# Difficulties of tensor NN and brief simulations for polynomial time algorithms

Chanwoo Lee, July 8, 2021

## 1 The difficulty of tensor NN

My original trial of extension from [2] is to estimate the probability tensor by

$$\hat{P}_{ij} := \frac{1}{|\mathcal{N}_i||\mathcal{N}_j|} \sum_{i' \in \mathcal{N}_i, j' \in \mathcal{N}_j} A_{i'j'},$$

where $\mathcal{N}_i$ and $\mathcal{N}_j$ are neighbors of $i$ and $j$-th nodes respectively which should be defined well. Then estimation error is decomposed as

$$\begin{aligned}
&\frac{1}{d^2} \sum_{i,j \in [d]} \left\{ \hat{P}_{ij} - P_{ij} \right\}^2 \\
&= \frac{1}{d^2} \sum_{i,j \in [d]} \left\{ \frac{\sum_{i' \in \mathcal{N}_i, j' \in \mathcal{N}_j} P_{i'j'} - P_{ij}}{|\mathcal{N}_i||\mathcal{N}_j|} \right\}^2 \\
&\lesssim \frac{1}{d^2} \left\{ \underbrace{\sum_{i,j \in [d]} \left( \frac{\sum_{i' \in \mathcal{N}_i, j' \in \mathcal{N}_j} P_{i'j'} - A_{i'j'}}{|\mathcal{N}_i||\mathcal{N}_j|} \right)^2}_{\text{variance}} + \underbrace{\sum_{i,j \in [d]} (A_{ij} - P_{ij})^2}_{\text{variance}} + \underbrace{\sum_{i,j \in [d]} \left( \frac{\sum_{i' \in \mathcal{N}_i, j' \in \mathcal{N}_j} A_{i'j'} - A_{ij}}{|\mathcal{N}_i||\mathcal{N}_j|} \right)^2}_{\text{block error}} \right\} \\
&\lesssim \underbrace{\left( \frac{1}{h^2} \right)}_{\text{variance}} + \underbrace{\sum_{i,j \in [d]} \left( \frac{\sum_{i' \in \mathcal{N}_i, j' \in \mathcal{N}_j} A_{i'j'} - A_{ij}}{|\mathcal{N}_i||\mathcal{N}_j|} \right)^2}_{\text{block error}}, \qquad (1)
\end{aligned}$$

block error and noise error. b.c. A is random

In analogy to Lemmas 1-4 in SAS algorithm. the ``block error'' corresponds to Lemma 1.

where $|\mathcal{N}_i| = \mathcal{O}(h)$ and the the variance term is bounded by Hoeffding's inequality. However, to bound the block error, we need several techniques that has some difficulties.

**Assumption 1** (Pieacewise Lipchitz graphon family)**.** There exists an integer $K \geq 1$ and a sequence $0 \leq x_0 < x_1 < \cdots < x_K = 1$, satisfying $\min_k |I_k|/\sqrt{\log d/d} \to \infty$ and $f$ is a bi-Lipchitz function on $I_k \times I_\ell$ for all $1 \leq k, \ell, \leq K$ where $I_k := [x_{k-1}, x_k]$

For any $\xi \in [0,1]$, let $I(\xi)$ denote the $I_k$ that contains $\xi$. Define $s_i(\Delta) = [\xi_i - \Delta, \xi_i + \Delta] \cap I(\xi_i)$.

**Lemma 1.** Let $\{\xi_i\}_{i \in [d]}$ are random variables i.i.d. drawn from $U[0,1]$. Define $\Delta_d = \{c_1 + (\tilde{c}_1 + 4)^{1/2} \sqrt{\frac{\log d}{d}} < \frac{\min_k |I_k|}{2}$, then

$$\mathbb{P}\left( \min_i \frac{\{i' \neq i \colon \xi_{i'} \in S_i(\Delta_d)\}}{d-1} \geq c_1 \sqrt{\frac{\log d}{d}} \right) \leq 1 - 2d^{-\tilde{c}_1/4}$$

We can think of $S_i(\Delta)$ as true neighbor of $\xi_i$. Lemma 1 makes sure that the proportion of elements in $\tilde{S}_i(\Delta_d) := \{i' \neq i \colon \xi_{i'} \in S_i(\Delta_d)\}$, a neighbor of $\xi$, is greater than $\sqrt{\frac{\log d}{d}}$.

Good observation. You essentially sketched out the proof in Han's paper ``exact clustering…"

We might need extra assumptions to bound (4). This is exactly why earlier algorithm needs assumptions:
1) sufficiently distinct degree (Airoldi); or 2) sufficiently distinct blocks (in Han's SBM paper)
to provide extra information for neighbor partition that satisfy (4).

Notice that for true $\tilde{S}_i(\Delta)$, we are able to bound the block error

$$\sum_{i,j\in[d]} \left( \frac{\sum_{i'\in S_i(\Delta_d), j'\in S_j(\Delta_d)} A_{i'j'} - A_{ij}}{|S_i(\Delta_d)||S_j(\Delta_d)|} \right)^2$$

by the following decomposition

$$|A_{ij} - A_{i'j'}| \le |P_{ij} - P_{i'j'}| + |A_{i'j'} - P_{i'j'}| + |A_{ij} - P_{ij}|, \tag{2}$$

where the last two term is easily bounded by Hoeffding's inequality. For the first term, we can use the Lipchitz assumption as

$$|P_{ij} - P_{i'j'}| \le |f(\xi_i, \xi_j) - f(\xi_i, \xi_{j'})| + |f(\xi_i, \xi_{j'}) - f(\xi_{i'}, \xi_{j'})| \le 2L\Delta_d, \tag{3}$$

where the last inequality is from Lipchitz continuity and definition of $\tilde{S}_i(\Delta_d)$.

If we find neighbors $(\mathcal{N}_1, \ldots, \mathcal{N}_d)$ whose the number of elements is the same order of $|S_i(\Delta_d)|$ (Notice $|\tilde{S}_i(\Delta_d) \times \tilde{S}_j(\Delta_d)| \asymp d\log d$) and satisfies,

$$\sum_{i,j\in[d]} \left( \sum_{i'\in\mathcal{N}_i, j'\in\mathcal{N}_j} |A_{ij} - A_{i'j'}| \right)^2 \le \sum_{i,j\in[d]} \left( \sum_{i'\in\tilde{S}_i(\Delta_d), j'\in\tilde{S}_j(\Delta_d)} |A_{ij} - A_{i'j'}| \right)^2, \tag{4}$$

Then by the upper bound of block error with true neighbors, we are able to bound the (1).

To summarize, the main idea is to find neighbors $(\mathcal{N}_1, \ldots, \mathcal{N}_d)$ that has small block error that can be bounded by the block error with true neighbors.

**Task:** How can we estimate neighbors $\{\mathcal{N}_i\}_{i=1}^d$ that satisfies (4)? I found it not an easy task.

**Remark 1** (Previous approach)**.** [2] only focuses on row-wise distance because (4) becomes

$$\sum_{i,j\in[d]} \left( \sum_{i'\in\mathcal{N}_i} |A_{ij} - A_{i'j}| \right)^2 = \sum_{i\in[d]} \sum_{i'\in\mathcal{N}_i} \|A_{i\cdot} - A_{i'\cdot}\|_F^2 \le \underbrace{\sum_{i\in[d]} \sum_{i'\in S_i(\Delta_d)} \|A_{i\cdot} - A_{i'\cdot}\|_F^2}_{(*)},$$

where the last inequality is from the definition of $\mathcal{N}_i := \{i' \ne i : \|A_{i\cdot} - A_{i'\cdot}\|_F^2 \le q(h/d)\}$ and $|\tilde{S}_i(\Delta_d)| \asymp h$. Notice $(*)$ is easily bounded based on the decompositions similar to (2) and (3).

**Remark 2** (Tensor generalization based on [2])**.** I found a recent paper [1] which extends NN method to tensor case. This paper unfolds tensors to matrices and follows the exactly the same row-wise distance approach to estimate the probability tensor. I know the paper. The results are not optimal though.

## 2 Simulations for polynomial time algorithms

I have performed brief simulations for polynomial time algorithms. I compare three different algorithms: `SAS` is smooth and sorting algorithm, `Spectral` is truncated SVD algorithm, and `SBM` is stochastic block approximation algorihtm based on `functions_sbm`. Detailed simulation setting is as follows

Wainwright's paper provides an algorithm for which (4) holds in their contexts (monotonic but nonsmooth matrix). The technique might be applicable to our case but is rather complicated.



Let's focus spectral algorithm in this paper, and defer the NN algorithm (open even for matrix case) to the 2nd paper.

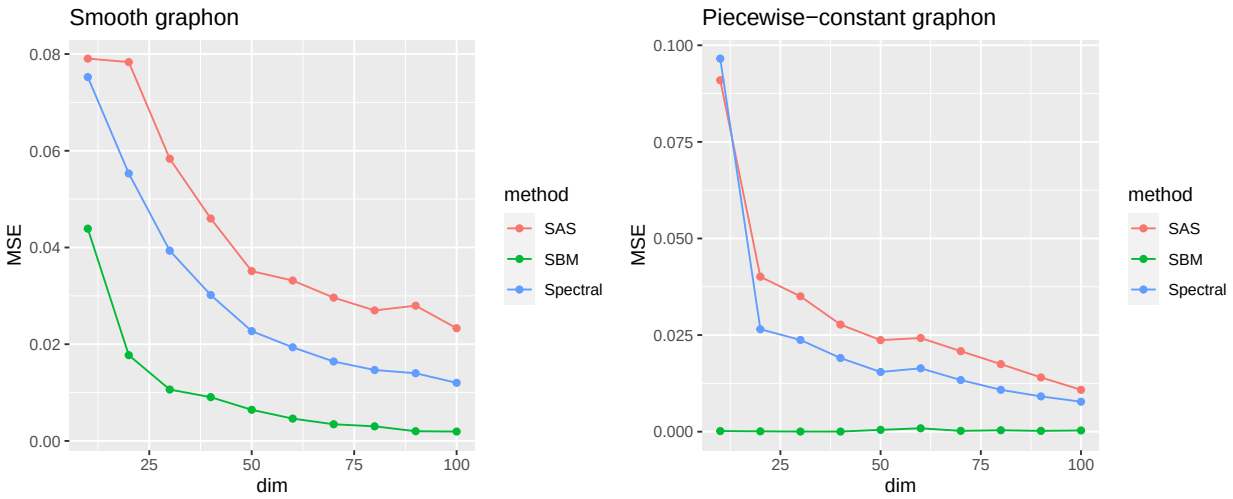


Figure 1: Mean squared error of probability tensor estimation depending on tensor dimension $d$ for three different methods: SAS,SBM, and Spectral.

1. (Smooth graphon model)

$$A_{ijk} = \text{Bernoulli}(\Theta_{ijk}), \quad \text{and } \Theta_{ijk} = \frac{1}{1+\exp(-(\xi_i^2+\xi_j^2+\xi_k^2))},$$

where $\xi_\ell \overset{\text{i.i.d}}{\sim} U[0,1]$ for all $\ell \in [d]$.

Try fixed design over gird {1/d, 2/d, ...} for now.
plot the marginal degree. Is it monotonic?
If not, design functions that have monotonic degree.
refer to the simulation section in NN paper.

2. (Piece-wise constant graphon model)

$$A_{ijk} = \text{Bernoulli}(\Theta_{ijk}), \quad \text{and } \Theta_{ijk} = Q_{z^{-1}(i,j,k)},$$

,where $Q \in [0,1]^{K\times K\times K}$ and entries of $Q$ are i.i.d. drawn from $U[0,1]$. In simulations, I set $K = d/2$ for $d = \{10, 20, \ldots, 100\}$.

Since `SAS` and `SBM` require the number of group $k$, I set $k = \lceil d^{\frac{m}{m+2}} \rceil$.

Figure 1 shows the performance of each algorithm according to two different settings for $d \in \{10, 20, \ldots, 100\}$. The results are consistent regardless of simulation settings: smooth graphon and piece-wise smooth graphon. `SBM` performs the best while `SAS` shows the worst performance across most of the scenarios. In addition, I have checked the computing time under the smooth graphon model simulation when $d = 100$. It shows that `SAS` and `Spectral` algorithms are much faster than `SBM` as one can see in the following table.

| Method | `SAS` | `Spectral` | `SBM` |
|---|---|---|---|
| Computing time | 0.609 sec | 0.371 sec | 4.545 sec |

## References

[1] Yihua Li, Devavrat Shah, Dogyoon Song, and Christina Lee Yu. Nearest neighbors for matrix estimation interpreted as blind regression for latent variable model. *IEEE Transactions on Information Theory*, 66(3):1760–1784, 2019.

[2] Yuan Zhang, Elizaveta Levina, and Ji Zhu. Estimating network edge probabilities by neighborhood smoothing. *arXiv preprint arXiv:1509.08588*, 2015.