Modeling Hypergraph Data with Diversity

Xianshi Yu

Department of Computer Science
University of Wisconsin-Madison

July 20, 2023
StatsNet Seminar, São Carlos

Overview of my research

Network Data Analysis | Signal Processing | EHR Data

- Network community detection
- Collaborative filtering using social networks
- Hypergraph data analysis

Yu, X. & Zhu, J. (2023+). Modeling Hypergraphs with Diversity and Heterogeneous Popularity. Major revision at *JASA*.

Hypergraph data—examples

Hypergraph data characterize ‘multi-actor’ relations

Collaborations

paper 1-- Kulesza, A. & Taskar, B. (2012)

paper 2-- Brunel, V.-E., Moitra, A., Rigollet, P., & Urschel, J. (2017)

...

paper 8-- Gartrell, M., Paquet, U., & Koenigstein, N. (2017)

Shopping orders

order 1-- scissors, pencil, cheese, spinach

order 2-- tape, tissues, lemons

...

order 9-- pork, vitamins, pan, lock, brush

Medical codes in electronic health records (EHR)

patient visit 1-- J44.9, J45.9, B44.9, O60, L50.5

patient visit 2-- J45.9, J46, O60, L50.5

...

patient visit 7-- Z11.52, Z20.822, Z86.16, U07.1, J12.82

Ingredients in cooking recipes

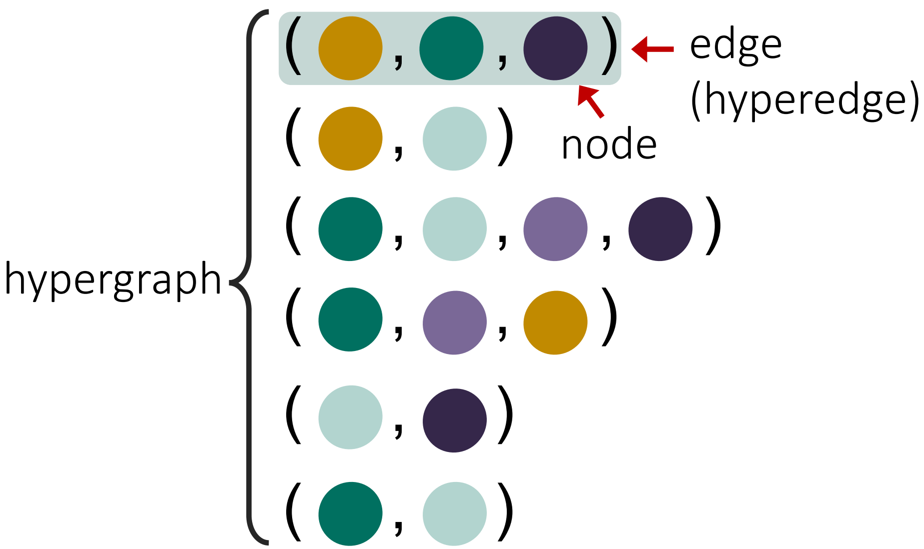
recipe 1-- 🍞, 🥬, 🥓, 🧀, 🍅

recipe 2-- 🍅, 🥔, 🐮, 🧂

...

recipe 8-- 🥒, 🥜, 🐔, 🌶️, 🧂, 🧅

Hypergraph data—concepts



A hypergraph is a collection of sets.

Electronic health records (EHR)

J44.9, J45.9, B44.9, O60, L50.5 ← edge = patient visit

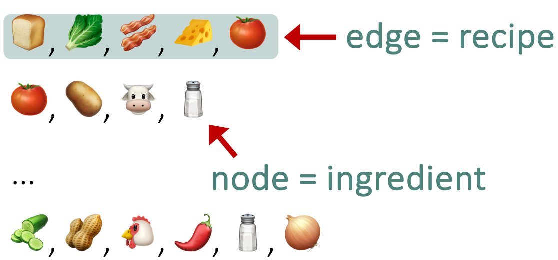
J45.9, J46, O60, L50.5

node = medical code

…

Z11.52, Z20.822, Z86.16, U07.1, J12.82

Cooking recipes



Hypergraph data – what can we learn from it?

Row = edge = recipe

Node = ingredient

recipe 1 (🍞, 🥬, 🥓, 🧀, 🍅)

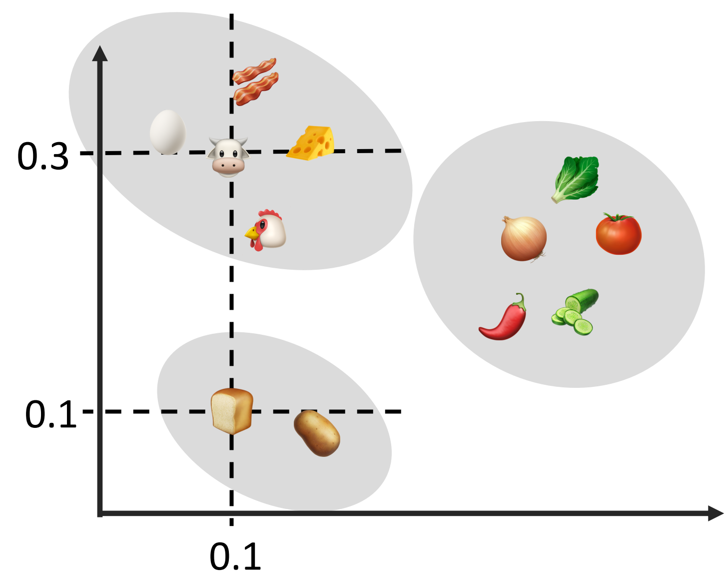
recipe 2 (🍅, 🥔, 🐮)

recipe 3 (🥒, 🥜, 🐔, 🌶️, 🧅)

recipe 4 (🌶️, 🐔, 🧄)

recipe 5 (🍅, 🥚)

2-D embedding



- Node embedding, e.g., 🐮=(0.1,0.3), 🍞=(0.1, 0.1)
  enables clustering

Hypergraph data – what can we learn from it?

Row = edge = patient visit

Node = medical code

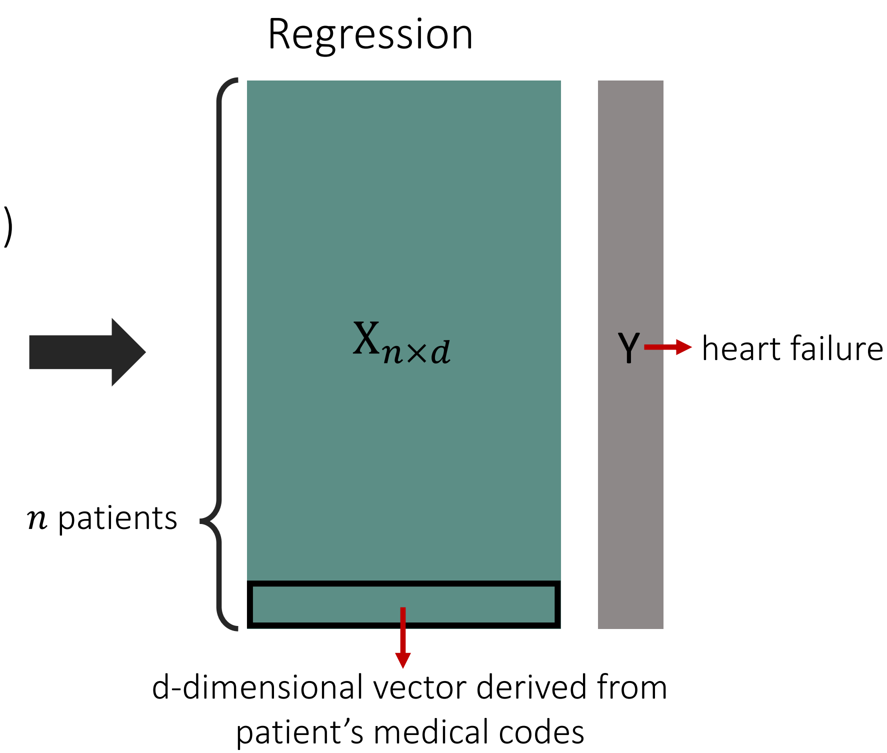
visit 1 (J44.9, J45.9, B44.9, O60, L50.5)

visit 2 (J45.9, J46, O60, L50.5)

visit 3 (Z11.52, Z20.822)

visit 4 (Z86.16, U07.1, J12.82)

visit 5 (O60, L50.5)



- Node embedding

  enables **clustering**, regression, can preserve privacy
- Edge prediction, e.g., (🐮, 🌶️, 🧄, ?)

Gap in literature on hypergraph modeling

In practice, hypergraph data are often projected to a network (i.e., pairwise relations), causing information loss. Direct modeling of hypergraph data is a relatively open area.

e.g., to allow recipes to have different numbers of ingredients

e.g., to allow the same set of medical codes be observed in multiple patient visits

	Edges with varying cardinality	Edge multiplicity	Model characteristic
Ghoshdastidar and Dukkipati (2014, 2015, 2017b); Chien et al. (2018); Kim et al. (2018); Ke et al. (2019)			Clustering of nodes
Lyu et al. (2021); Yuan and Qu (2021)			**Latent space model**
Stasi et al. (2014)	✓		β-model
Zhang and McCullagh (2015)	✓		Hereditary hypergraph
Lunagómez et al. (2017)	✓		Random geometric graph
Ghoshdastidar and Dukkipati (2017a)	✓		Clustering of nodes
Turnbull et al. (2019)	✓		**Latent space model**
Zhen and Wang (2021)	✓		Clustering and **latent position** of nodes
Chodrow et al. (2021)	✓	✓	Clustering of nodes
Ng and Murphy (2021)	✓	✓	Clustering of hyperedges
Yu and Zhu (2023 +)	✓	✓	**Latent space model**

Gap in literature on hypergraph modeling

In practice, hypergraph data are often projected to a network (i.e., pairwise relations), causing information loss. Direct modeling of hypergraph data is a relatively open area.

Up next: the proposed model

✓ General types of hypergraphs
- edges can have varying numbers of nodes
- a given edge can appear more than once

✓ Node embedding

✓ Edge prediction

Model — notation

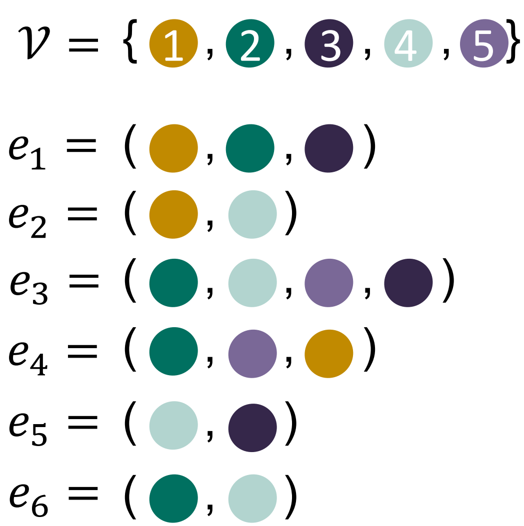
Notation

$\mathcal{V} = \{1,2, \dots, n_v\}$ is the set of all nodes.

Each observed edge is a subset of $\mathcal{V}$.

$e_1, \dots, e_{n_e}$ denote all edges.

Thus, there are n_v nodes and n_e edges.



$n_v = 5, n_e = 6$

Model — motivations from real-world observations

Diversity within each edge

- Nodes in an edge often complement each other, *e.g., different expertise in a collaboration*.
- Diversity also appears when selections are made to prevent redundancy, *e.g., products of various categories in a shopping order*.

Heterogeneous node popularity

- Different nodes appear with very different frequencies.

Model — how to encourage diversity and heterogeneity?

Diversity

- Each node is represented by a vector of latent features
- An edge contains multiple nodes
- *Prob*[observing an edge] is large when the corresponding set of vectors have different “directions”

Heterogeneous popularity

- Each node is associate with a popularity parameter
- *Prob*[observing an edge] is large when nodes in the edge have large popularity parameters

oyster egg

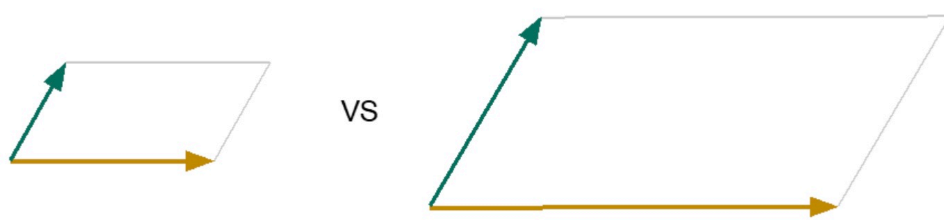
Model — how to encourage diversity and heterogeneity?

$$\tilde{v}_i = \Big(\overbrace{v_{i1}, \dots, v_{id}}^{\text{latent feature vector}},\ 0, \dots, 0, \overset{\text{popularity parameter}}{\overset{\uparrow}{a_i}}, 0, \dots 0\Big) \text{ for node } i$$

How to use $\tilde{v}_i$?

- Let E denote a random edge. $P(E = \{i, j\}) \propto \text{area}^2(\tilde{v}_i, \tilde{v}_j) =$ 

  $P(E = \{i, j\})$ is large when $\tilde{v}_i, \tilde{v}_j$ are long



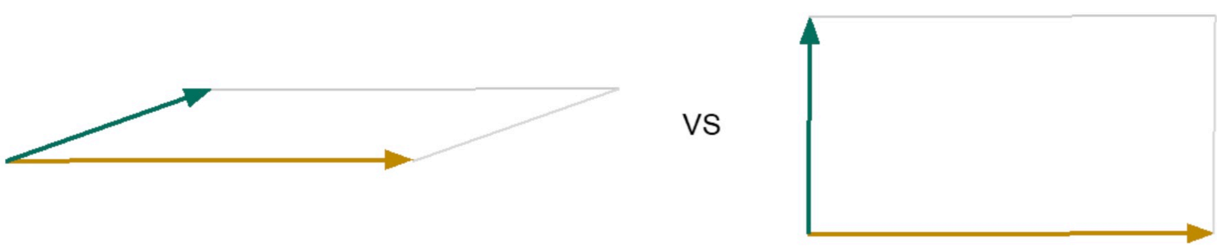
Model — how to encourage diversity and heterogeneity?

$$\tilde{v}_i = (\overbrace{v_{i1,}, \dots, v_{id},}^{\text{latent feature vector}} \quad 0, \dots, 0, \overset{\text{popularity parameter}}{a_i}, 0, \dots 0) \text{ for node } \boldsymbol{i}$$

How to use $\tilde{v}_i$?

- Let E denote a random edge. $P(E = \{i,j\}) \propto \text{area}^2(\tilde{v}_i, \tilde{v}_j) =$ 

  $P(E = \{i,j\})$ is large when $\tilde{v}_i, \tilde{v}_j$ are long and have separated directions, i.e., as close to orthogonal as possible.



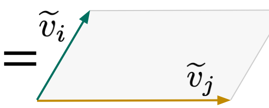
Model — how to encourage diversity and heterogeneity?

latent feature vector popularity parameter

$$\tilde{v}_i = (\overbrace{v_{i1}, \dots, v_{id}}, 0, \dots, 0, a_i, 0, \dots 0) \text{ for node } i$$

live on a sphere pairwise orthogonal

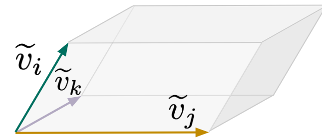
How to use $\tilde{v}_i$?

- Let E denote a random edge. $P(E = \{i, j\}) \propto \text{area}^2(\tilde{v}_i, \tilde{v}_j) =$ 

  $P(E = \{i, j\})$ is large when $\tilde{v}_i, \tilde{v}_j$ are long and have separate directions, i.e., as close to orthogonal as possible.
- $\| v_i \|_2$ is constant across i, $a_i > 0$ is the ith entry of $(0, \dots, 0, a_i, 0, \dots 0)_{1 \times n_v}$

➢ The lengths are driven by the **popularity** parameters a_i, a_j

➢ The separation is mainly driven by the **'diversity'** of the feature vectors v_i, v_j

- $P(E = \{i, j, k\}) \propto \text{volume}^2(\tilde{v}_i, \tilde{v}_j, \tilde{v}_k) =$ 

Model — nice properties

The random edge E can be any subset e of $\mathcal{V}$, e.g., $e = \{1,3\}, e = \{1,3,5,7\}$.

$$P(E = e) = \frac{\text{volume}^2(\tilde{v}_i | i \in e)}{\sum_{e' \subset \mathcal{V}} \text{volume}^2(\tilde{v}_i | i \in e')}$$

Observed edges $e_1, \dots, e_{n_e}$ are i.i.d realizations of P.

Nice properties

- Explicit formulas for marginal probability & conditional probability
  - $P(i \in E)$, e.g., $P(\in E)$
  - $P(e \subset E)$, e.g., $P((, ,) \subset E)$
  - $P(E = e' | e \subset E)$, e.g., $P(E = (, ,) \mid ( ) \subset E)$
- Easy-to-apply sampling algorithm
- Distribution of $|E|$ has an explicit characterization

Method

Model fitting

- Maximum likelihood estimation (MLE)

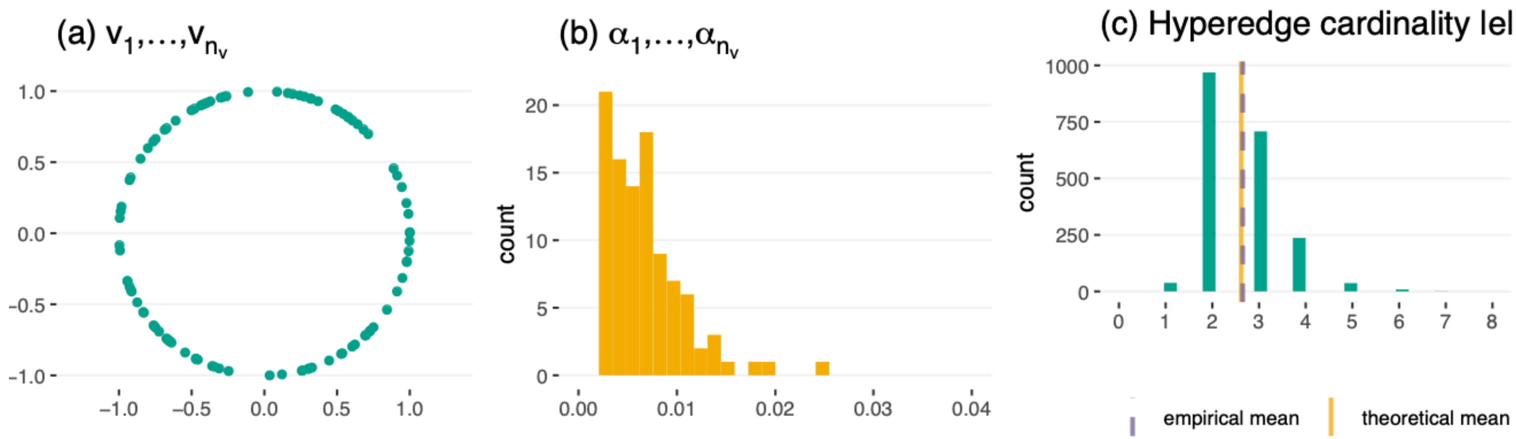
$$\underset{\substack{v_i \in \mathbb{R}^d, a_i > 0, \|v_i\|_2 \\ \text{is constant across } i}}{\arg\max} \quad \underbrace{-\log \sum_{e' \subset \mathcal{V}} \operatorname{volume}^2 (\widetilde{v}_i \mid i \in e')}_{\substack{\text{determinant of a matrix} \\ \text{defined by } v_i, a_i \text{ for all } i}} + \frac{1}{n_e} \sum_{\ell=1}^{n_e} \log \underbrace{\operatorname{volume}^2 (\widetilde{v}_i \mid i \in e_l)}_{\substack{\text{determinant of a matrix} \\ \text{defined by } v_i, a_i \text{ for } i \text{ in } e_l}}$$

- Gradient descent algorithm
  - Accelerated projected gradient methods for nonconvex programming [†]
  - Mini batch gradient descent, i.e., using a small sample of edges in each iteration, and Adam adaptive learning rate

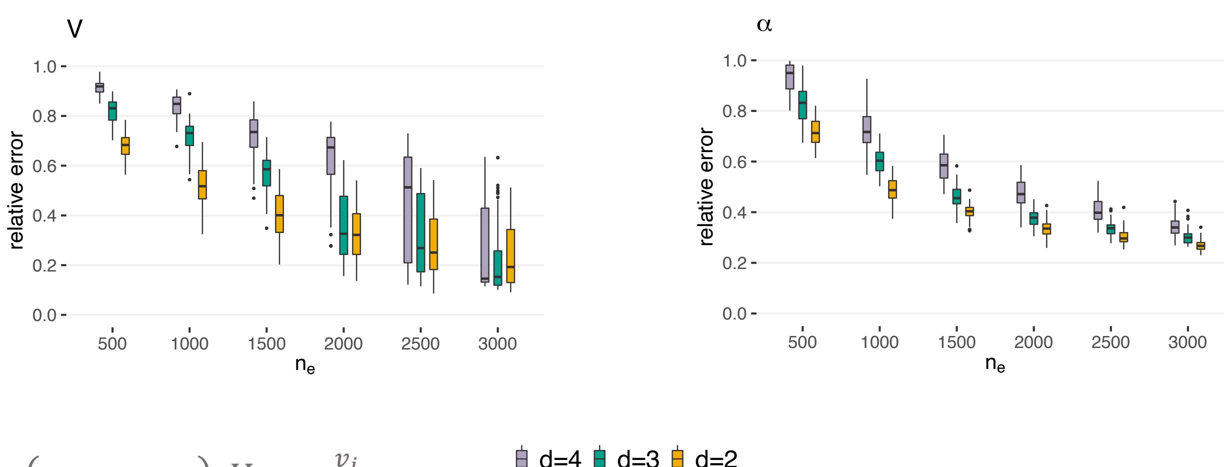
[†] Li & Lin (2015) NeurIPS

Simulations (consistency)

Model setting: n_v=100, $d = 2, n_e = 2000$



Results of 50 repeated simulations: n_v=100, $d = 2, 3, 4$



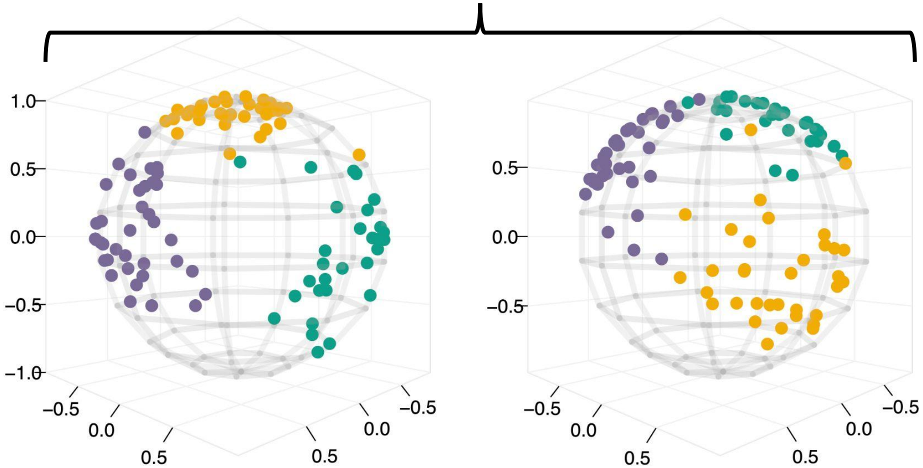
† $\alpha_i = a_i^2, \alpha = (\alpha_1, \ldots, \alpha_{n_v}); V_{i.} = \frac{v_i}{\|v_i\|_2}$.

Simulation (clustering performance)

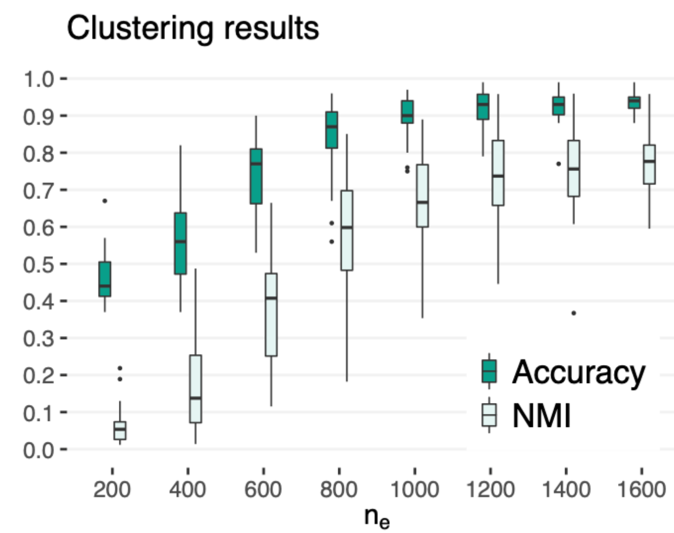
Evaluating clustering performance

- $n_v = 100$ nodes are assigned to three even clusters.
- v_i's are on the unit sphere in $\mathbb{R}^3$ and are generated by three von Mises–Fisher distributions.

color = true cluster label; $n_e = 2000$



$v_1, \dots, v_{100}$ $\hat{v}_1, \dots, \hat{v}_{100}$



Clustering result as n_e varies

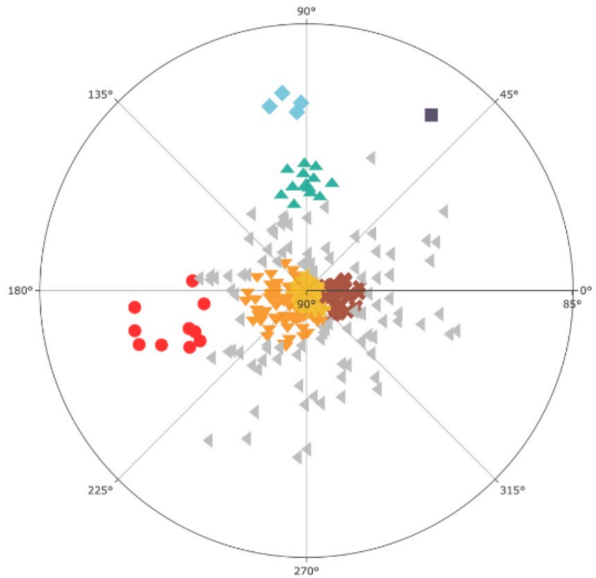
∗ NMI = normalized mutual information

Numerical results — recipe data

Recipes on Yummly.com

$n_e = 2673$ recipes involving $n_v = 906$ ingredients; fit a model with $d = 3$

Estimated latent vectors $\hat{v}_i$ (embeddings)



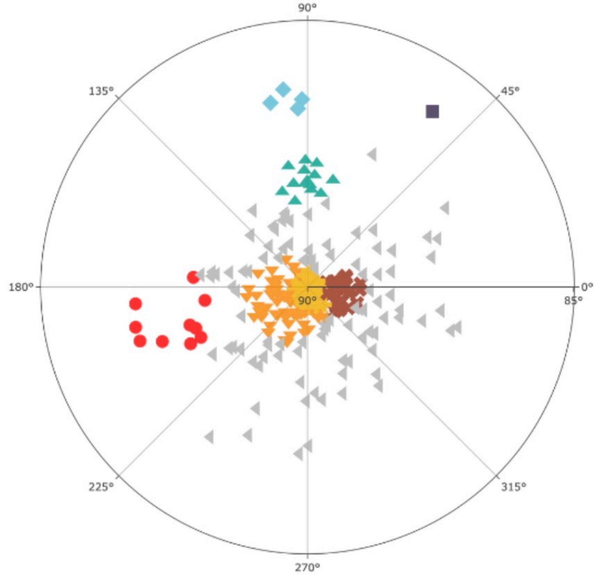
Numerical results — recipe data

Recipes on Yummly.com

$n_e = 2673$ recipes involving $n_v = 906$ ingredients; fit a model with $d = 3$

Applications of the fitted model

- Clustering ingredients using embedding $\hat{v}_i$'s, which lie on a sphere in $\mathbb{R}^3$



Clustering method: fit a **mixture of von Mises–Fisher model** (i.e., analogy to Gaussian on a sphere) on embeddings of **ingredients that appear in 10+ recipes** (298 ingredients in total)

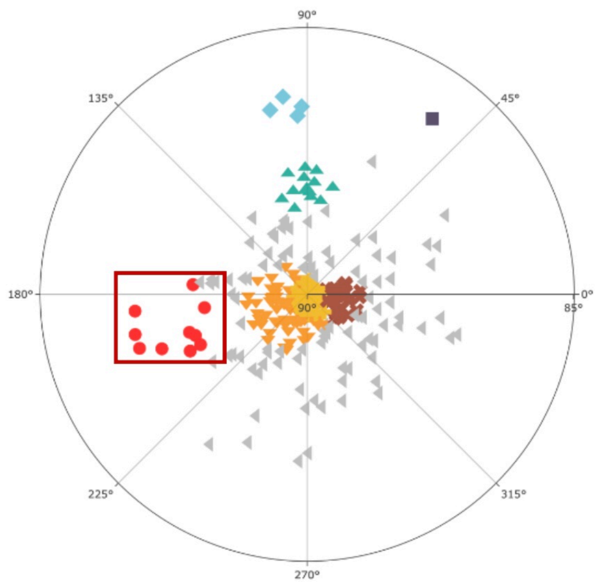
Numerical results — recipe data

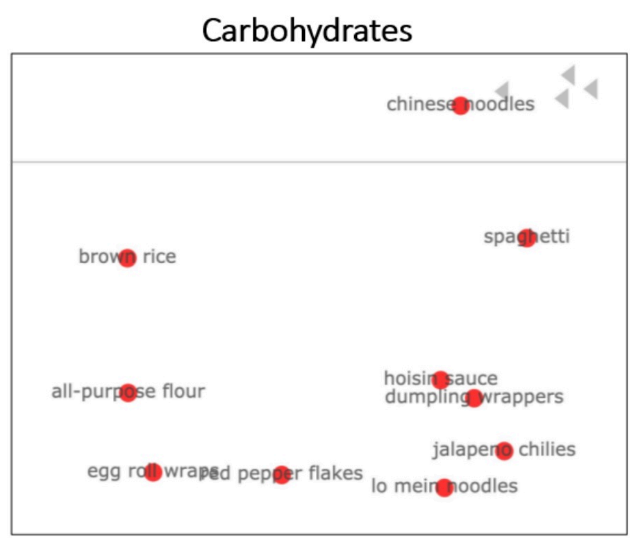
Recipes on Yummly.com

$n_e = 2673$ recipes involving $n_v = 906$ ingredients; fit a model with $d = 3$

Applications of the fitted model

- Clustering ingredients using embedding $\hat{v}_i$'s, which lie on a sphere in $\mathbb{R}^3$





Clustering method: fit a **mixture of von Mises–Fisher model** (i.e., analogy to Gaussian on a sphere) on embeddings of **ingredients that appear in 10+ recipes** (298 ingredients in total)

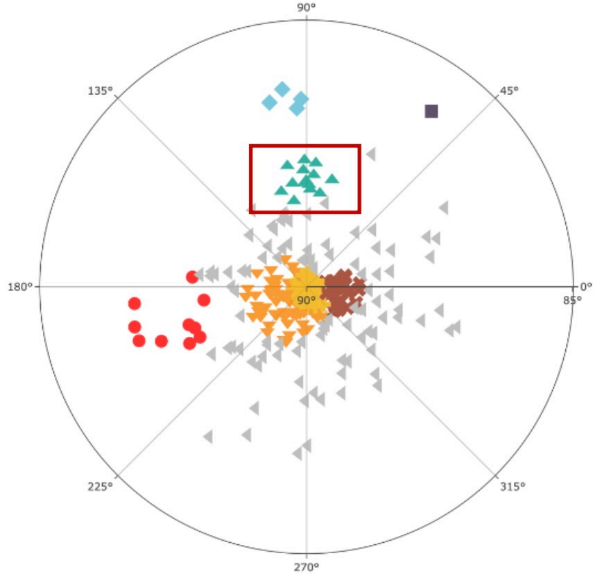
Numerical results — recipe data

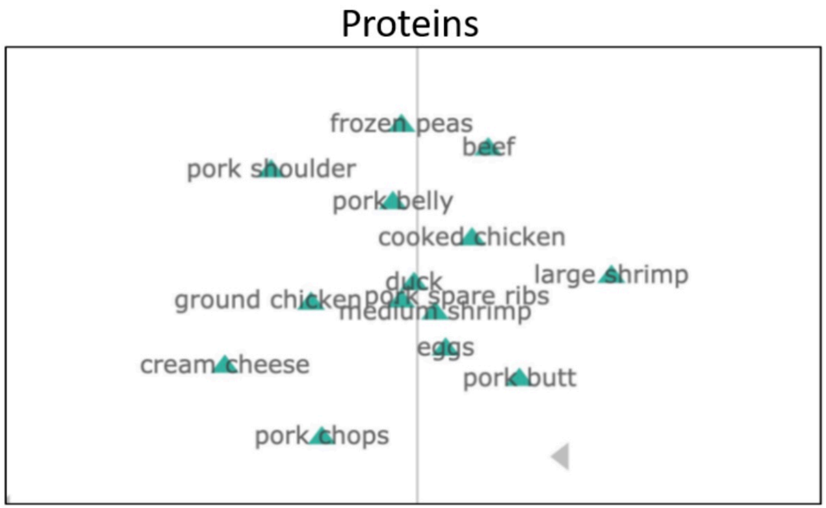
Recipes on Yummly.com

$n_e = 2673$ recipes involving $n_v = 906$ ingredients; fit a model with $d = 3$

Applications of the fitted model

- Clustering ingredients using embedding $\hat{v}_i$'s, which lie on a sphere in $\mathbb{R}^3$





Clustering method: fit a **mixture of von Mises–Fisher model** (i.e., analogy to Gaussian on a sphere) on embeddings of **ingredients that appear in 10+ recipes** (298 ingredients in total)

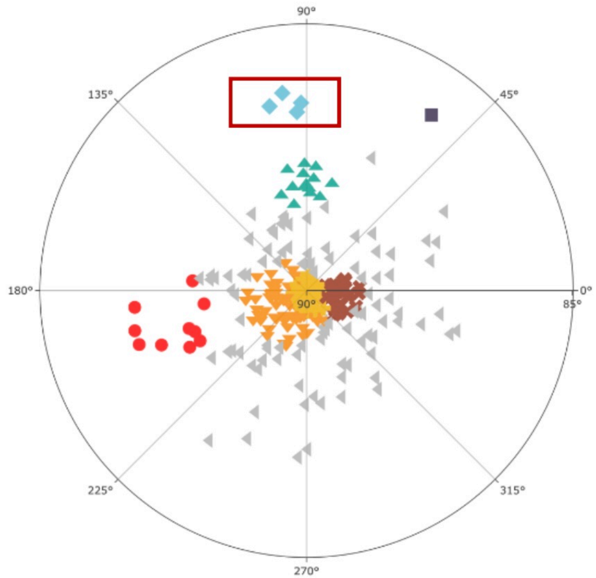
Numerical results — recipe data

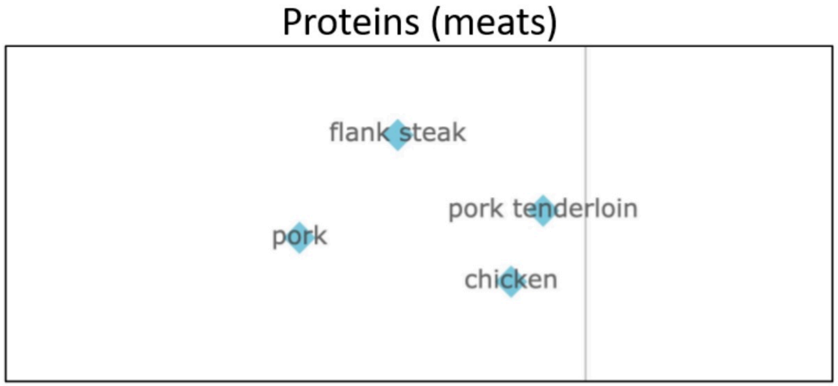
Recipes on Yummly.com

$n_e = 2673$ recipes involving $n_v = 906$ ingredients; fit a model with $d = 3$

Applications of the fitted model

- Clustering ingredients using embedding $\hat{v}_i$'s, which lie on a sphere in $\mathbb{R}^3$





Clustering method: fit a **mixture of von Mises–Fisher model** (i.e., analogy to Gaussian on a sphere) on embeddings of **ingredients that appear in 10+ recipes** (298 ingredients in total)

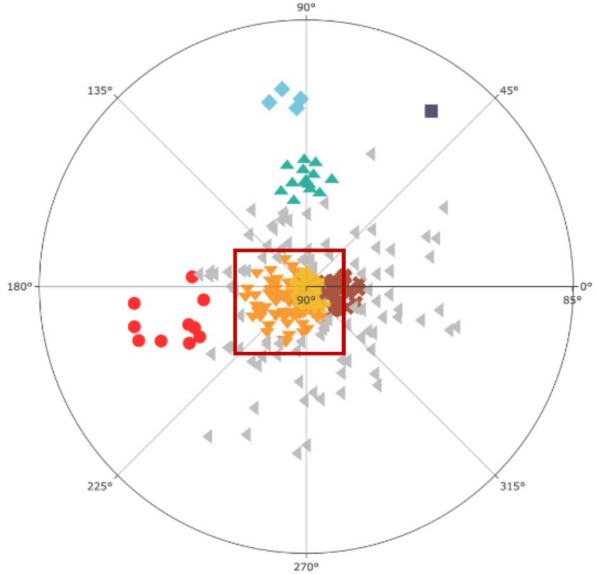
Numerical results — recipe data

Recipes on Yummly.com

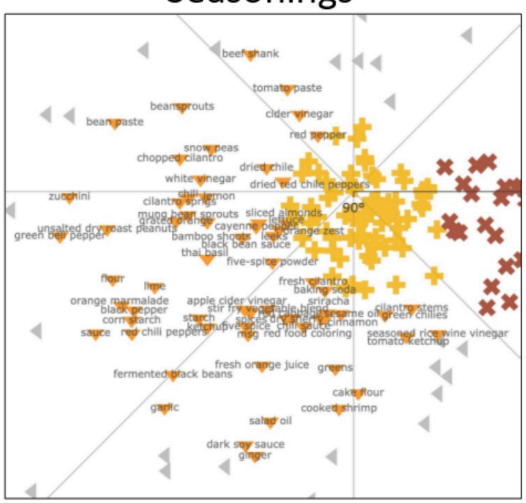
$n_e = 2673$ recipes involving $n_v = 906$ ingredients; fit a model with $d = 3$

Applications of the fitted model

- Clustering ingredients using embedding $\hat{v}_i$'s, which lie on a sphere in $\mathbb{R}^3$



Seasonings



Clustering method: fit a **mixture of von Mises–Fisher model** (i.e., analogy to Gaussian on a sphere) on embeddings of **ingredients that appear in 10+ recipes** (298 ingredients in total)

Numerical results — recipe data

Applications of the fitted model

- Completing recipes
- e.g., (🍗, 🫑, 💧, 💧, 🥣, ?)

e

```
> selected=c("pork belly","green bell pepper","cooking oil","soy sauce","sugar")
> recommend_one_ingredient(L_hat,selected,ingredients)
Knowing that a recipe has (PORK BELLY, GREEN BELL PEPPER, COOKING OIL, SOY SAUCE, SUGAR),
the one additional ingredient that is most likely to be in this recipe is GARLIC.
```

$$\boxed{\arg\max_{i \notin e} \widehat{P}(i \in E | e \subset E)}$$

$\hat{P}$ is parameterized by $\hat{v}_i$ and $\hat{a}_i$ for all i

The probability that ingredient i is in a recipe, given that ingredients in the selected set e are in the recipe

Theory — consistency

Proposition (Identifiability)

If $n_v > 2d$, then, given any fixed model, $a = (a_1, \cdots, a_{n_v})$ is identifiable and $v_1, \cdots, v_{n_v}$ are identifiable up to a shared rotation and individual sign flips (i.e., multiplication with ± 1).

Theorem 1 (Consistency)

If $n_v > 2d$ and $\{v_1, \cdots, v_{n_v}\}$ span $\mathbb{R}^d$, then, as $n_e \to \infty$, with proper rotation and sign flips of $\hat{v}_i$

$$\sum_{i=1}^{n_v} \|\hat{v}_i - v_i\|_2 \xrightarrow{p} 0,$$

$$\|\hat{a} - a\|_2 \xrightarrow{p} 0.$$

Theory — asymptotic distribution

Parameterize the model using matrix $\boldsymbol{L}$

- The proposed model is a special determinantal point process (DPP)
- A DPP is defined using a matrix L (L can be any semi-definite matrix)

* For the proposed model, $L = \left(v_i^T v_j\right)_{i,j=1}^{n_v} + \text{diag}(a_1^2, \dots, a_{n_v}^2)$

Theorem 2 (Asymptotic normality)

If $n_v > 2d$, $\{v_1, \cdots, v_{n_v}\}$ span $\mathbb{R}^d$ and can not be divided into multiple groups that are mutually orthogonal, then, as $n_e \to \infty$, with proper sign flips of $\hat{v}_i$, $\|\widehat{L} - L\|_F \to 0$ in probability and

$$\sqrt{n_e} \cdot \text{vec}(\widehat{L} - L) \xrightarrow{dist.} N(\mathbf{0}, \Sigma).$$

Here Σ is a matrix that we have derived which is defined by v_i, a_i, $i \in \mathcal{V}$.

Theory — asymptotic result (challenges)

- This asymptotic result is one regarding constrained M-estimation, since the parameter space of L is special.
- The theoretical development requires non-trivial analysis of the local geometry of this parameter space, where we applied recent results in variational geometry.
- This is the first asymptotic result, to our knowledge, for structured determinantal point processes.

Take home messages

- The proposed model is the first hypergraph model that
  - considers diversity
  - enables embedding while allowing edges to have different numbers of nodes and to appear more than once in data

- The model can be applied for
  - node embedding
  - node clustering
  - edge prediction

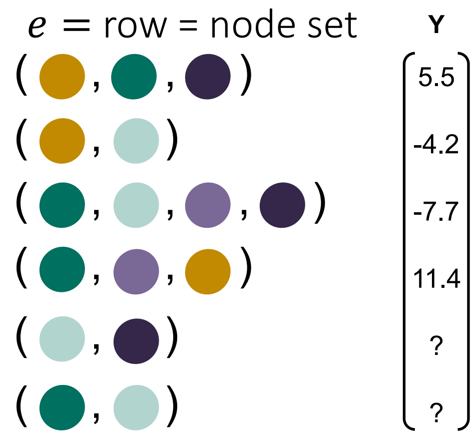
- We have established the consistency and asymptotic normality of the estimates of model parameters.

Ornes, Stephen. "How Big Data Carried Graph Theory Into New Dimensions." Quanta Magazine (Aug 2021).

Planned future work on hypergraph data

Develop regression model on node set

- Consider $y = f(e)$ for $e \subset \mathcal{V}$ and estimate f
- e.g., $\hat{f}$ predicts quality of collaboration for a given team, future medical needs given current medical codes

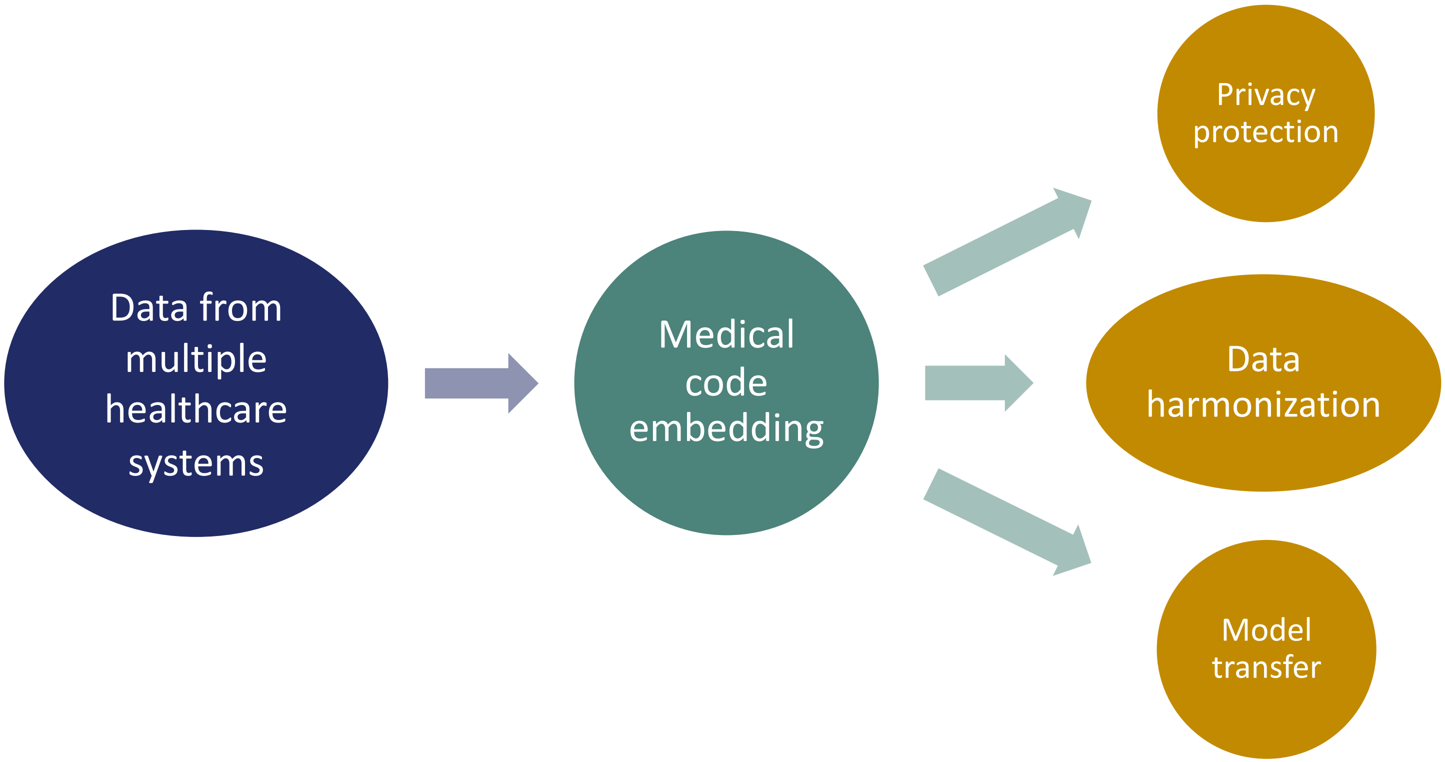


Extend the hypergraph model to incorporate

- observed covariates of nodes
- more complex mechanism beside diversity and popularity
- information on nodes' different roles within edges

Planned future work on hypergraph data

Application to distributed EHR data network



Thank you

References

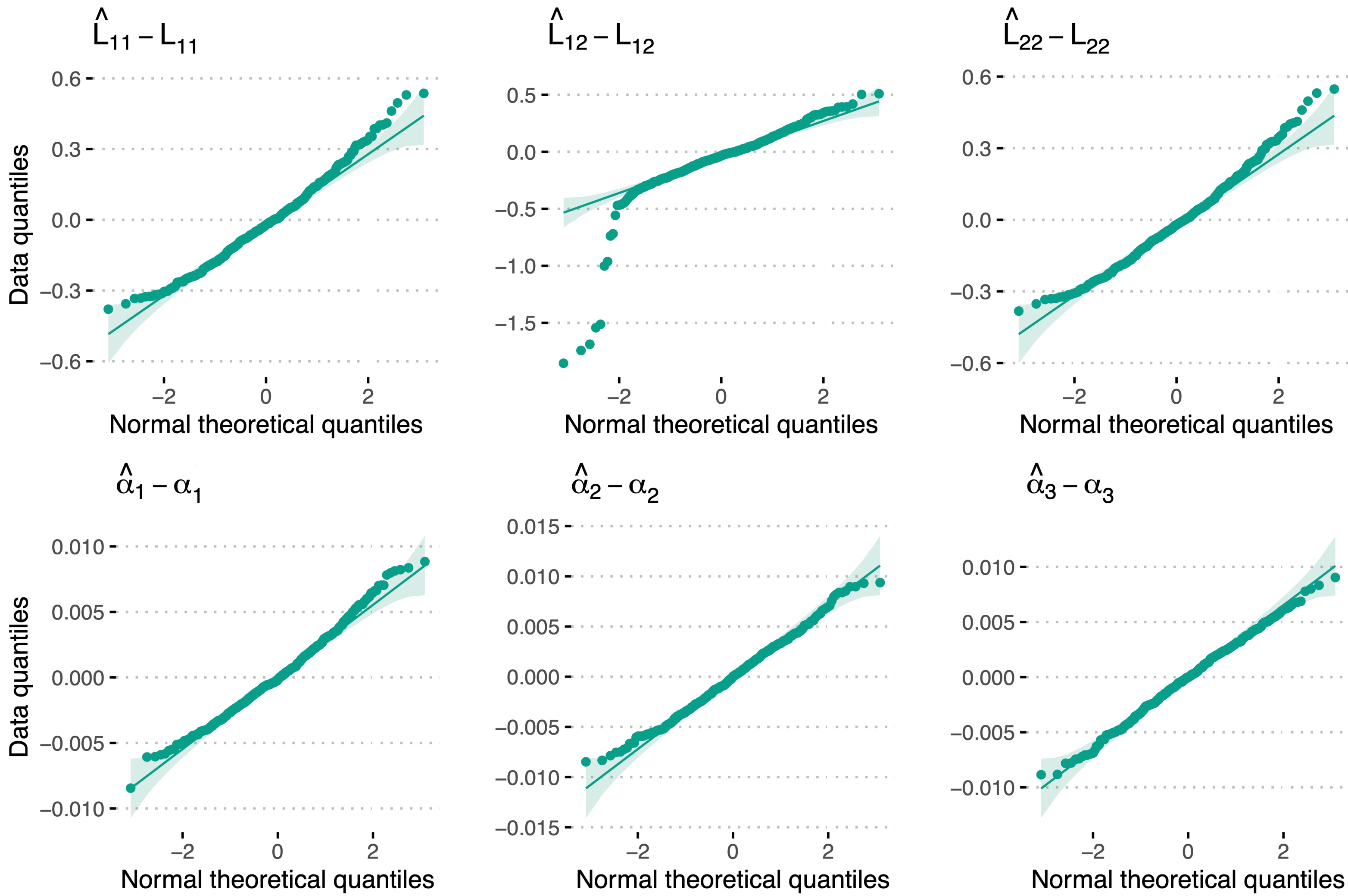
- Kulesza, A. and Taskar, B. (2012). Determinantal point processes for machine learning. Foundations and Trends in Machine Learning, 5(2–3).
- Brunel, V.-E., Moitra, A., Rigollet, P., and Urschel, J. (2017). Rates of estimation for determinantal point processes. In Conference on Learning Theory, pages 343–345. PMLR.
- Gartrell, M., Paquet, U., and Koenigstein, N. (2017). Low-rank factorization of determinantal point processes. In Thirty-First AAAI Conference on Artificial Intelligence.
- Li, H. and Lin, Z. (2015). Accelerated proximal gradient methods for nonconvex programming. Advances in neural information processing systems, 28:379–387.
- Ghoshdastidar, D. and Dukkipati, A. (2014). Consistency of spectral partitioning of uniform hypergraphs under planted partition model. Advances in Neural Information Processing Systems, 27:397–405.
- Ghoshdastidar, D. and Dukkipati, A. (2017b). Uniform hypergraph partitioning: Provable tensor methods and sampling techniques. The Journal of Machine Learning Research, 18(1):1638–1678.
- Chien, I., Lin, C.-Y., and Wang, I.-H. (2018). Community detection in hypergraphs: Optimal statistical limit and efficient algorithms. In International Conference on Artificial Intelligence and Statistics, pages 871–879. PMLR.
- Kim, C., Bandeira, A. S., and Goemans, M. X. (2018). Stochastic block model for hypergraphs: Statistical limits and a semidefinite programming approach. arXiv preprint arXiv:1807.02884.
- Ke, Z. T., Shi, F., and Xia, D. (2019). Community detection for hypergraph networks via regularized tensor power iteration. arXiv preprint arXiv:1909.06503.
- Lyu, Z., Xia, D., and Zhang, Y. (2021). Latent space model for higher-order networks and generalized tensor decomposition. arXiv preprint arXiv:2106.16042.
- Yuan, Y. and Qu, A. (2021). High-order joint embedding for multi-level link prediction. Journal of the American Statistical Association, (just-accepted):1–39. Lunagomez, S., Mukherjee, S., Wolpert, R. L., and Airoldi, E. M. (2017). Geometric representations of random hypergraphs. Journal of the American Statistical Association, 112(517):363–383.

References

- Stasi, D., Sadeghi, K., Rinaldo, A., Petroví c, S., and Fienberg, S. E. (2014). beta models for random hypergraphs with a given degree sequence. arXiv preprint arXiv:1407.1004.
- Zhang, D. and McCullagh, P. (2015). Exchangeable random hypergraphs. working paper.
- Ghoshdastidar, D. and Dukkipati, A. (2017a). Consistency of spectral hypergraph partitioning under planted partition model. The Annals of Statistics, 45(1):289–315.
- Turnbull, K., Lunagomez Coria, S., Nemeth, C., and Airoldi, E. (2019). Latent space representations of hypergraphs. arxiv. org.
- Chodrow, P. S., Veldt, N., and Benson, A. R. (2021). Generative hypergraph clustering: from blockmodels to modularity. arXiv preprint arXiv:2101.09611.
- Ng, T. L. J. and Murphy, T. B. (2021). Model-based clustering for random hypergraphs. Advances in Data Analysis and Classification, pages 1–33.
- Zhen, Y. and Wang, J. (2021). Community detection in general hypergraph via graph embedding. arXiv preprint arXiv:2103.15035.

Appendix – simulations (asymptotic normality)

Results of 500 repeated simulations: n_v=100, $d = 2, n_e = 3000$



Appendix – simulations (asymptotic normality)

Results of 500 repeated simulations: n_v=100, $d = 2, n_e = 3000$