open: can we find cheaper algorithms with similar robustness guarantees?

Thanks for listening! Any questions?